AMSTRAD

EXAKOMP



KULLANMA KILAVUZU

İÇİNDEKİLER

KONU 1
TEMEL KURS

Kurma
Çevre elemanlarının bağlanması
Disk hakkında
Klavyeye alışma
Yazılım ve 'Welcome' programının yüklenmesi
BASIC komutlara giriş
Disk işlemlerine giriş
Modlar, renkler ve grafikler
Ses
AMSDOS ve CP/M'e giriş
Bellek bankası yöntemine giriş

KONU 2
Temel kursun ilerisinde

Basit bir program yazalım
Çözüm ve düşünceler
Dizi kullanımı
Bir menü tanıtımı
Diske değişkenleri yükleme ve kaydetme
Düzeltme ve satır numaraları
Programın son şekli

KONU 3
Kullanılan notasyonların tanıtımı Aşağıdaki komutları içeren komutların alfabetik dizimi.

Komut kullanış şekli
örnek
Açıklama
özel notlar
Bağlı olduğu komutlar

KONU 4
Disk ve kaset kullanımı

Esas diskin kopyasını çıkarmak
CP/M plus ile başlarken
Help'in kullanılması
Tek ve çift disket kullanımı
Dosyaların kopyalanması
BASIC için sadece disk
GSX ile başlarken
CP/M 2,2
Kasetle çalışma

KONU 5
AMSDOS ve CP/M

AMSDOS:
- AMSDOS'a giriş
- Disk kataloğu
- Dosya isimleri, tipleri ve başlıkları
- Rastgele kartlar
- AMSDOS komutlarının özeti
- Dosyaların kullanımı ve kopyalanması

CP/M:
- CP/M'in tanıtımı
- CP/M Plus'ı yükleme
- Direkt konsol komutları
- Geçiş komutları
- Çevre idaresi
- CP/M 2,2 ile çalışma

KONU 6
LOGO diline giriş

LOGO nedir?
Dr, LOGO yöntemleri
Programları ve yöntemleri düzeltme
Kullanım yardımları
Dr, LOGO komutlarını açıklama
Kelime ve Listeleme komutları
Aritmetik işlemler
Mantık işlemler

Değişkenler
Prosedürler
Düzeltme
Yazıcı fonksiyonları
Metin ekranı
Grafik ekran
Kaplumbağa grafikleri
Çalışma alanı düzenlemesi
Destekleme listeleri
Disk dosyaları
Tuş takımı ve oyun çubuğu
Ses
Kontrol çıkışı
Özel kullanımlar
Sistem ilkeleri
Sistem değişkenleri
Sistem desteklemeleri

KONU 7
Bilginiz için

BASIC kursör yerleri ve kontrol kod uzantıları
Kesintiler (interrupts)
ASCII ve grafik karakterler
Tuş referansları
Ses
Hata mesajları
BASIC komutları listesi
Planlar
Bağlantılar
Yazıcılar
Oyun çubukları
Disk organizyonu
Yerleşmiş sistem uzantıları (RSX)
Hafıza
CP/M Plus terminal benzeticisi
CP/M Plus karakter seti

KONU 8
Bellek bankası yöntemi

İkinci 64K hafızayı kullanma
Ekran görüntülerini depolama

Bellek bankasının kullanılması
Ekranların kaydırılması
Ekranların kopyalanması
Sahte dosya işlemleri
Bir RAM disk yaratma
Şimdiki kayıt
Dizilerin okunması, yazılması ve aranması
RAM dosyalarının örnek programları

KONU 9
Boş vaktinizde

Genel:
Mikrobilgisayarların dünyası
Yazılım ve donanım
Bilgisayarların karşılaştırılması
Yapılan bazı kavram yanlışlıkları
Bilgisayar sizin komutlarınızı nasıl anlar
Digital dünyası
Bit'ler ve Byte'ler
İkili sistem
Onaltılı sistem
CPC 6128'in özellikleri:
Karakter seti
Değişkenler
Mantık
Kullanıcı tarafından tanımlanan karakterler
Yazı formatlanması
Pencereler
Kesmeler
Veri
Ses
Grafik
İkinci 64K hafızayı kullanarak grafik
Screen designer programı

EKLER

EK 1 : Son kullanıcı programı ruhsat anlaşması
EK 2 : Terminal sözlüğü
EK 3 : Bazı programlar
EK 4 : INDEX

KONU 1
TEMEL KURS

Bölüm 1: Kurma....

CPC 6128 aşağıdaki şekillerde kurulabilir ve kullanılabilir.

1. AMSTRAD GT 65 Yeşil tüplü monitör ile.
2. AMSTRAD CTM 644 Renkli monitör ile.
3. AMSTRAD MP2 Modülatör/Güç kaynağı ve standart ev tipi (UHF) televizyon ile.

Fişe takma

CPC 6128 220-240 V/50 Hz AC şehir şebeke ceryanı ile çalışır. GT 65, GT 644 veya MP2 için uygun cereyan fiş ve soketi kullanın.

Önemli

Cereyan kablosundaki renk kodlaması şu şekilde yapılmıştır.

Mavi: Nötr
Kahverengi: Canlı

Bilgisayar kullanılmadığı zaman fişi çekiniz.
Bigisayarın kasasını, monitörü veya modülatör/güç kaynağını açmayın ve vidalarını gevşetmeyin. CPC 6128 ve MP2'nin kasasının altında yer almış ve GT 65 veya CTM 644 kutusunun arkasında yer almış ikazlara daima uyunuz.

İKAZ: HİÇ BİR VİDAYI SÖKMEYİN. GERİLİM VAR

Bilgisayarın Monitöre Bağlanması

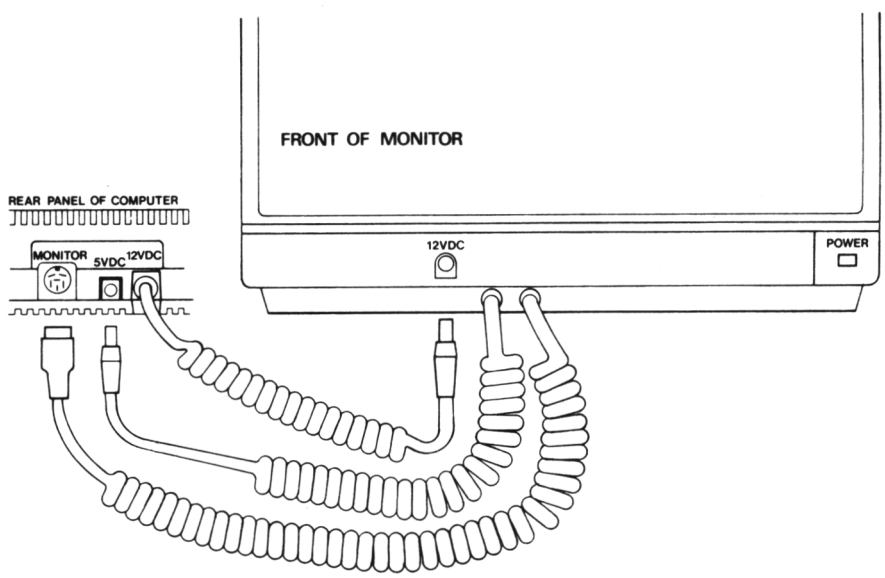
(Eğer CPC 6128'i MP2 güç kaynağı ile kullanıyorsanız bu bölümü atlayıp bir sonraki bölüme geçiniz.)

1. Monitörün şehir şebeke ceryanına bağlı olmadığından emin olun.

2. Monitörün önünden çıkan kablonun ucundaki büyük fişi (6-uçlu DIN), bilgisayarın arkasında, MONİTÖR yazılı sokete sokunuz.

3. Monitörün önünden çıkan kablonun ucundaki küçük fişi (5 V DC), bilgisayarın arkasında, 5 V DC yazılı sokete sokunuz.

4. Bilgisayarın arkasından çıkan kablonun ucundaki küçük fişi (12 VDC), monitörün önündeki sokete sokunuz.



Bilgisayarı MP2 Modülatör/Güçkaynağı Ünitesine Bağlama

MP2, halen kullanılmakta olduğunuz GT65 Yeşil monitörlü CPC 6128 bilgisayarın isteğe bağlı olarak alınabilecek ilave bir parçasıdır. MP2 sayesinde bilgisayarınızı, ev tipi renkli televizyon ile kullanabilir ve böylelikle CPC 6128 bilgisayarın bütün renk fonksiyonlarından faydalanabilirsiniz.

MP2 cihazınızı koyacağınız yerin CPC 6128 bilgisayarının sağ yanı olmasını tavsiye ederiz.

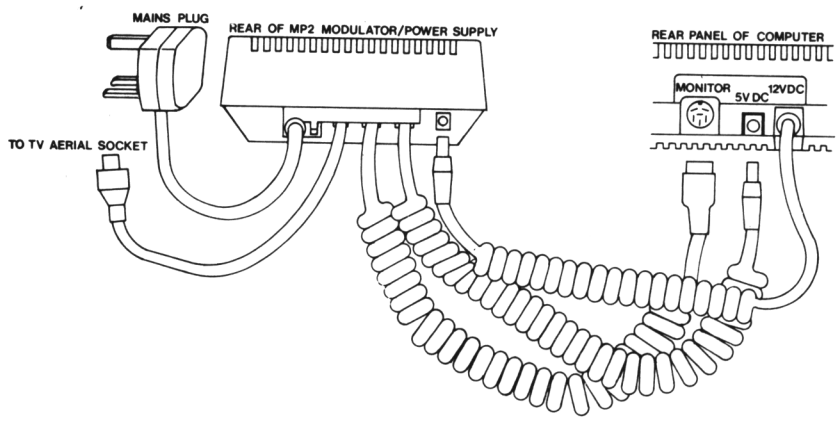
1. MP2 cihazının fişinin şehir şebeke prizine takılı olmamasına dikkat ediniz.

2. MP2'den çıkan ve ucunda büyükçe bir (6-uçlu DIN) fişi olan kabloyu, bilgisayarın arka tarafında MONİTÖR diye belirtilmiş sokete sokunuz.

3. MP2'den çıkan ve ucundan küçükçe bir (5 V DC) fişi olan kabloyu bilgisayarın arka tarafında 5 V DC diye belirtilmiş sokete sokunuz.

4. MP2'den çıkan ve ucunda anten fişi olan kabloyu TV'nizdeki AERIAL veya ANTEN soketine sokunuz.

5. Bilgisayarın arka tarafından çıkan ve ucunda küçükçe bir (12 V DC) fişi olan kabloyu MP2'nin arka tarafındaki sokete sokunuz.



CPC 6128 ve GT65/CTM 644 sisteminin çalıştırılması

Sistemi birbirine bağladıktan sonra fişi ceryana takın. Monitörün üstünde bulunan on-off düğmesine basın. Düğme içerideyken açık, dışarıdayken kapalı durumdadır.

Bilgisayarınızın arkasında bulunan açık-kapalı düğmesini ON durumuna getirin.

Bilgisayarınızın üzerindeki kırmızı lamba yanmalı ve aşağıdaki mesajı görmelisiniz.

```
Amstrad 128K Microcomputer  (v3)
©1985 Amstrad Consumer Electronics plc
         and Locomotive Software Ltd.

BASIC 1.1

Ready
```

Gözlerinizin yorulmasını önlemek için monütörünüzde bulunan BRIGHTNESS (parlaklık) düğmesi ile oynayarak iyi bir resim alın.
GT65 monütörü kullanıyorsanız kontrast ve düşey durdurma düğmeleri de ayarlanmalıdır.

Genelde kontrast düğmesi gözleri korumak amacı ile kısık seviyelere ayarlanmalıdır.

CPC 6128 ve MP2 Modülatör/Güç Kaynağı Sisteminin Çalıştırılması

Daha önce gösterilen şekilde bağlantıları yapılmış sistemin fişini şehir şebeke prizine sokunuz.

Bilgisayarın sağ arkasındaki sürgülü anahtarı kullanarak bilgisayarı çalıştırınız. Tuş takımının üst ortasındaki kırmızı ON lambası yanacak ve televizyonunuzu aşağıdaki görüntüyü verene kadar ayarlayacaksınız. (Şayet TV'niz işaretli ayarlama skalasına sahipse yaklaşık olarak 36'ncı kanalda çıkacaktır.)

```
Amstrad 128K Microcomputer  (v3)
©1985 Amstrad Consumer Electronics plc
        and Locomotive Software Ltd.
BASIC 1.1
Ready
```

TV setinizi ekranda temiz bir görüntü elde edinceye kadar ayarladıktan sonra yazılar koyu mavi fon üzerinde altın/sarı renkte olacaktır.
Eğer TV'niz döner anahtarlı program seçicisine sahipse yine yukarıda görülen en temiz görüntüyü alana kadar döndürünüz. (Burada da yine 36'ncı kanalda gözükecektir.)

Diğer Bağlantılar

Eğer siz aşağıda adları belirtilen diğer çevre elemanlarını bağlamak istiyorsanız detayları bu temel kursun ikinci kısmında bulabilirsiniz.

Oyunçubuğu(ları)
Kaset ünitesi
Yazıcı
İkinci disk sürücüsü
Harici amplifikatör/Hoparlör
Genişletici elemanlar

Sonuç olarak bu kitapçığın başlangıcında ÖNEMLİ başlığında verilen

MONTAJ BİLGİLERİ 1, 2, 4, 5, 6, 7, 8
ÇALIŞTIRMA BİLGİSİ 1

bilgilerini gözden geçiriniz.

Bölüm 2: Çevre Elemanlarının Bağlanması

Bu kısımda size değişik çevre elemanlarının CPC 6128 sistemine nasıl bağlanacağını izah edeceğiz. Bu cihazların nasıl kullanılacağı hakkındaki detaylı bilgiler bu kitapçığın uygun bölümlerinde bulunabilir.

Oyunçubuğu

AMSOFT oyunçubuğu JY2 modeli 6128 bilgisayarının oyunçubuğu kontrolu ve ateşi gerektiren oyun programları için alabileceğiniz ilave bir cihazdır.

JY2'den çıkan kablonun ucundaki fişi bilgisayarın üzerinde bulunan ve JOYSTICK diye belirtilen sokete takınız. 6128 iki adet oyunçubuğu ile kullanılabilir. Bu şekilde kullanılmak istendiği takdirde ikinci oyunçubuğu birinci oyunçubuğunun üzerinde bulunan sokete bağlanmalıdır.
AMSOFT oyunçubuğu bu bilgisayarla kullanılmaya müsaittir.

Oyunçubuğu ile ilgili diğer bilgiler daha sonra verilecektir.

Kaset Ünitesi

Programlar disk yerine kasetden yüklenebilir veya saklanabilir. Yükleme ve kayıt emirleri daha ileride açıklanacaktır.

6128 bilgisayarına kaset ünitenizi bağlamak için AMSOFT CL1 kablosu veya eş standartta kaset-arakablosu gerekir.

Kablonun uçundaki büyükçe (5 uçlu DIN) fişini bilgisayarın üzerindeki; TAPE yazan sokete sokunuz. Mavi kablonun ucundaki fişi, kaset ünitenizdeki REM veya REMOTE yazan sokete sokunuz.

Kırmızı kablonun ucundaki fişi kaset ünitesinin MIC, COMPUTER IN veya INPUT yazan soketine sokunuz.

Beyaz kablonun ucundaki fişi kaset ünitenizin EAR, COMPUTER OUT veya OUTPUT yazan soketine sokunuz. 6128 bilgisayarı ile kaset ünitesi arasında başarılı

veri transferi için kaset ünitesindeki LEVEL (seviye) veya VOLUME (ses) kontrollarının iyi ayarlanması gerekmektedir. Şayet yüklediğiniz veya kaydettiğiniz programlar doğru ve tam olarak kayıt edilmemiş veya yüklenmemişse, değişik seviye kontrol pozisyonlarını deneyerek en müsait pozisyonu bulabilirsiniz.

YAZICI

6128 herhangi bir Centronics eşdeğeri paralel yazıcı ile kullanılabilir. Eğer AMSTRAD DMP1 kullanmak isterseniz sadece yazıcının bağlantı kablosunu kullanmanız yeterlidir.

Şayet, herhangi bir Centronics eşdeğeri yazıcı kullanmak isterseniz o zaman AMSOFT PL1 yazıcı bağlantı kablosu gerekir.

Kablonun ucundaki yassı edge-konnektörü bilgisayarın arkasındaki PRINTER yazılı sokete sokunuz.

Kablonun diğer ucundaki Centronics sitilindeki fişi yazıcının arkasındaki sokete sokunuz. Eğer yazıcınızdaki sokette emniyet mandalları varsa bunları yazıcı fişinin kenarındaki ayaklara mandallamanız gerekmektedir.

Yazıcı operasyonlarının detayları bu kitapçığın ileriki konularındaki bulunmaktadır.

İKİNCİ BİR DİSK SÜRÜCÜSÜ (AMSTRAD FD1)

AMSTRAD FD1 ikinci bir disk sürücüsü olarak sisteme ilave edilebilir. İki sürücülü sistemin CP/M kullanıcısı için avantajı ise programın bir diskette, bilgilerin ise diğer bir diskette saklanmasıdır.

CP/M çalışması daima programın diskten yüklenmesini gerektirir. (Makinenin ROM'undaki BASIC'e giriş yoktur) RAM hafızasından daha geniş programlara izin veren çalışma tekniği (overlay tecnique) muhtelif dosyaları kullanmasına müsade eden CP/M den sonra, gerçek sistem diski verilere az bir yer bırakacak şekilde birçok program dosyası ihtiva edebilir.

6128 sistem diskinin ulaşılmaz becerileri sayesinde siz bütün gerekli olan dosya bakım, kopya, silme vb. işlemleri tek bir disk sürücüsü ile yapabilirsiniz. Bununla

beraber, ikinci bir sürücü elbette bu işlemlerin hızını arttıracak, kazaların faaliyet sahalarını azaltacaktır.

Büyükçe bir kenar konnektörü (edge-konnektör) olan kablonun ucunu bilgisayarın arkasında DISC DRIVE 2 işaretli sokete sokunuz.

Kablonun diğer ucundaki ufak fişi FD1 disk sürücüsünün arkasındaki sokete sokunuz.

UNUTMAYINIZ-İkinci disk sürücüsünü bağlarken veya sökerken her iki sürücüden de disklerin çıkarılmış olması ve sistemin kapalı olması gerekmektedir. Şayet sistem çalışır haydeyken bağlantıları değiştirmeyi, bilgisayarın hafızasındaki programın bozulmasına benzetebilirsiniz. Bağlantıları karıştırmadan önce değerli programlarımızı koruyalım.

FD 1 disk sürücüsü 6128'e bağlandığında önce FD 1'in sürgülü anahtarını, (disk sürücüsünün arka panelindedir) sonra da 6128'in anahtarını açınız. FD 1'in ön panelindeki kırmızı ve yeşil lambalar yandığı zaman sistemimiz çalışmaya hazır durumdadır.

2-sürücü operasyonunun detaylı bilgileri bu kitapçıkta daha sonraki bölümlerde verilecektir.

Harici Amplifikatör/Hoparlör

6128 stereo amplifikatöre ve hoparlöre bağlanabilir ve bilgisayarın 3 kanal kapasitesinden faydalanabilir.

Stereo amplifikatörün giriş (input) kablosunun ucu, bilgisayara uyması için 3,5 mm.lik jak fiş olmalıdır. Jak fişindeki bağlantılar şöyledir:

Fiş tepesi..... Sol kanal
Fiş orta halkası..... Sağ kanal
Fiş arka saftı..... Nötr

6128 cihazının STEREO soketinde sabit bir sinyal voltaj çıkışı vardır ve siz amplifikatörünüzün kendi ses, denge ve ton kontrolleriyle istediğiniz ayarları yapabilirsiniz.

Yüksek empedanslı kulaklıklar 6128 ile birlikte çalışabilir fakat ses bilgisayarın üzerindeki VOLUME kontrolu tarafından ayarlanamaz. Düşük empedanslı kulaklıklar, normal stereo sistemlerde kullanılanlar gibi direkt olarak bilgisayara girilerek çalışamazlar. Sesin istenilen çıkış kanalına yönlendirilmesi ile ilgili bilgiler ileride açıklanacaktır.

Genişleme Araçları

Seri interface, modem, ışıklı kalem, ROM vb. genişleme araçları 6128'in arkasındaki EXPANSION yazılı sokete bağlanabilirler.

AMSOFT konuşma sentezcisi/amplifikatörü olan SSA2 modeli (speech synthesiser/amplifier) de bu sokete bağlanabilir.

EXPANSION soketine bağlantılar hakkındaki detaylar "Sizin Bilginiz İçin" adı altındaki bölümde verilecektir.

Sonuç olarak, bu kitapçığın başlangıcında ÖNEMLİ başlığında verilen:

MONTAJ BİLGİLERİ 6,7
ÇALIŞTIRMA BİLGİLERİ 4,8

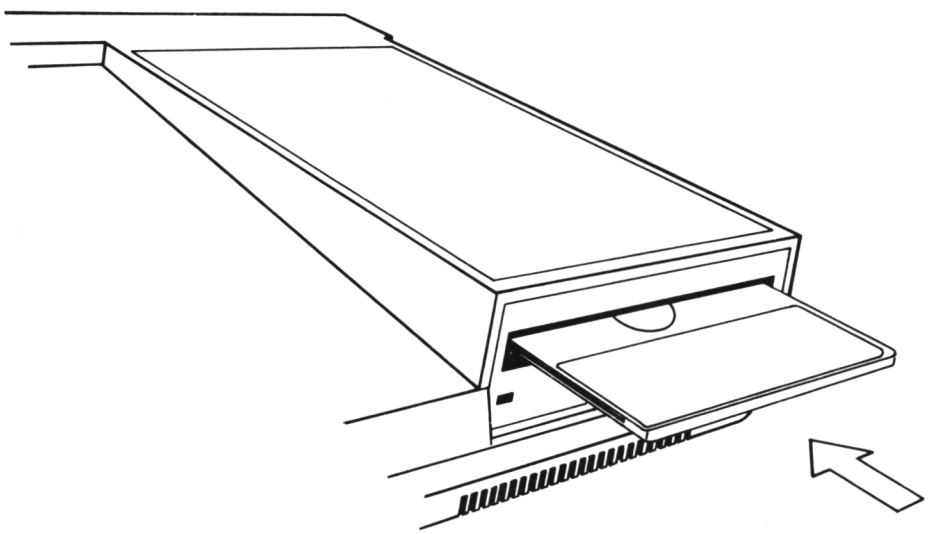
Bilgilerini gözden geçirip geçirmediğinize dikkat ediniz.

Bölüm 3: Disk Hakkında

AMSTRAD CPC 6128'de 3 inçlik compact floppy disk kullanılır. Biz, diske iyi ve doğru bir veri transferi için sadece AMSOFT CF2 compact floppy diskini kullanmanızı tavsiye ederiz.

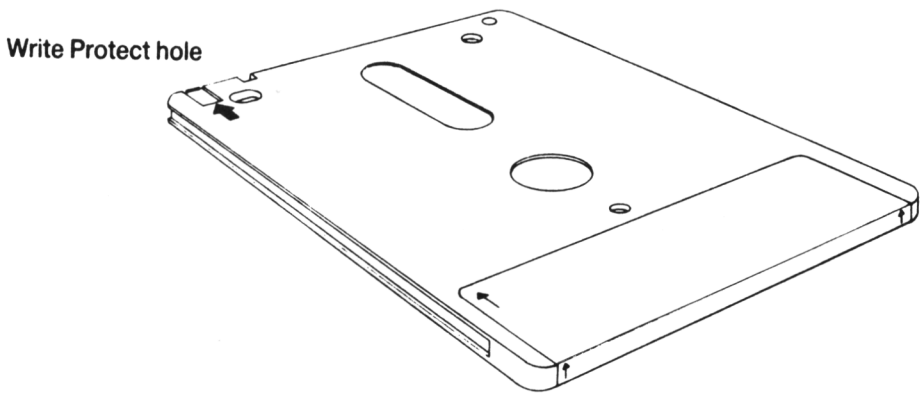
Diski Yerleştirme

Diskin her iki tarafıda ayrı ayrı kullanılabilir. Disk yerleştirilirken yazılı (antetli) tarafı sürücünün dış tarafına gelecek şekilde ve kullanmayı düşündüğünüz tarafı yukarıya gelecek şekilde konulması gerekir.



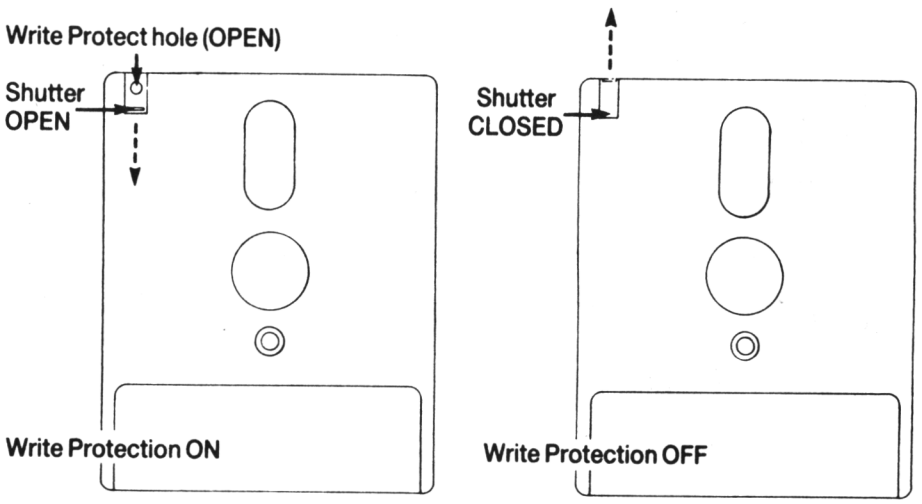
Yazmaya Karşı Koruma

Boş diskin her iki tarafında, sol köşede bir okun işaret ettiği kapatılan delik göreceksiniz. Bu yazıma karşı koruma deliğidir. Bu delik, silmeye ve tekrar kaydetmeye karşı koruma görevi yapar.



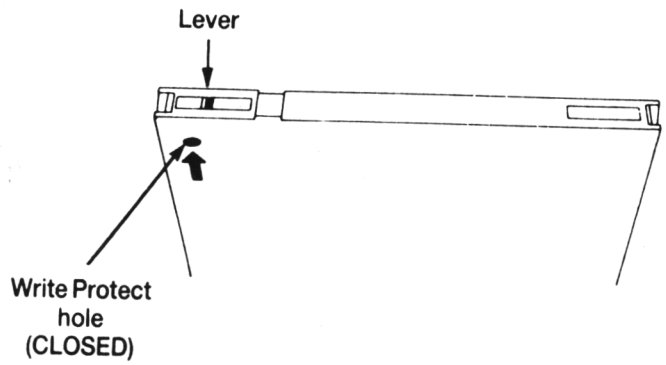
Bu delik kapalı iken, veriler bilgisayar tarafından diske yazılabilir. Fakat delik açık iken disk herhangi bir verinin üzerine yazılmasını önler, böylelikle değerli programlarınızın kazara silinmesi önlenmiş olur. Yazı koruma deliğinin açılması ve kapanması aşağıda belirtildiği gibidir.

Yazı koruma deliğinin açılması için, diskin sol köşesindeki küçük düğmeyi (sürgüyü) kaydırmanız gerekir.

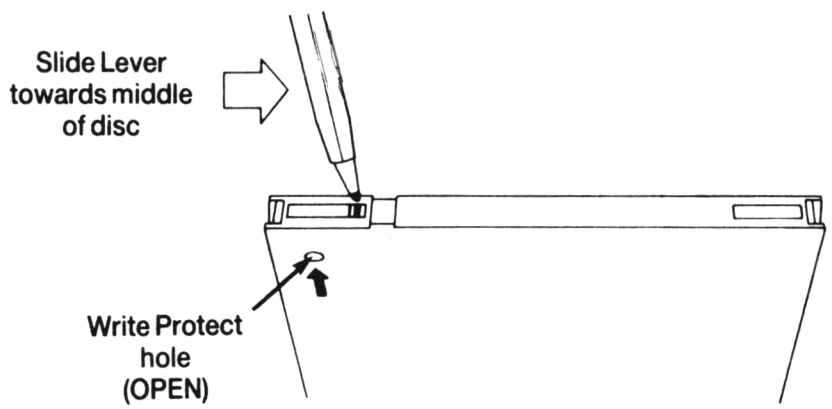


Yazı koruma deliğini kapamak için sürgüyü eski pozisyonuna getirmek yeterlidir.

Bazı compact floppy disk üreticileri ise, diskin sol köşesinde bir yuva içinde küçük bir plastik ayırıcı koymuşlardır.



Bu tür disklerin yazı koruyucu deliklerini açmak için ayırıcıyı diskin orta tarafına doğru kaydırmak gerekir. Bu işlem için tükenmez kalem ucu veya benzeri birşey kullanabilirsiniz.



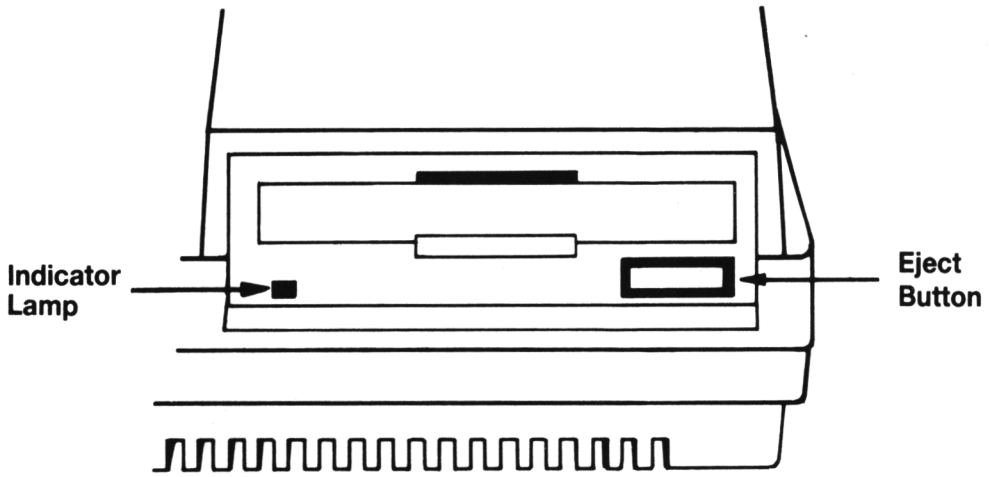
Hangi metod ile olursa olsun, yazıma karşı koruma deliğinin açılması her şartta disk üzerine yazmayı önler.

Önemli

Daima ana CP/M sistem diskinin yazıma karşı koruma deliğinin açık olduğundan emin olunuz.

Diskiniz İçerideyken

Bilgisayarın disk sürücüsünün önünde kırmızı indikatör lambası ve EJECT butonu vardır.



İndikatör Lambası

Bu lamba verilerin diske yazıldığını veya diskden okunduğunu belirtir.

Şayet ikinci disk sürücüsü bağlanmışsa, ikinci disk sürücüsünün kırmızı indikatör lambası sürekli yanacaktır. Bilgisayarla birlikte olan esas disk sürücüsü diske yazarken veya diskten okurken ikinci disk sürücüsünün indikatör lambası söner.

Dışarı Atma (Eject) Butonu

Bu butona basınca diskimizi disk sürücüden dışarıya alabiliriz.

Sonuç olarak, bu kitapçığın başlangıcında ÖNEMLİ başlığında verilen

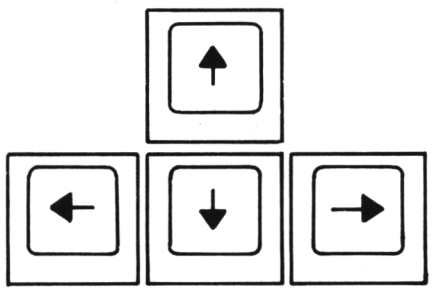
ÇALIŞTIRMA BİLGİLERİ 1, 3, 4, 5, 6

Bilgilerini gözden geçirip geçirmediğinize dikkat ediniz.

Bölüm 4: Başlarken

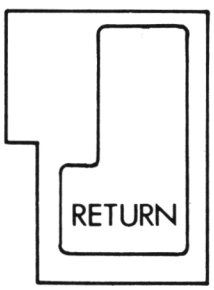
Yazılım (software) yüklemeye ve programları diske kaydetmeye başlamadan önce bilgisayardaki bazı tuşları tanımak faydalı olacaktır.

Bilgisayarı açıp, açılış mesajını ekranda gördükten sonra çeşitli tuşların neler yaptığını görelim.



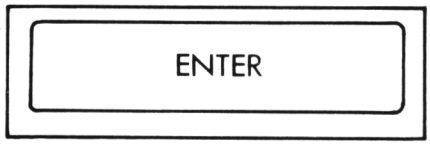
Kursör tuşları (klavyenin sağ alt köşesinde) kursörün (ekrandaki ışıklı kare) ekrandaki yerini değiştirir.

Kursör tuşlarına sırayla basınız ve kursörün ekrandaki hareketini izleyiniz.



[RETURN] tuşu bilgisayara yazdığınız bilgileri, bilgisayara girer. (işleme konulmasını sağlar) [RETURN] tuşuna basıldıktan sonra ekranda yeni bir satır başlar. Bilgisayara girmesini (işleme konulmasını) istediğiniz her bilginin sonunda [RETURN] tuşuna basmalısınız.

Bundan sonra [RETURN] gördüğümüz zaman, her program satırından veya komutundan sonra [RETURN] tuşuna basmamız gerektiğini anlayacağız.

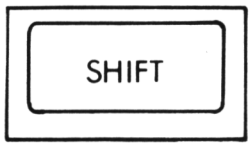


Normal şartlar altında bu tuş [RETURN] tuşuyla aynı işlemi yapar ve kullanılabilir. Bununla beraber nümerik tuş takımındaki fonksiyon tuşları gibi (ENTER) tuşu diğer kullanımlar için yeniden tanımlanabilir. Bu işlem daha sonraki bölümlerde izah edilecektir.



Bu tuş ekranda kursörün solundaki gerekli olmayan karekteri (harf veya sayı) silmek için kullanılır.

Sırasıyla a b c d yazınız, d harfinin kursörün solunda kaldığını göreceksiniz. d harfini istemediğinize karar verdiğinizde [DEL] tuşuna bir bir kez basın, d harfinin kaybolduğunu göreceksiniz. Şayet [DEL] tuşuna basar ve parmağınızı çekmeden basmayı sürdürürseniz c, b, a harfleri sırasıyla kaybolacaktır.



Bilgisayarda iki tane [SHIFT] tuşu vardır. Bunlardan birine basar ve parmağınızı kaldırmadan diğer bir tuşa basarsanız, tuş bir harfi gösteriyorsa o harfin büyüğü; iki karakterli tuş ise üstteki karakter ekranda yazılacaktır.

e tuşuna basınız, sonra [SHIFT] tuşuna basıp tekrar e tuşuna basınız. Ekranda aşağıdaki mesaj görülür:

eE

Şimdi ara çubuğuna basın. Tuş takımının üzerindeki 2 rakamının bulunduğu tuşa önce yalnız, sonra [SHIFT] ile basın. Ekranda aşağıdaki mesaj görülür:

2''

Artık [SHIFT] tuşu ve bir tuşa basılınca ne olduğunu biliyorsunuz. Alıştırma için önce yalnız, sonra [SHIFT] ile tuşlara basarak denemeler yapınız.



Bu [SHIFT]'e benzeyen bir işlemdir, sadece bu tuşa bir defa basmak yeterlidir. Bu tuşa bir defa basıldıktan sonra yazacağınız her harf büyük harf olarak gözükecektir, ancak rakamlar değişmeyecektir.

[CAPS LOCK] tuşuna basınız ve abcdef123456 yazınız;

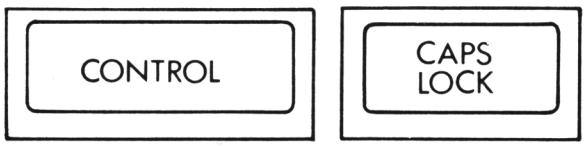
Ekranda aşağıdaki mesaj görülecektir:

ABCDEF123456

Dikkat ederseniz harfler büyük harf, rakamlar ise aynen yazıldı. Şimdi isterseniz [CAPS LOCK] çalışırken [SHIFT] tuşuna basıp aynı işlemi tekrarlarsak ekranda aşağıdaki mesaj görülür;

ABCDEF !" #$%&

Eğer tekrar küçük harflere dönmek isterseniz [CAPS LOCK] tuşuna bir defa daha basınız.



Eğer bütün büyük harfleri ve tuşların üst kısımlarındaki sembolleri sürekli [SHIFT] tuşuna basmaksızın kullanmak isterseniz [CONTROL] tuşuna basarken [CAPS-LOCK] tuşuna bir kez basınız. Bu işlem "SHIFT LOCK" fonksiyonunu meydana getirir. Şimdi yukarıdaki işlemi tekrar yapalım; abcdef123456 yazınız;

Ekranda aşağıdaki mesaj görülecektir:

ABCDEF !" #$%&

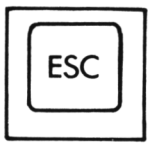
Şunuda hatırlatmak isteriz ki[CONTROL] ve [CAPS LOCK]beraberce çalışır vaziyette iken, rakamları yazmak mümkündür. Bunu yapabilmek için, tuş takımının sağındaki (F0'dan F9'a kadar) rakam tuşlarına basmak yeterlidir.

[CONTROL] tuşuna basarken [CAPS LOCK]'a birkez basarsak tuşların alt kısımlarını kullanabiliriz. Küçük harf için tekrar [CAPS LOCK] tuşuna basınız.



Bu tuş kursörün içindeki harf veya rakamı silmek için kullanılır. ABCDEFGH yazınız. Kursör son harfin (H) sağında yer alacaktır. Şimdi sol kursör tuşuna dört defa basın. Kursör dört hane sola gidecek ve E harfinin üzerinde duracaktır.

E harfinin kursörün içinde göründüğüne dikkat ediniz. [CLR] tuşuna bir kez basınız, E harfi silinecektir ve FGH harfleri bir hane sola kayacaktır, F harfi kursörün içindedir. Şimdi [CLR] tuşuna tekrar basarsanız, F harfi silinip yerine G harfi gelir. Bu işlemi tekrarlarsak G ve H harfleri de sırayla silinecektir.



Bu tuş bilgisayarın yapmakta olduğu işlemi durdurması için kullanılır. Bu tuşa bir kez basıldığında bilgisayar geçici olarak bekleme (pause) durumuna girer ve her hangibir tuşa basmakla tekrar eski işlemine devam eder.

[ESC] tuşuna iki kez basarsanız bilgisayar yapmakta olduğu işlemi tamamen durdurur ve sizden talimatlar almaya hazır duruma gelir.

Önemli

Ekranda bir satıra 40 karakter yazıldığı zaman ekranın sağ kenarına gelinmiş olur. Bundan sonra yazılacak karakter otomatik olarak bir sonraki satırın baş tarafına gider. Karakteri satır başına almak için [RETURN] tuşuna basmamalısınız. Bilgisayar bu işi otomatik olarak yapmaktadır.

Şayet [RETURN] tuşuna basarsak bilgisayar hemen veya program çalışırken hata mesajı (Syntax error) verir.

Yazılım Hatası (Syntax Error)

Ekranda Syntax error mesajı görünürse, bilgisayar sizin verdiğiniz talimatı anlamıyor demektir.

Örneğin aşağıdaki talimatı yazınız.

printt [RETURN]

Eranda aşağıdaki mesaj görülür:
Syntax error

Bu mesaj, bilgisayarın iki t'li print sözcüğünün anlamını çıkaramadığı için ekrana gelmiştir.

Eğer bir program satırında hata yaparsanız, örneğin;

10 printt "abc" [RETURN]

Syntax error mesajı bilgisayar programı çalıştırırken verilen konutu işleme sokuncaya kadar ekranda gözükmez. Şimdi;

run [RETURN] yazalım. Ekranda görülen mesaj şöyledir:

Syntax error in 1Ø
10 **P** rintt "abc"

Bu mesaj size hatanın hangi satırda olduğunu belirtir ve program satırı ile birlikte hatayı düzeltmek için kursörü de gösterir.

Şimdi sağ kursör tuşuna kursör printt'teki t harfine gelinceye kadar basınız. Sonra da [CLR] tuşuna basarak fazla t'yi silin, [RETURN] tuşuna basarak düzeltilmiş satırı bilgisayara girin ve run [RETURN] yazın. Bilgisayar komutu kabul ederek ekrana;

abc

yazacaktır.

Son olarak bu kitapçığın başlangıcından ÖNEMLİ başlığında verilen bilgileri tekrar gözden geçiriniz.

Montaj Bilgileri 4,5
Çalıştırma Bilgileri 1

Bölüm 5: Yazılım ve Oyunları Yükleme

Yazılım (software) diskinin yükleme hızının demostrasyonu için sistemi açalım ve CP/M diskimizin 4'ncü yüzü yukarı gelecek şekilde disk sürücüsüne yerleştirelim. Ekrana aşağıdaki emiri yazalım;

run "rointime. dem" [RETURN]

Birkaç saniye sonra program hafızaya yüklenecektir. Kullandığınız monitörün yeşil olup olmadığına göre ekrandaki soruya cevap verin. (Monitör yeşilse Y, renkli ise N) Ekranda "Roland In Time" oyununun sürekli demonstrasyonu görülecektir. Belki de dışarıya çıkıp oyunun bir kopyasını alabilirsiniz.

Demonstrasyonu izlemeyi bitirdiğiniz zaman [CONTROL] ve [SHIFT] tuşlarına sırayla basınız, bu tuşları basılı tutarken [ESC] tuşuna basınca programdan kurtulabilirsiniz. Bu işlemi yapmakla bilgisayarı resetlemiş olursunuz ve isteğinize göre yeniden başlayabilirsiniz. (Bilgisayarı bu şekilde reset yaptıktan sonra diski, disk sürücüsünden çıkarmaya gerek yoktur.)

Şayet program yüklenmemişse nerede yanlış yaptığımızı ekranda görebiliriz.

Drive A: disc missing
Retry, Ignore or Cancel?

Bunun anlamı, diskinizi ya doğru olarak A sürücüsüne sokmadınız yada sizin iki sürücünüz olup diski B sürücüsüne soktunuz.

ROINTIME.DEM not found

Bunun anlamı, ya yanlış disk soktunuz (veya diskin yanlış yüzünü) yada ROINTIME.DEM adını tam ve doğru olarak yazmadınız.

Bad command

Bunun anlamı ROINTIME.DEM ismini, istenmeyen bir boşluk vererek veya noktalamada hata yaparak eksik yazdınız.

Type mismatch

Anlamı, tırnak işaretini ('') unutmuşsunuz

Syntax error

Anlamı, run kelimesini yanlış kullandınız

Drive A: read fail
Retry, Ignore or Cancel?

Anlamı, bilgisayar diskinizdeki verileri okumayı beceremediği demektir. Doğru diski koyup koymadığınızı kontrol edin ve tekrar denemesi için R tuşuna basın. Bu mesaj, diskiniz sürücüde iken sistemi açıp kaparken diskin bozulması yüzünden verilir.

Öncelikle bir diskin nasıl kopya edilebileceğini öğrenmeniz gerekir. Böylelikle değerli programlarınızın ve özellikle CP/M sistem diskinizin kopyalarını çıkarıp saklayabilirsiniz.

Amsoft Yazılımının ve Hoşgeldin Programının Yüklenmesi

Yazılım diskinizi sürücüye sokunuz ve şunu yazınız:

run "disc" [RETURN]

Birkaç saniye sonra oyun yüklenip oynanmaya hazır hale gelecektir.

CP/M sistem disk paketinden 4'ncü yüzü sürücüye sokup şunu yazınız:

run "disc"

Birkaç saniye sonra devamlı "Welcome" demonstrasyonu yüklenecektir.

Sonuç olarak bu kitapçığın başında ÖNEMLİ başlığı altında verilen bilgileri gözden geçiriniz.

MONTAJ BİLGİLERİ 6
ÇALIŞTIRMA BİLGİLERİ 1, 5, 6

Bölüm 6: Hesap Edelim

Şimdilik bilgisayar ile ne yapıp ne yapmayacağımızı, bilgisayarı nasıl kuracağımızı ve çevre elemanlarını bağlamayı öğrendik. Şimdi ise bazı komutların ne işe yaradığını öğreneceğiz.

Bilgisayar, herkes gibi sadece kendi bildiği lisandaki komutları anlayabilir ve bu lisan BASIC'dir. (Beginners'All-purpose Symbolic Instruction Code) Basic'in sözlüğünde kelimelere şifre anahtarı (keyword) denir ve her biri bilgisayarın belli bir işlevi yapmasını sağlar. Bütün lisanların gramer kuralları gibi BASIC'inde kuralları vardır. Burada gramer söz dizimidir (Syntax) ve bilgisayar söz dizimi hatası (Syntax error) yaptığınızı size söyler.

AMSTRAD BASIC Komutlarının Tanıtımı

"AMSTRAD CPC 6128 BASIC Komutlarının Komple Listesi" başlıklı kısımda, AMSTRAD BASIC'de bulunan bütün komutların tanımları verilecektir. Bu bölümde biz çok kullanılan BASIC komutlarını tanıtacağız.

CLS

Ekran temizleme komutudur. Şimdi şu mesajı yazınız:

cls [RETURN]

Büyük yada küçük harf kullanabilirsiniz. Bu komuttan sonra ekranın temizlendiğini ve sol üst köşede kursör ile Ready (hazır) sözcüğünün belirdiğini göreceksiniz.

PRINT

Bu; program içinde karakter, kelime ve sayıların ekrana yazılmasını istediğiniz zaman kullanılır. Bilgisayara aşağıdaki komut satırını yazınız.

print "merhaba" [RETURN]

Ekranda aşağıdaki mesaj görülür:

merhaba

Tırnak işaretleri "" bilgisayara neyin yazılması gerektiğini belirtmek için kullanılır. Merhaba sözcüğü, ekrandan [RETURN] tuşuna basar basmaz gözükür. Şimdi;

cls [RETURN]

yazın ve ekranı temizleyin.

RUN

Bir önceki örnekte tek satırlık bir komut gösterilmişti [RETURN] tuşuna basılınca bu komut işleme sokulmuş ve işlem bitince unutulmuştu. Bilgisayarda, seri komutlar belirlenmiş bir sırada saklanabilir. Buda program yazmakla mümkündür. Programa yazdığınız BASIC komutlar, daha önce gördüğünüz komutlara benzer fakat her komut satırının önüne bir satır numarası vermek gerekir. Eğer programda birden fazla satır varsa, satır numaraları bilgisayara çalışma veya işlem sırasını belirtir. [RETURN] tuşuna basıldığında satır program çalışıncaya kadar belleğe alınır. Şimdi aşağıdaki satırı yazın:

10 print "merhaba" [RETURN]

[RETURN] tuşuna basınca merhaba ekranda yazılmadı fakat bilgisayarın belleğine, bir satırlık program olarak alındı. Bu programın işleme konulması için run sözcüğünün kullanılması gerekir.

run [RETURN]

Şimdi ekranda merhaba göreceksiniz. Şunu unutmayın ki sürekli print yazacak yerde ? (soru işareti) de kullanabilirsiniz. örneğin;

10 ? "merhaba" [RETURN]

LIST

Program belleğe alındıktan sonra ne yazıldığını kontrol etmek için kullanılır. Bilgisayara şu mesajı girersiniz:

list [RETURN]

10 PRINT "merhaba" ekranda görülür.

PRINT'i BASIC komutu olarak kabul ettiğini gösterir.

Şimdi CLS [RETURN] yazarak ekranı temizleyin. Ekran CLS [RETURN] komutu ile temizlendiği halde programınız bilgisayarın belleğinden silinmemiştir.

GOTO

GOTO komutu bilgisayara bir satırdan bir başka satıra bazı satırları atlamak için veya bir çevrim oluşturmak için gitmesini sağlar. Denemek için şunu yazınız.

10 PRINT "merhaba" [RETURN]
20 GOTO 10 [RETURN]

run [RETURN]

Ekranda merhaba sözcüğünün solda altalta devamlı yazıldığını göreceksiniz. Bunun sebebi şudur. Programdaki 20. satır bilgisayara 10. satıra geçmesini söylüyor ve programın işlemesini burada başlatıyor.

Bu programın işlemesini geçici olarak durdurmak için [ESC] tuşuna bir kere basınız. Tekrar başlatmak için her hangi bir tuşa basınız. Programı başka komutlar ilave etmek için durdurmak isterseniz [ESC] tuşuna iki defa basınız.

cls [RETURN]

Tuşuna basarak ekranı temizleyiniz.

Merhaba sözcüğünü sürekli yan yana yazarak ekranı doldurmasını isterseniz bir önceki programı yeniden yazınız. Fakat tırnak işaretinden sonra noktalı virgül (;) koyunuz.

10 PRINT "merhaba"; [RETURN]

10 GOTO 10 [RETURN]

run [RETURN]

Noktalı virgül bilgisayara daha sonra yazacağı karakterler gurubunu bir öncekinin hemen yanına yazmasını belirtmiş oluyor.

[ESC] tuşuna iki defa basınız ve programı durdurunuz. Şimdi 10 uncu satırı tekrar yazınız. Fakat bu defa noktalı virgül yerine virgül (,) kullanın.

10 PRINT "merhaba", [RETURN]

run [RETURN]

Görüyorsunuz ki virgül bilgisayara sonraki karakter gurubunu bir önceki karakter gurubundan 13 kolon sonraya koymasını belirtiyor. Bu özellik bilgileri ayrı sütunlarda yazabilmek açısından faydalıdır. Ancak dikkat edilecek husus şayet ilk guruptaki karakter sayısı 12'yi geçerse sonraki karakter gurubu bir 13 kolon daha araverilerek yazılır. Böylelikle kolonlar arasında daima bir ara mevcut olur. Bu 13 kolonluk şekil ZONE komutu kullanarak ayarlanabilir. Bu ileriki konularda anlatılacaktır.

INPUT

Bu komut bilgisayara bir şeyin yazılmasını beklemesi gerektiğini belirtir. Örneğin bir sorunun cevabı olabilir.

10 INPUT "kaç yaşındasınız"; yaş [RETURN]
20 PRINT "siz"; yaş; "yaşında göstermiyorsunuz" [RETURN]

run [RETURN]

Ekranda şunu göreceksiniz.

Kaç yaşındasınız?

Şimdi yaşınızı yazın ve [RETURN] e basın. Eğer yaşınız 18 ise ekranda şunu göreceksiniz.

Siz 18 yaşında göstermiyorsunuz.

Bu örnek INPUT komutunun ve sayısal değişkeninin kullanımını göstermektedir. Burada yaş sözcüğü değişken olarak kullanılmıştır.

[CONTROL] [SHIFT] [ESC]tuşlarını kullanarak bilgisayarın belleğini temizleyin. Eğer girişin harfler veya harf ve rakam gurubundan oluşmasını isterseniz değişkenin sonuna ($) işareti koymanız gerekir. Bu tip değişkenlere 'dizi değişkeni' denir.

Şu programı yazınız. (20 nci satırda merhaba'nın a sından sonra ve benim'in b sinden önce birer boşluk veriniz)

```
10 INPUT "İsminiz nedir"; isim$ [RETURN]
20 PRINT "merhaba  "; isim$; "  benim adım Ronald" [RETURN]
run  [RETURN]
```

Ekranda şunu göreceksiniz.

```
İsminiz nedir?
```

İsminizi yazın ve [RETURN] tuşuna basın. Eğer isminiz Tezer ise ekranda şunu göreceksiniz.

```
Merhaba Tezer benim adım Ronald
```

Örnekte dizi değişkeni olarak isim$ kullanmıştık. Biz bunun yerine tek bir harfde kullanabiliriz. Örneğin a$. Şimdi yukarıdaki 2 örneği bir programda birleştirelim.

```
5 CLS [RETURN]
10 INPUT "İsminiz nedir"; a$ [RETURN]
20 INPUT "Yaşınız kaç";b [RETURN]
30 PRINT "doğru söylüyorum "; a$ " hiç de"; b; " göstermiyorsun" [RETURN]
run [RETURN]
```

Bu programda ad ve yaş olarak sırasıyla a$ ve b değişkenleri kullandık. Ekranda şunu göreceksiniz.

```
İsminiz nedir?
```

Şimdi adınızı yazınız (örneğin Tamer) ve [RETURN]'e basın; size şu soru sorulacaktır.

```
Yaşınız kaç?
```

Yaşınızı yazınız (örneğin 27) ve [RETURN]'e basınız.
Ekranda şunu göreceksiniz.

```
Doğru söylüyorum Tamer hiç de 27 göstermiyorsun.
```

Program Düzeltme

Bir programda satırlardan herhangi biri yanlış yazılmışsa ve syntax error veya bir

başka hata mesajı alınmışsa bu satırı tekrar yazmadan düzeltebiliriz. Bunu göstermek için önceki programı yanlış yazalım.

```
5 CLSS [RETURN]
1Ø INPUT "isminiz nedir"; a$ [RETURN]
2Ø INPUT "yaşınız kaç"; b [RETURN]
3Ø PRINT "doğru söylüyorum"; a$"iç de"; b;" yaşında göstermiyorsunuz"
[RETURN]
```

Yukarıdaki programda 3 hata yapılmıştır:
5 nci satırda cls yerine clss yazılmış

1Ø ncu satırda isminiz yerine ismini yazılmış
3Ø ncu satırda söylüyorum kelimesi ile tırnak işareti arasında 1 boşluk bırakılmamış.

Programı düzeltmek için 3 temel metod vardır. Birincisi tekrar yeni bir satır yazmak. Yeni satır yazarken yanlış yazılan satır ile aynı satır numarası taşıması gerekir, aksi takdirde bellekte yeni bir program satırı olarak saklanır.

İkincisi edit metodu (kürsör düzeltme).

Son olarak kürsör kopya metodu

Edit Metodu (kursör düzeltme)

5 nci satırdaki hatayı düzeltmek için:

edit 5 [RETURN] yazınız

5 nci satır kursör clss sözcüğündeki c harfinin üzerinde olarak 3Ø ncu satırın altında gözükür.

clss sözcüğündeki fazla s harfini yok etmek için kursör sağ tuşuna, kursör en sondaki fazla s harfinin üzerine gelinceye kadar basınız. Sonra [CLR] tuşuna basın. s harfinin silindiğini göreceksiniz.

Şimdi [RETURN] 'e basın ve böylelikle 5'nci satır bellekte düzeltilmiş olur. Bilgisayara aşağıdaki komutu girin:

list [RETURN]

5'nci satırın düzeltilmiş olduğunu görün.
Kursör düzeltme metodunu kullanma amacı ile aynı amaca sahip olan AUTO komutu, ileride açıklanacaktır.

Kopya Kürsörü Metodu

Kopya kürsörü (ekranda görülene ek olarak), [SHIFT] basılı iken kürsör tuşlarından birine basınca meydana çıkan bir kürsördür. Bu yolla ana kürsörden ayrılır ve ekranda bağımsız olarak hareket ettirilebilir.

10 ve 30 uncu satırlardaki yanlışları düzeltmek için, [SHIFT] tuşunu basılı tutun ve kopya kürsörü 10 numaralı satırın başına gelinceye kadar kürsör yukarı tuşuna basın. Ana kürsörün hareket etmediğini ve şimdi ekranda iki kürsörün olduğunu göreceksiniz. Şimdi kopya kürsörü 'you' ve 'name' arasındaki boşluğa gelene dek [COPY]'ye basın. 10 numaralı satırın en alt satıra yeniden yazıldığını ve ana kürsörün, kopya kürsörü ile aynı yerde durduğunu göreceksiniz. Şimdi 'r' harfini yazın. Bu sadece alttaki satırda görülecek.

Ana kürsör hareket etti fakat kopya kürsörü olduğu yerde kaldı. Şimdi 10. satırın tamamı kopya edilene kadar [COPY]'ye basın. [RETURN]'e basın ve bu yeni 10. satır hafızaya alınsın. Kopya kürsörü kaybolur ve ana kürsör yeni 10. satırın altında belirir. İkinci yanlışı düzeltmek için, [SHIFT]'i basılı tutun ve kürsör yukarı tuşuna basarak kopya kürsörünü 30. satırın bazına getirin.

[COPY] tuşuna, kopya kürsörü 'say' dan sonraki tırnak işaretinin üzerine gelinceye kadar basın. Şimdi bir kere boşluk tuşuna basın. Alttaki satıra bir boşluk yazılacak. 30. satırın tamamı kopyalanana kadar [COPY]'yi basılı tutun, sonra [RETURN]'a basın.

Şimdi hafızada programın düzeltilip düzeltilmediğini kontrol etmek için liste alabilirsiniz.

```
List [RETURN]
```

yazın.

NOT: (Düzeltme esnasında) kürsörü hızlı hareket ettirmek için [CONTROL] tuşuna basılı tutun ve sol veya sağ kürsör tuşuna bir kez basın.

Şimdi [CONTROL] [SHIFT] ve [ESC] tuşlarına basarak bilgisayarı sıfırlayın.

IF

IF ve THEN komutları bilgisayara, verilen bir koşulu denemesini ve sonuca göre hareket etmesini bildirir. Örneğin şu komutta:

Örnek:

```
if 1 + 1 = 2 then print "doğru" [RETURN]
```

bilgisayar belirlenen şartı kontrol eder ve buna göre işleme geçer.

ELSE komutu da IF THEN komutu ile kullanılarak, kontrol edilen şart yanlış ise bilgisayarın yapacağı işlemi belirler. Örneğin;

```
if 1 + 1 = 0 then print "doğru" else print "yanlış" [RETURN]
```

Şimdi bir önceki programı if ve then kullanarak yazalım.

Aşağıdakileri yazınız. Dikkat ederseniz iki sembol daha ilave ettik. < daha küçük demektir (M tuşunun yanında), > daha büyük demektir (? tuşunun yanında)

```
5 cls [RETURN]
10 input "isminiz nedir"; a$ [RETURN]
20 input "yaşınız kaç"; yaş [RETURN]
30 if yaş < 13 then 60 [RETURN]
40 if yaş < 20 then 70 [RETURN]
50 if yaş > 20 then 80 [RETURN]
60 print a$; "sen"; yaş; "yaşında olduğuna göre henüz küçük sayılırsın":end
[RETURN]
70 print a$; "sen"; yaş; "yaşındasın oldukça büyümüşsün":end [RETURN]
80 print a$; "sen"; yaş; "yaşında olduğuna göre artık çocuk değilsin" [RETURN]
```

Programın doğru olup olmadığını kontrol etmek için,

```
list [RETURN]
```

yazınız.

```
run [RETURN]
```

yazarsanız, bilgisayar size soruları soracaktır. Sorduğu soruları cevaplayın sonucu izleyiniz.

IF ve THEN komutlarının program içindeki etkilerini gördünüz. Aynı zamanda 60 ve 70 'nci satırların sonuna bir de END sözcüğü ekledik. Eğer 60'ncı satırda bu komutu kullanmasaydık program çalışmaya devam eder, 70 ve 80'nci satırları uygulamaya geçerdi.

Aynı şekilde, şayet 70 'nci satırın sonunda END olmasaydı program devam eder ve 80'nci satırı işleme sokardı. İki nokta üstüste (:) bir önceki komutu bir sonrakinden ayırmak için kullanılır. Böylece bir program satırına

birden fazla komut yazılabilir. 5'nci satırı program çalışmadan önce ekranı temizlemek için kullandık.

Bilgisayarı [CONTROL] [SHIFT] ve [ESC] tuşlarına basarak resetleyiniz.

For ve Next

FOR ve NEXT komutları bir operasyonun muhtelif defalar çalışması için kullanılır. Bu operasyondaki komutlar FOR ve NEXT çevrimiyle sınırlanmıştır. Aşağıdaki programı bilgisayara yazınız.

```
5 cls [RETURN]
10 for a=1 to 10 [RETURN]
20 print "operasyon numarası";a [RETURN]
30 next a [RETURN]
run [RETURN]
```

20 satırdaki işlem, 10'ncu satırdaki FOR komutunun belirttiği gibi 10 defa yapılacaktır. Değişken a değeri her seferinde 1 artacaktır.

STEP sözcüğü FOR NEXT komutunda değişkenin her işlemde ne kadar artacağını belirtmek için kullanılır. Örneğin 10'ncu satırda şu değişikliği yapalım:

```
10 for a= 100 to 0 step-10 [RETURN]
run [RETURN]
```

REM

REM bir nevi hatırlatmadır. Komut, bilgisayara kendisini takip eden satırdaki her şeyi yok saymasını, işleme almamasını ikaz eder. Siz bu komutu, bir programın adını, değişkenlerin ne için kullanıldığını ve bunun gibi program takibinde size kolaylık sağlayacak her türlü hatırlatmaları eklemek için kullanabilirsiniz.

```
10 REM programın ilk bölümü [RETURN]
20 L=5: REM yaşayanların sayısı [RETURN]
```

Tekli tırnak işareti (') [SHIFT] tuşuna basarken 7 tuşuna basınca yazılır) REM yerine kullanılabilir.

```
10 ' programın ilk bölümü [RETURN]
20 L=5: ' yaşayanların sayısı [RETURN]
```

GOSUB

Program içinde bir dizi komut muhtelif yerlerde çok sık kullanılıyorsa bu komutları bir altprogram (subroutine) şeklinde yazabilir vede GOSUB komutu ile çağı-

rabilirsiniz. GOSUB komutunu satır numarası izler. GOSUB-işleminin sonu RETURN komutu ile sonuçlanır. Bu anda bilgisayar daha önce itaat ettiği GOSUB komutunu takip eden satıra geri döner.

(Bundan sonraki iki program ekranda kelimeleri yazmaktan başka bir şey yapmaz. Burada sırf programın tekrarlı işleri yapması gösterilmiştir.)

```
10 MODE 2 [RETURN]
20 PRINT "Ali babanın bir çiftliği var" [RETURN]
30 PRINT "Çiftliğinde kedileri var" [RETURN]
40 PRINT "Miyav miyav diye bağırır" [RETURN]
50 PRINT "Çiftliğinde Ali babanın" [RETURN]
60 PRINT "Ali babanın bir çiftliği var" [RETURN]
70 PRINT "Çiftliğinde köpekleri var" [RETURN]
80 PRINT "Hav hav diye bağırır" [RETURN]
90 PRINT "Çiftliğinde Ali babanın" [RETURN]
run [RETURN]
```

Dikkat ederseniz programda bazı satırları tekrar etmek zorunda kaldık. Bu tekrar edilen satırları altprograma koyup sonunada RETURN komutunu ilave edeceğiz. Altprogramlarıda GOSUB 150 ve GOSUB 170 ile çağıracağız. Şimdi programı şöyle yazalım:

```
10 MODE 2 [RETURN]
20 GOSUB 150 [RETURN]
30 PRINT "Çiftliğinde kedileri var" [RETURN]
40 PRINT "Miyav miyav diye bağırır" [RETURN]
50 GOSUB 170 [RETURN]
60 GOSUB 150 [RETURN]
70 PRINT "Çiftliğinde köpekleri var" [RETURN]
80 PRINT "Hav hav diye bağırır" [RETURN]
90 GOSUB 160 [RETURN]
100 END [RETURN]
150 PRINT "Ali babanın bir çiftliği var" [RETURN]
160 RETURN [RETURN]
170 PRINT "Çiftliğinde Ali babanın" [RETURN]
180 RETURN [RETURN]
run [RETURN]
```

İyi tasarlanmış altprogramlar bilgi işlemin önemli kısımlarından biridir. Bu altprogramlar, programların yapısını sağlamlaştırırlar ve iyi programlama alışkanlığı kazandırırlar.

Her zaman aklımızda tutacağımız bir konuda, mutlaka alt programın belirli bir noktasına atlamamız gerekmez. (örneğin başlangıcına) bir altprogram, 500. satır ile 800. satır arasında yer alabilir ve biz GOSUB 500, GOSUB 640 veya GOSUB 790 diye çağırabiliriz.

Yukarıdaki programda 100. satıra END komutunu koyduk. Bu konutu koymasaydık proram 90. satırdan sonra devam edecek ve GOSUB ile çağrılmadıktan sonra gerekmeyen 150. satır işleme konulacaktı.

Basit Aritmetik

Bilgisayarınızı rahatlıkla hesap makinesi olarak da kullanabilirsiniz.

Bu işlemi daha iyi anlamak için aşağıdaki örnekleri yapınız. Bu örneklerde print yerine ? işaretini kullanacağız. Soruların cevapları [RETURN] tuşuna basar basmaz ekrana yazılır.

TOPLAMA

(artı işareti için [SHIFT] ve ; tuşlarını kullanın)

```
? 3+3 (RETURN)
  6

(= işaretini kullanmadığımıza dikkat ediniz)
?  8+4 (RETURN)
  12
```

ÇIKARTMA

[SHIFT]'siz işaretini kullanın)

```
? 4-3 [RETURN]
  4

? 8-4 [RETURN]
  4
```

ÇARPMA

[SHIFT] ve : tuşlarını kullanın)

```
? 3*3 [RETURN]
  9

 ? 8*4 [RETURN]
32
```

BÖLME

[SHIFT] siz ? işaretini (/) için kullanınız)

```
? 3/3 [RETURN]
  1

? 8/4 [RETURN]
  2
```

TAMSAYI BÖLMESİ

(Kalansız bir bölme için (\ ) işaretini kullanın)

```
? 10 \ 6 [RETURN]
  1

? 20 \ 3 [RETURN]
  6
```

MODÜLÜS

(Tamsayı bölmesinden sonra kalan kısım için MOD kullanın)

```
? 10 MOD 4 [RETURN]
  2

? 9 MOD 3 [RETURN]
  0
```

KAREKÖK

Bir sayının kare kökünü almak için sqr ( ) kullanın ve parantezin içine kare kökünü alacağınız sayıyı yazın.

```
? sqr (16) [RETURN] (bunun anlamı √16'dır)
  4

? sqr (100) [RETURN]
  10
```

ÜS ALMA İŞLEMİ

((SHIFT)'siz £ tuşunu kullanın)

```
? 3 ↑ 3 [RETURN] (3^3 demektir)

? 8 ↑ 4 [RETURN] (8^4 demektir)
  4096
```

KÜPKÖK

Küpkök işlemi üs alma örneğinde kullanılan metoda benzer.

27'nin küpkökünü bulmak için

? 27 ↑ (1/3) [RETURN] ($\sqrt[3]{27}$ demektir)

125'in küpkökünü bulmak için

? 125 ↑ (1/3) [RETURN]

Karışık Hesaplar

(+, -, *, /)

Bilgisayarla karışık hesaplar yapmak mümkündür. Fakat bu işlemlerin bir öncelik sırası vardır.

İlk öncelik sırası çarpma ve bölmeye, sonra toplama ve çıkartmaya verilir. Bu öncelikler sadece dört işlem hesaplarında uygulanır.

Yapılacak hesap aşağıdakigibi ise;

$3 + 7\text{-}2 \times 7/4$

Siz şöyle düşünebilirsiniz

$3 + 7\text{-}2 \times 7/4$
$= 8 \times 7/4$
$= 56/4$
$= 14$

Fakat gerçekte ise şöyle hesaplanır

$3 + 7\text{-}2 \times 7/4$
$= 3 + 7\text{-}14/4$
$= 3 + 7\text{-}3.5$
$= 10\text{-}3.5$
$= 6.5$

Bu hesabı ispat için örneği tekrar yazınız.

```
? 3+7-2*7/4 [RETURN]
  6.5
```

Siz parantezler ilave ederek bilgisayarın işlem yapış yolunu değiştirebilirsiniz. Bil-

gisayar parantezin dışında kalan çarpma, bölme işlemlerinden önce parantezin içindeki işlemlere öncelik verecektir.

```
? (3 + 7-2 ) * 7/4 [RETURN]
   14
```

Bütün aritmetik işlemlerdeki öncelik sırası şöyledir:

↑	üs alma
MOD	Modülüs
-	Rakamın negatif işareti
* ve /	Çarpma ve bölme
\	Tamsayı bölmesi
+ ve -	Toplama ve çıkartma

İleri İşlemler

Eğer hesaplarınızda çok büyük yada çok küçük sayılar kullanmak istiyorsanız bazen matematiksel özel semboller kullanmak yararlıdır. E harfi 10'un katı olan rakamların üssü için kullanılır. Burada küçük e veya büyük E kullanabilirsiniz.

Örneğin 300 ile 3×10^2 aynı sayıyı gösterir. Bununla birimsel gösterimi ise 3 E

2'dir. Aynı şekide 0.03 ile 3×10^{-2} aynı sayıyı gösterir ve bunun bilimsel gösterimi ise 3 E -2'dir. Şimdi aşağıdaki örnekleri deneyiniz.

```
? 30 * 10 [RETURN]
  300
```

Bu işlemi şöyle de yazabilirsiniz:

```
?3E1 * 1E1 [RETURN]
  300
```

```
? 3000 * 1000 [RETURN] veya ? 1E3 * 1E3[RETURN]
  3000000
```

```
? 3000 * 0.001 [RETURN] veya ?3E3 * 1E-3[RETURN]
   3
```

Bölüm 7: Kaydetme

Şimdi siz bir programı bilgisayardan diske kaydedip saklamayı ve diskden bilgisayara geri yüklemeyi öğreneceksiniz.

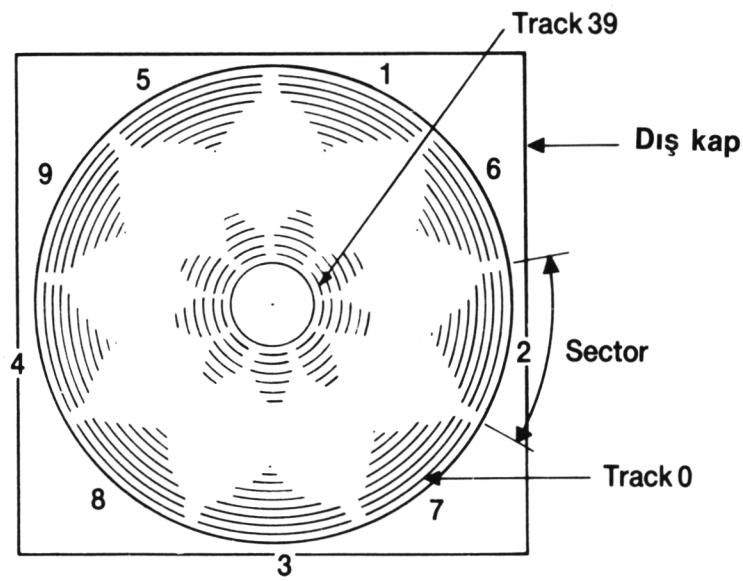
Siz kasete program kaydetmeyi ve kasetteki, kaydedilmiş programları bilgisayara yüklemeyi biliyor olsanız dahi diskdeki program dosyalarıyla ilgili bazı önemli bilgileri öğrenmeniz gerekmektedir.

Kaset ile disk arasındaki en önemli fark şudur. Boş bir diskin, kasette olduğu gibi kutusundan çıkartılıp üzerine kayıt yapılamamasıdır. Yeni bir disk önce formatlanmalıdır.

Diğer bir fark ise disk dosyalarındaki doğru isimlemenin (naming) önemidir. Kaset dosya isimleri genel olarak çok gevşek standartlar ihtave ederler. (uzunluk olarak değişiklik arzederler) Bu işlem disk de böyle değildir. Disk dosya adları kesinlikle CP/M standartlarında olmalıdır. Bu konu daha sonra izah edilecektir.

KULLANIM İÇİN DİSK FORMATLANMASI

Yeni bir boş bir diske herhangi bir veri yazmadan önce diskin kendisinin formatlanması gerekir. Formatlama; bilgiler yerleştirilmeden önce diskin üzerine birseri raflar, ayırıcılar yapmaya benzer. Diğer bir deyişle verilerin konup alınacağı yere bir çerçeve yerleştirmeye benzer. Formatlama, diski 360 kesin ayrı alana böler.



Diskde dışarıdan (kanal 0) içeriye (kanal 39) doğru 40 kanal vardır ve diskin çevresi 9 daire dilimine bölünmüştür.

Her daire dilimindeki kanalda 512 bayt'lık veri saklanabilir. Böylelikle diskin her bir yüzünde 180 KBayt'lık toplam kullanılabilir yer vardır.

Ana CP/M Sistem Disk Paketinin Kullanılması

Programlarınızın üzerine yazılacağı, üzerinden okunacağı, yeni ve boş bir diskin formatlanarak hazırlanması için ana CP/M sistem disk paketinizin 1. yüzünü kullanmanız gerekir. (Bu diskler bilgisayar ile verilmiştir)

Sistemi açınız ve sistem disk paketinizin 1. yüzünü sürücüye sokunuz.

Eğer 2 disk sürücüsüyle çalışıyorsanız sistem diskinin 1. yüzünü daima bilgisayarınızla birlikte olan (A sürücüsü) disk sürücüsüne giriniz.

Şimdi aşağıdaki komutu yazınız:

```
Icpm [RETURN]
```

(Çubuk sembolünü [SHIFT] tuşuna basarken @ tuşuna basarak yazabilirsiniz)

Birkaç saniye sonra ekranın yukarısında şu mesajı görürsünüz.

```
CP/M Plus Amstrad Comsumer Electronics plc
```

Bu mesaj, bilgisayarın CP/M artı çalışma sisteminin kontrolu altında olduğunu belirtir. Aynı zamanda ekranda kursör ile birlikte A > harfide göreceksiniz. Bu "prompt"dur (Ready gibi) ve bilgisayarın sizin komutlarınızı beklediğinizi belirtir.

CP/M ile çalışırken BASIC komutlarını bilgisayara veremezsiniz çünkü anlaşılmaz. Bilgisayara şu emri giriniz:

```
cls [RETURN]
```

Bilgisayar bu komutu soru işareti ile birlikte size geri gönderip, komutun anlaşılmadığını belirtir.

```
CLS ?
```

Şimdi bazı CP/M komutlarını inceleyelim. Bilgisayara şu komutu girelim.

```
dir [RETURN]
```

Ekranda CP/M'in rehberini (DIRectory) ve kullanım komutlarını görürsünüz ki bunlardan biri de DISCKIT3'dür.

disckit3 [RETURN]

Birkaç saniye sonra ekranın üst tarafında DISC KIT açılış mesajını görürsünüz. Bunu şu mesaj takip eder.

One drive found

Bu mesaj sizin DISC KIT kullanım programını çalıştırdığınızı söyler ve bilgisayar bir disk sürücüsüyle çalıştığınızı kontrol etmiştir. Eğer ilave bir disk sürücüsü sistemi varsa, mesaj şöyle çıkacaktır.

Two drives found

Ekranın altında ise şunu görürsünüz.

Copy	7
Format	4
Verify	1
Exit from program	Ø

Bu şekil DISC KIT seçenek (menü) tablosudur. Kutudaki numaralar tuş takımının sağındaki fonksiyon tuşlarına tekabül eder. (f0, f1, f4, f7) Bunlardan birine basarak listeden istediğinizi seçebilirsiniz.

Fonksiyon tuşlarından Ø'a bastığınızı kabul edelim. Bu anda bilgisayar DISC KIT programından çıkar CP/M'e geri döner.

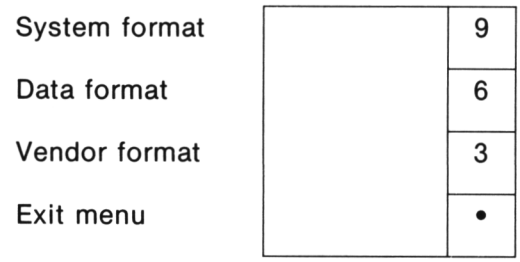
Şimdi bir diski formatlayalım. Bunun için fonksiyon tuşlarından 4'e basalım.

DİKKAT

DAHA ÖNCE KAYDEDİLMİŞ DİSKİ FORMATLAMAK

HERŞEYİ SİLER

Şimdi size seçiminiz için değişik formatlar sunan yeni seçenek tablosu görüyorsunuz, şöyleki;



Daha önce olduğu gibi fonksiyon tuşlarından birine basınız (f3, f6, f9) ve istediğiniz tip formatı seçiniz. Bütün bu farklı formatlar daha sonra açıklanacaktır fakat şimdilik Data formatı (veri formatı) 6 nolu tuşa basarak seçiniz.

Şunu unutmayınız ki (•) tuşuna (f3, f6, f9'un altında) basınca program format konumundan çıkıp ana DISC KIT seçenek tablosuna geri döner.

6 nolu fonkisyon tusuna basıldığını ve bilgisayarınıza ilave disk sürücüsü bağlanmamış olduğunu kabul edersek şu mesajı görürsünüz.

Y Format as Data
Any other key to exit menu

Bu anda, ana CP/M sistem diskini çıkartınız ve formatlamak istediğiniz diski formatlanacak tarafı üste gelecek şekilde sürücüye sokunuz.

Şimdi Y tuşuna basınız.(Y evet anlamına gelmektedir) Disk kanal Ø'dan kanal 39'a kadar formatlanır ve ekranın sol üstünde kanal numarası belirtilir.

Yazı koruma deliği açık olan disket formatlanamaz. Şayet bu şekilde formatlamak isterseniz şu mesajı ekranda görürsünüz.

Disc write-protected
Insert disc to format
R-etry or C-ancel

İptal için C yazıp diski çıkartabilirsiniz ve yazı koruma deliği kapalı olan formatlanacak doğru diski formatlayabilirsiniz.

Saklamak istediğiniz disklerinizin ve ana CP/M diskinizin üzerindeki yazı koruma deliğini hiçbir zaman kapatmayınız.

Formatlama bittiğinde devam için herhangi bir tuşa basınız.

Böyle yaptıktan sonra diğer diski formatlamak için diski sokunuz ve Y tuşuna basınız. Bu işlem sizin ihtiyacınız olan tüm formatlanacak diskler için tekrarlanabilir.

Diskin formatlanması bittiği zaman herhangi bir tuşa (Y'den farklı) basınız ve tekrar DISC KIT seçenek tablosuna dönünüz.

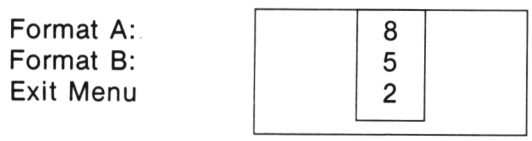
Seçenek tablosundaki Copy ve Verify daha sonra izah edilecektir. Bilgisayarı [CONTROL] [SHIFT] ve [ESC] kullanarak resetleyiniz.

CP/M sistem diskinin esas kopyasını iyi ve emniyetli bir yerde saklayınız. Daha sonraki konularda size sistem diskinin çalışan kopyalarının nasıl yapılacağı gösterilecektir.

2 Sürücü Sisteminde Formatlama

Yukarıdaki komut ve bilgileri uygulayarak ana DISC KIT tablosundan Format kısmını f4 tuşuna basarak seçiniz ve sonra f6 nolu tuşa basarak Data format kısmını seçiniz.

Bu anda size, formatlamak için sürücü seçimini sunan üçüncü bir menü çıkacaktır.



Format B: seçeneği (f5 tuşu) sistem diskini (1. yüzü) sürücü A da bırakıp, formatlanacak diski B sürücüsüne yerleştirme imkanı sağlar.

Format B: yi seçtikten sonra siz, formatlama işlemi için Y tuşuna basabilirsiniz veya DISC KIT seçenek tablosuna çıkmak için herhangi bir tuşa basabilirsiniz.

Eğer Format A: seçilmişse (f8 tuşu) sistem diskinizi A sürücüsünden çıkarmayı unutmayınız ve yerine formatlanacak diski yerleştiriniz.

Şunu hiç unutmayın: ANA CP/M SİSTEM DİSKİNİN ÜSTÜNE YAZMAYA KALKIŞMAYIN

Şimdi elimizde formatlanmış boş bir (iki) disk vardır. Artık BASIC programları diske yükleyip tekrar çağırabiliriz.

Programın Disk Üzerinde Hafızaya Alınması

Bilgisayara yazılmış programı, disk üzerinde saklamak için şunu yazınız.

save "dosya ismi" [RETURN]

Programın isimlendirilmesi mecburidir.

Diskdeki bir dosya adı iki kısımdan meydana gelmiştir. Birinci kısım mecburidir ve 8 karakterden uzun olamaz. Burada harfler ve rakamlar kullanılır fakat ara ve noktalama işaretleri kullanılmaz. Bu birinci alan, programın adını ihtiva eder.

İkinci alan ise isteğe bağlıdır. Burada 3 karakter kullanabilirsiniz, fakat yine ara ve noktalama işaretleri kullanılamaz. İki alan birbirinden (•) ile ayrılırlar.

Eğer ikinci alanı belirtmiyorsanız, sistem otomatik olarak kendisi başlık koyar. Örneğin BASIC dosyaları için. BAS veya binary (makine kodu) için .BIN gibi.

Diskde saklamaya bir örnek olarak kısa bir program yazın ve formatlanmış diski yerleştirip, şunu yazınız.

save "örnek" [RETURN]

Birkaç saniye sonra Ready hatırlatması ekranda belirir ve program diske kayıt edilmiş olur. (Eğer bu işlem olmazsa ekranda hata mesajı olup olmadığını kontrol ediniz. Hata mesajı var ise; diskinizi doğru sürücüye girmediniz veya yazı koruma deliğini kapatmadınız yada yanlış komut verdiniz demektir.)

Katalog

Programı sakladıktan sonra, şunu yazınız:

cat [RETURN]

Ekranda şunları göreceksiniz.

Drive A: user 0

ÖRNEK.BAS 1K

177 K free

Dosya adı ve belirlenmiş veya alınmış ikinci alan ve dosya uzunluğu (en yakın büyük Kbayt) ekranda görünür, diskdeki serbest alan miktarıda ekrana gelir.

Diskden Yükleme

Şu komutu kullanmakla program diskden yüklenir ve çalışmaya başlar:

load "dosya ismi" [RETURN]
run [RETURN]

veya run direkt olarak kullanılabilir.

run "dosya ismi" [RETURN]

Kilitli programlar yalnız run ile çalışırlar.

IA ve IB

Eğer ilave bir disk sürücüsü ile çalışıyorsanız hangi sürücünün (A vey B) çalışması gerektiğini şu komutu yazarak sağlayabiliriz.

Ia [RETURN]

veya

Ib [RETURN]

Bunlar SAVE, CAT veya LOAD komutlarından önce yazılır.

Diskden Diske Program Kopyası

Bu bölümde, şu ana kadar öğrendiğimiz komutlarla, diskden diske program kopyasını; orjinal kaynak diskinden programı bilgisayara yükleyip sonra kaynak diskini çıkarıp yeni diski girerek programı ona yükleme şeklinde yapabiliriz.

Programı iki disk sürücülü sistemde kopya için kaynak diskimizi B sürücüsüne gireriz. Kaydedilecek diskimizi de A sürücüsüne gireriz. Programı kopya için şunu yazınız.

Ib [RETURN]
load "dosya ismi" [RETURN]
Ia [RETURN]
save "dosya ismi" [RETURN]

CPC 6128'de normal BASIC dosya saklanmasına ilave olarak çok özel maksatlar için kullanılan üç metot daha vardır.

ASCII Dosyaları

save "dosya ismi", a [RETURN]

Sondaki ,a eki; bilgisayara programın veya datanın ASCII metin dosyası formunda saklanacağını bildirir. Bu metotla saklanan veriler kelime işlemciler (word processors) ve diğer uygulama programları tarafından meydana getirilen dosyalarda kullanılır. Bunun kullanılması ileride aplikasyonlar ele alındığında izah edilecektir.

Korunan Dosyalar

save "dosya ismi", p [RETURN]

Son (,p) ekini ilave etmek bilgisayara verileri o şekilde koru ki, program yüklendikten sonra LIST edilmesin veya çalışırken ancak [ESC] tuşunu kullanarak durabilsin anlamına gelir.
Bu şekilde saklanmış programlar direkt olarak şu şekilde çalıştırılabilirler.

run "dosya ismi" [RETURN]

veya

chain "dosya ismi" [RETURN]

Eğer programı değiştirmek veya düzeltmek isterseniz, kendinize korunmamış bir kopya çıkartmalısınız.

Binary Dosyalar

save "dosya ismi", b <başlangıç adresi>, <bayt olarak uzunluğu>, [isteğe bağlı giriş noktası] [RETURN]

Bu seçenek bilgisayarın RAM'ında bulunan komple veri bloklarını, hafızada gözüktüğü gibi binary olarak diskde saklamayı sağlar. Kaydetmeye, hafızadaki bölümün neresinden başlayacağını, uzunluğunu ve gerekiyorsa program olarak çalıştırılan dosyanın uygulamaya sokulacak bellek adresini bilgisayara bildirmeniz gerekir.

Ekran Dökümü

Bu, binary koruma işlemi, ekran hafızasındaki dataları direkt olarak, ekran dökümü formunda diskde saklar. Ekranın muhteviyatının görüldüğü gibi saklanması için şu komut kullanılır.

save "scrndump",b, 49152, 16384 (RETURN)

49152 ekran belleğinin başlangıç adresidir ve 16384 ise saklamak istediğiniz ekran belleğinin uzunluğudur.

Ekrana geri çağırmak için şunu yazınız.

load "scrndump" [RETURN]

Son olarak bu kitapçığın başlangıcında ÖNEMLİ başlığında verilen,

MONTAJ BİLGİLERİ 5, 6, 7
ÇALIŞTIRMA BİLGİLERİ 1, 2, 3, 4, 5, 6, 7, 9,

bilgilerini tekrarlayınız.

Bölüm 8: Renk, Grafik ve Hal (Mode)

Amstrad CPC 6128 renkli personel bilgisayarın üç tane ekran modu vardır. Mode Ø, Mode 1, Mode 2

Bilgisayar ilk açıldığı an otomatik olarak Mode 1'dedir.

Değişik modları anlamak için bilgisayarı açınız ve 1 tuşuna basınız ve iki satır 1 ile doluncaya kadar basmaya devam ediniz. Şayet 1'leri sayarsanız (bir satırdaki) 40 tane olduğunu göreceksiniz. Bu Mode 1'de 40 kolon var demektir. [RETURN] tuşuna basınız, Syntax error mesajını alacaksınız fakat merak etmeyin, bu Ready mesajına geri dönmenin en kolay ve çabuk yoludur ki bu mesaj ile bilgisayar sizden yeni komut bekliyor demektir.

Şimdi şunu yazınız:

mode Ø [RETURN]

Ekrandaki karakterlerin daha büyük olduğunu göreceksiniz. Yukarada bahsedilen işlemi aynen yapınız, ekranda ilk satırdaki 1'leri sayarsanız 20 tane olduğunu göreceksiniz. Bu mode 0'da 20 kolon var demektir. Tekrar [RETURN] tuşuna basınız.

Şimdi şunu yazınız:

mode 2 [RETURN]

Görüyorsunuz ki mode 2 en küçük karakteridir. Yine 1 tuşuna sürekli basınız, bu sefer bir satırda 80 tane 1 sayarsanız, yani mode 2'de 80 kolon var demektir.

özetlersek:

Mode Ø = 20 kolon
Mode 1 = 40 kolon
Mode 2 = 80 kolon

karakter yazıyor demektir.

Renkler

27 renk seçeneği vardır. Bunlar yeşil monitörde (GT65) yeşilin değişik tonları olarak gösterilmiştir. Eğer GT65 monitörü aldıysanız, bilgisayarın renk olanaklarını ev-

deki TV'de kullanmak için AMSTRAD MP2 modülatör/güç kaynağı satın alabilirsiniz.

Mode 0'da 27 renkten 16'sını istediğiniz zaman ekrana getirebilirsiniz.
Mode 1'de 27 renkten 4'ünü kullanabilirsiniz.
Mode 2'de 27 renkten 2'sini kullanabilirsiniz.

BORDER (kenar), PAPER (sayfa), PEN (karakterin kendisi) renkleri birbirinden bağımsız olarak değiştirilebilir. Elde edilen 27 renk Tablo 1'de INK (mürekkep) rengi referans numaralarıyla liste halinde verilmiştir.

ANA RENK KARTI

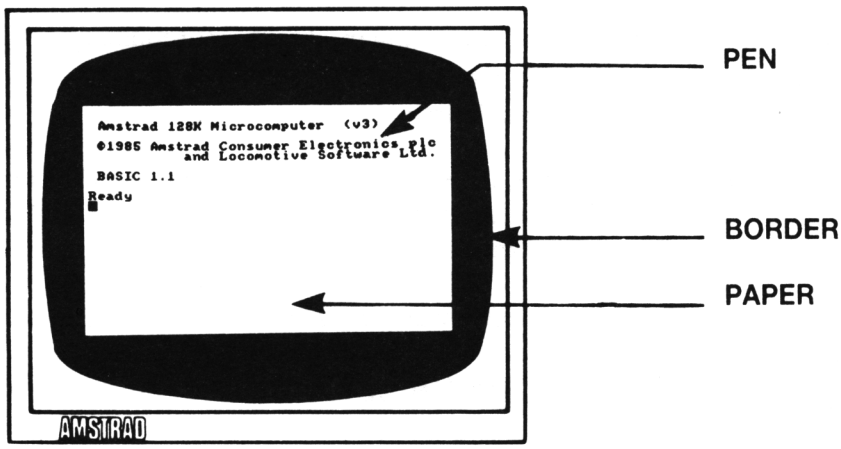
INK NO	RENK/INK	INK NO	RENK/INK
0	Siyah	14	Pastel Mavi
1	Mavi	15	Portakal
2	Parlak Mavi	16	Pembe
3	Kırmızı	17	Pastel Magenta
4	Magenta	18	Parlak Yeşil
5	Leylak	19	Deniz Yeşili
6	Parlak Kırmızı	20	Parlak Cyan
7	Mor	21	Kireç Yeşili
8	Parlak Magenta	22	Pastel Yeşil
9	Yeşil	23	Pastel Cyan
10	Açık Mavi	24	Parlak Sarı
11	Gök Mavisi	25	Pastel Sarı
12	Sarı	26	Parlak Beyaz
13	Beyaz		

Tablo 1; Mürekkep Renkleri ve Numaraları

Daha önce açıklandığı gibi, bilgisayar Mode 1 durumundadır. Diğer Mode'lardan Mode 1'e dönmek için şu yazılır.

```
mode 1 [RETURN]
```

Ekran Görüntüsü



BORDER kısmı, PAPER (kağıt) bölümünü çevreleyen kısımdır. Dikkat ederseniz bilgisayar ilk açıldığında BORDER ve PAPER mavi renktedir. PAPER, karakterlerin arka fonudur ve bu fon üzerine yazılan karakterlerin rengini PEN belirler.

Şimdi ekranda gördüğünüz renkleri nasıl seçeceğinizi ve değiştirebileceğinizi göstereceğiz.

Bilgisayar ilk açıldığı zaman BORDER daima 1 nolu renge ayarlıdır. Bu renk mavidir. Border rengini değiştirmek için border komutu ve bunu takip eden renk numarası yazılır. Şunu yazınız:

border 13 [RETURN]

Şimdi işin en ustalık isteyen bölümüne geldik.
Bilgisayar açıldığı zaman PAPER numarası Ø ve PEN numarası 1 olarak otomatik seçilir. Bu tablo 1'deki Ø ve 1 numaralarına bakmak anlamına gelmez.

Burada Ø ve 1; PAPER ve BORDER numaralarıdır, INK renk numaraları yoktur. Şimdi şöyle düşünelim: Masamızda Ø, 1, 2, 3 diye numaralandırılmış 4 dolmakalemizi ve bu kalemlere doldurabileceğimiz, Ø'dan 26'ya kadar 27 renk mürekkep var. Buradan da anlaşılıyorki 1 nolu kalemimiz her zaman aynı renk mürekkep ile dolu değildir, değişik renkte mürekkep ile dolu olabilir ve 4 dolmakalem de aynı renk mürekkep ile doldurulmuş olabilir.

Bu iş, bilgisayarınızda da böyledir. PEN ve INK komutlarını kullanırken kalem numarasını ve bu kalem (PEN) için mürekkep (INK) rengini seçebilirsiniz.

Mode 1'de (40 kolon) çalıştığımızı düşünerek aşağıdaki Tablo 2'den 1. ve 3. kolonlarda PEN numarası 1'e karşılık olan INK renk numarası 24'dür. Tablo 1'deki INK Numarası 24 olan renk parlak sarıdır.

BAŞLANGICDAKİ AYARLAMALAR

Paper/Pen No	Ink Rengi Mode Ø	Ink Rengi Mode 1	Ink Rengi Mode 2
Ø	1	1	1
1	24	24	24
2	2Ø	2Ø	1
3	6	6	24
4	26	1	1
5	Ø	24	24
6	2	2Ø	1
7	8	6	24
8	1Ø	1	1
9	12	24	24
1Ø	14	2Ø	1
11	16	6	24
12	18	1	1
13	22	24	24
14	Yanıp sönme 1,24	2Ø	1
15	Yanıp sönme 16,11	6	24

Tablo 2: PAPER/PEN/MODE/INK referansları

Tablo 2'de verilen PAPER/PEN/INK bağıntıları sabit değildir. Siz INK kullanarak bunları değiştirebilirsiniz. Komut iki kısımdan meydana gelir; 1. kısım PAPER veya PEN numarasıdır ki siz INK (mürekkep) dolduracaksınız ve 2. kısım ise INK'in (mürekkep veya renk) rengidir. Komuttaki bu iki kısım birbirinden (,) ile ayrılır.

PEN numarası 1'i kullandığımızı biliyoruz ve bu kalemin (PEN) INK rengini portakal rengine değiştirelim. Şunu yazınız.

```
ink 1,15 [RETURN]
```

Ekrandaki karakterlerin renginin değiştiğini görürsünüz.

Fon rengide INK ile değiştirilebilir. Bilgisayar açılınca PAPER numarasının Ø olarak seçildiğini biliyoruz, Ø nolu PAPER'ın rengini INK'in rengini değiştirmekle değiştirebiliriz. PAPER'ın rengini yeşil yapmak için (9 nolu renk) INK'in rengini şu komutu kullanarak değiştirelim.

```
ink Ø,9 [RETURN]
```

Şimdi hepsi için değişik bir pen kullanalım.

pen 3 [RETURN]

Yeni karakterlerin renginin nasıl değiştiğine dikkat ediniz.

Şimdi siz 3 nolu PEN'i kullanıyorsunuz. Tablo 1 ve Tablo 2'den önceden doldurulmuş olan 3 nolu PEN'in INK renginin 6 (parlak kırmızı) olduğunu göreceksiniz. Bunu pembeye değiştirmek için şunu yazınız.

ink 3,16 [RETURN]

Burada 3 PEN, 16 ise INK rengidir.
Şimdi yeni bir sayafaya geçelim. Yeni bir sayfa seçildiği zaman önceki fon rengi (karakterlerin arkasındaki) değişmeyecektir. Çünkü bu renk farklı bir paper komutu ile basılmıştır. Bunu görmek için şunu yazalım.

paper 2 [RETURN]

Yine Tablo 1 ve 2'yi kullanarak fon renginin parlak cyan olduğunu görürüz. Siyah yapmak için şunu yazın.

ink 2,Ø [RETURN]

Şimdi ekranda 1 ve 3 nolu PEN ile Ø ve 2 nolu PAPER fonu üzerinde yazılmış karakterler vardır. Kullanmayacağınız PEN veya PAPER, INK'leri değiştirebilir. Örneğin:

ink 1,2 [RETURN]

PEN no 1 yazılmış tüm karakterlerin rengini değiştirdi.

cls [RETURN]

yazarak ekranı temizleyiniz.

Burada bilgisayara gerekli komutları vererek orjinal renklerine (çerçeve ve fon mavi, karakterler parlak sarı) (BORDER, PAPER, PEN ve INK ile) döndürebilirsiniz. Şayet bunu yapamazsanız bilgisayarı [CONTROL] [SHIFT] ve [ESC] tuşları ile resetleyiniz.

Yanıp Sönen Renkler

Karakterlerin renklerini, bu renkten diğer bir renge yanıp sönmelerini sağlamak

mümkündür. Bu işlem PEN'in INK komutuna bir fazla renk nosu ilave etmekle sağlanır.

Karakterlerin parlak beyaz ve parlak kırmızı arasında yanıp sönmesi için bilgisayarı resetleyiniz. Şunu yazınız.

ink 1,26,6 [RETURN]

Bu durumda "1" PEN numarası, "26" parlak beyaz rengin numarası ve "6" da değişecek parlak kırmızıdır.

Karakterlerin arkasındaki PAPER'ın da renginin iki renk arasında yanıp sönmesi sağlanabilir.

Karakterlerin arkasındaki PAPER'ın yeşil ve parlak sarı arasında yanıp söndüğünü görmek için şunları yazınız.

ink Ø,9,24 [RETURN]

Bu durumda "Ø" PAPER numarası, "9" yeşil, "24" ise parlak sarıdır. Şimdi bilgisayarı yeniden resetleyiniz.

Mode Ø'da PEN'in 14 ve 15 numaraları ile PAPER'ın 14 ve 15 numaraları zaten yanıp sönen renkler oldukları için bunlara fazladan INK komut numarası yazılmaz.

Şimdi şunu deneyelim:

mode Ø [RETURN]
pen 15 [RETURN]

Ekranda Ready sözcüğünün gökmavisi ve pembe arasında yanıp söndüğünü göreceksiniz.

Bu sefer de:

paper 14 [RETURN]
cls [RETURN]

Şimdi görüyorsunuz ki bir yandan Ready sözcüğü, gökmavisi ve pembe arasında yanıp sönerken fondaki PAPER da sarı ve mavi yanıp sönmektedir.

Bu yanıp sönen renkleri de PEN ve PAPER için yeni bir INK komutu ile değiştirmek veya kalıcı hale getirmek mümkündür.

BORDER'ın da iki renk arasında yanıp sönmesini bir renk numarası ilavesiyle sağlayabiliriz. Şunu yazınız:

border 6,9 [RETURN]

Şimdi BORDER'ın parlak kırmızı ve yeşil arasında yanıp söndüğünü görüyorsunuz. Şunu unutlayınız; BORDER, 27 renkten birine veya ikisine ayarlanabilir. (Hangi mode da olduğunuz farketmez)

Bilgisayarınızı [CONTROL] [SHIFT] [ESC] ile resetleyin.

```
10 MODE 0 [RETURN]
20 rate=600: REM programının hızını ayarlar [RETURN]
30 FOR b=0 TO 26 [RETURN]
40 LOCATE 3,12 [RETURN]
50 BORDER b [RETURN]
60 PRINT "border rengi";b [RETURN]
70 FOR t=1 TO rate [RETURN]
80 NEXT t,b [RETURN]
90 CLG [RETURN]
100 PAPER p [RETURN]
120 PRINT "paper";p [RETURN]
130 FOR n=0 TO 15 [RETURN]
140 PEN n [RETURN]
150 PRINT "pen";n [RETURN]
160 NEXT n [RETURN]
170 FOR t=1 TO rate*2 [RETURN]
180 NEXT t,p [RETURN]
190 MODE 1 [RETURN]
200 BORDER 1 [RETURN]
210 PAPER 0 [RETURN]
220 PEN 1 [RETURN]
230 INK 0,1 [RETURN]
240 INK 1,24 [RETURN]
run [RETURN]
```

ÖNEMLİ

Yukarıdaki programda ve daha sonraki bölümlerde ve bu kitapçığın listelerinde basıc komutları büyük harflerle yazılmıştır. Bu bilgisayar tarafından program LIST yapıldığı zaman komutların gözüktüğü gibidir. Genel olarak siz program yazarken komutları küçük harf ile yazınız. Böylelikle yazı hatalarınızı programı LIST yaptığınız zaman (yanlış yazılmış BASIC komutları büyük harfe çevrilmez) kolaylıkla farkedersiniz.

Değişkenlerin adları (x ve a$ gibi) program LIST yapıldığı zaman büyük harfe dönüştürülmez.

Dikkat

Bundan böyle her satırdan sonra [RETURN] tuşuna basmanızı hatırlatmaya cağız. Bunu otomatik olarak yaptığınız var sayıyoruz.

Grafikler

Bilgisayarın belleğinde birçok karakter sembolleri vardır. Bunlardan birini yazmak için aşağıdaki fonksiyon kullanılır.

chr$ ( )

Parantezin içine 32'den 255'e kadar uzanan sembol numarasını koymanız gerekir.

[CONTROL] [SHIFT] [ESC] tuşlarına basarak bilgisayarı resetleyiniz ve şunu yazınız:

```
print chr$ (25Ø)
```

Ekranda 25Ø kod numaralı sağa doğru yürüyen bir adam sembolü göreceksiniz.

Bütün karakterleri ve sembolleri görmek için aşağıdaki programı yazınız. (her satırdan sonra RETURN tuşuna basmayı unutmayınız)

```
1Ø for n = 32 to 255
2Ø print n;chr$(n);
3Ø next n
run
```

Gerektiğinde başvurmanız için karakterlerin türleri ve ilgili kod numaraları "Bilginiz için" adlı bölümde verilmiştir.

Locate

Bu emir; karekter kursörünün ekranda belirtilen bir tarafa gitmesini sağlar. Locate emriyle değiştirilmediği takdirde karakter kursörü ekranın sol üst köşesinden başlar. Yani x,y koordinatları 1,1'dir. (x yatay, y dikey pozisyonu belirtir) Mode 1'de 4Ø kolon ve 25 satır vardır. Bir karakteri üst satırın tam ortasına yerleştirmek istersek (mode 1'de) x,y koordinatları için 2Ø,1 kullanmalıyız. Bu durumu görmek için şunu yazalım.

mode 1 (ekran temizlenir ve kursör sol üst köşeye gider)

```
1Ø locate 2Ø,1
2Ø print chr$(25Ø)
run
```

Bunun en üst satır olduğunu kanıtlamak için şunu yazın:

```
border Ø
```

BORDER siyah oldu ve adamı en üst satırın ortasında göreceksiniz.

Mode Ø da 2Ø kolon ve 25 satır vardır. Şunu yazınız:

```
mode Ø: run
```

Şimdi ise adamı ekranın sağ üst köşesinde göreceksiniz. Bunun sebbi Mode Ø da, x koordinatının belirttiği 2Ø sayısı en son kolondur.

Mode 2 de ise 80 kolon ve 25 satır vardır. Aynı programı kullanarak adamın nerede olduğunu tahmin edebilirsiniz. Şunu yazınız:

```
mode 2: run
```

Mode 1'e dönmek için ise şu komutu yazınız:

```
mode 1
```

Şimdi kendi kendinize locate ve chr$( ) komutlarını daha iyi öğrenmek için deney yapmak üzere değişik pozisyon ve karakter numaraları koyunuz. Örneğin;

```
locate 2Ø,12: print chr$(24Ø)
```

Ekranı tam merkezinde bir ok göreceksiniz. Yukarıdaki örnekte;

20 yatay (x) koordinatı (1 ile 40 arası)
12 dikey (y) koordinatı (1 ile 25 arası)
240 karakter sembolü (32 ile 255 arası) gösterir.
Karakter sembolünü ekran boyunca tekrarlamak için şu programı yazın.

```
1Ø CLS
2Ø FOR x = 1 TO 39
3Ø LOCATE x,2Ø
5Ø PRINT CHR$ (25Ø)
6Ø NEXT x
7Ø GOTO 1Ø
run
```

Durdurmak için [ESC] tuşuna iki kez basınız.
Yeni bir karakter yazmadan önce bir evvelki karakteri silmek için şunları yazın:

```
5Ø print "  ";chr$(25Ø)
```

Yeni yazılmış 50. satır otomatik olarak bir önceki 50. satırın yerini alacaktır. Şimdi şunu yazınız.

```
run
```

FRAME

Karakterin ekran üzerindeki hareketini sağlamak için şu satırı ekleyiniz.

```
40 frame
```

FRAME komutu; ekrandaki nesnenin hareketini, ekran tarama frekansına senkronize eder. Bu komutu, karakteri veya grafiği ekran üzerinde yavaşça hareket ettirmek için kullanabilirsiniz.

Bu program bazı geciktirme çevrimleri ve değişik karakter sembolleri kullanarak genişletildiğinde daha iyi anlaşılır. Şunu yazınız:

```
list
```

Şimdi aşağıdakileri ekleyiniz.

```
70 FOR n=1 TO 300:NEXT n
80 FOR x=39 TO 1 STEP-1
90 LOCATE x,20
100 FRAME
110 PRINT CHR$ (251);"  "
120 NEXT x
130 FOR N=1 TO 300:NEXT n
140 GOTO 20
run
```

PLOT

LOCATE komutunun tersine PLOT komutu pixel koordinatlarını kullanarak grafik kursörün yerini belirlemek için kullanılır. (Pixel ekranın mümkün olan en küçük parçasıdır)

Grafik kursör ekranda görünmez yani karakter kursöründen farklıdır.

Ekranda 640 yatay, 400 dikey pixel vardır. x,y koordinatları ekranın x,y koordinatları 0,0 olan sol alt köşesinde yer almıştır. Locate komutunun tersine yerleri MODE 0, 1 ve 2'de değişmez.

Bu durumu iyice görmek için bilgisayarı resetleyin ve şunları yazınız.

```
plot 320,200
```

Küçük bir nokta ekranın merkezinde görülecektir.

Şimdi mode değiştirelim.

```
mode Ø
plot 32Ø,2ØØ
```

Nokta hâlâ ortadadır fakat daha büyüktür. Tekrar mode değiştirelim.

```
mode 2
plot 32Ø,2ØØ
```

Noktanın yeri değişmemiş fakat ufalmıştır.

PLOT komutuna iyice alışmak için ekranın değişik yerlerine, değişik mode'lar kullanarak noktalar koyunuz. Bitince tekrar MODE 1'e dönün ve ekranı temizleyiniz.

DRAW

Önce bilgisayarınızı resetleyiniz. DRAW (çiz) komutu grafik kursörünün son durumundan başlayarak çizgi çizmeye yarar. Bunu daha iyi anlamak için aşağıdaki program ile bir dikdörtgen çizelim.

PLOT komutunu kullanarak grafik kursörünü istediğimiz yere getiriyoruz sonra çizime başlıyoruz.

```
5 cls
1Ø plot 1Ø,1Ø
2Ø draw 1Ø,39Ø
3Ø draw 63Ø,39Ø
4Ø draw 63Ø,1Ø
5Ø draw 1Ø,1Ø
6Ø goto 6Ø
run
```

Programı durdurmak için [ESC] tuşuna iki defa basınız.

(60. satırda bilgisayara 60. satır çerçevesinde bir çevrim yapması söylenmiştir. Yapılan bu çevrim (loop) ancak [ESC] tuşuna iki kez basmakla durdurulabilir. Bu tip komutlar, program bittiğinde ekranda Ready yazısını görmek istemediğimiz zaman faydalıdır.)

Şimdi 1. dikdörtgenin içine 2.sini çizelim.

```
6Ø plot 2Ø,2Ø
7Ø draw 2Ø,38Ø
8Ø draw 62Ø,38Ø
9Ø draw 62Ø,2Ø
1ØØ draw 2Ø,2Ø
11Ø goto 11Ø
run
```

Programı durdurmak için [ESC] tuşuna iki defa basınız.

MOVE

MOVE (hareket) komutu PLOT komutu ile benzer gaye için kullanılır, grafik kursörü belirlenen x,y koordinatlarına hareket eder fakat pixel (nokta) grafik kursörün yeni yerinde görünmez (çizilmez). Şunu yazınız.

```
cls
move 639, 399
```

Grafik kursörü sağ üst köşeye göndermemize rağmen hiç bir işaret göremedik.

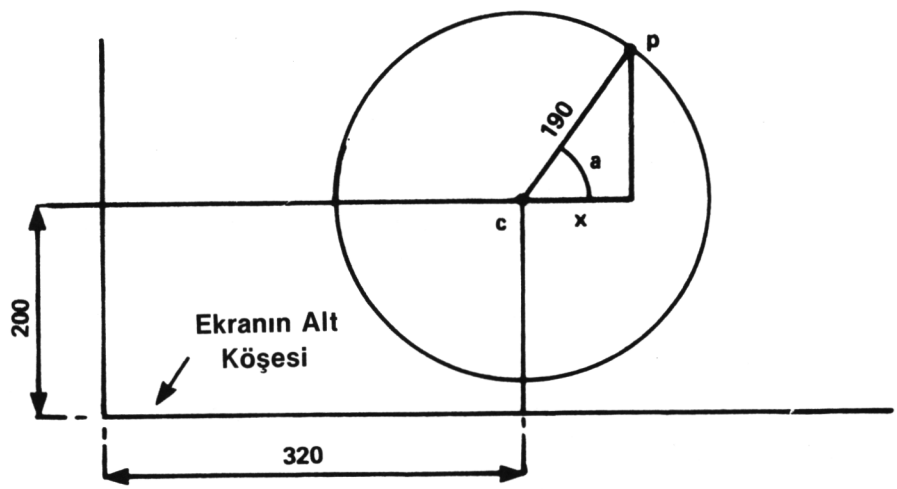
Bu durumu, bu noktadan ekranın merkezine bir çizgi çizerek kanıtlayalım.

```
draw 320,200
```

Çemberler

Çemberler, PLOT emri kullanılarak yapılabilir. PLOT'la çember oluşturma çember üzerindeki her noktanın x,y koordinatlarını işaretleyerek olur. Aşağıdaki şekilde çember üzerindeki p noktasının x,y koordinatları;

x = 190 × cos (a)
y = 190 × sin (a)



Plot ile Çember Çizimi

Daha önceki programlarda ekranın sol alt köşesine göre noktalar işaretlemiştik. Eğer biz çemberi ekranın ortasına yerleştirmek istersek, çemberin merkezinin koordinatlarını 320,200'e işaretlememiz gerekir. Sonra merkez pozisyon koordinatlarını ilave ederek çemberin tüm noktalarını ayarlarız. Buna göre çember şöyle çizilir.

```
new
10 CLS
20 DEG
30 FOR a=1 TO 360
40 MOVE 320,200
50 PLOT 320+190*COS (a), 200+190*SIN (a)
60 NEXT
RUN
```

Yukarıdaki programa başlarken NEW komutu yazıldı. Bu komut, bilgisayara hafızasındaki herhangi bir programı temizlemesini bildirir, fakat ekran temizlenmez.

Dairenin çapı, 190 sayısını (190 pixel ile alakalıdır) azaltarak küçültülebilir.

Dairenin daha değişik bir biçimde çizilmesi için (radyan) 20. satırı değiştirelim. Şunu yazınız.

```
20
```

Merkezden çizilen çizgilerden oluşan, içi dolu daire çizmek için 50. satırdaki plot komutunu draw ile değiştirelim.

```
50 draw 320+190*cos (a), 200+190*sin (a)
```

Bu işlemi 20. satır varken ve yokken bir daha deneyiniz.

Dikkat ederseniz 60. satırda NEXT a yerine NEXT yazmak yeterlidir. Bilgisayar bu NEXT'in hangi FOR'a ait olduğunu çıkartabilir. Programın içinde birden fazla FOR ve NEXT çevrimi varsa, değişkenlerin adlarını koymanız gerekmektedir.

BAŞLANGIÇ NOKTASI

Daha önceki programda dairenin merkezini belirtmek için MOVE komutunu kullandık ve x,y koordinatlarını bu merkez konumuna ilave ettik. Bu merkez koordinatlarını, PLOT komutuyla belirtilmiş merkez koordinatlarına eklemek yerine ORIGIN komutunu kullanabiliriz. Bu komut önce dairenin merkezini, sonra da x,y koordinatlarını bu ORIGIN'e göre belirler. Bunun için aşağıdaki programı yazalım.

```
new
10 cls
20 for a=1 to 360
30 origin 320,200
40 plot 190*cos (a), 190*sin (a)
50 next
run
```

Daha küçük dört daire çizmek için şu programı yazalım.

```
new
10 CLS
20 FOR a=1 TO 360
30 ORIGIN 196,282
40 PLOT 50*COS (a), 50*SIN (a)
50 ORIGIN 442,282
60 PLOT 50*COS (a), 50*SIN (a)
70 ORIGIN 196,116
80 PLOT 50*COS (a), 50*SIN (a)
90 ORIGIN 442,116
100 PLOT 50*COS (a), 50*SIN (a)
110 NEXT
run
```

Daire çiziminde değişik bir yol görmek için şu programı yazalım.

```
new
10 MODE 1
20 ORIGIN 320,200
30 DEG
40 MOVE 0,190
50 FOR a=0 TO 360 STEP 10
60 DRAW 190*SIN (a), 190*COS (a)
70 NEXT
run
```

Bu sefer çemberin etrafında çizgi çizildi, çember PLOT ile yapıldığından daha çabuk çizildi. Bir kere daha 30. satırdaki DEG komutunu kaldırarak programı çalıştırın.

FILL (DOLDUR)

FILL komutu çizilerek belirlenmiş alanları veya ekran kenarları/grafik pencere çevrelerinin doldurulması için kullanılır. Bilgisayarı resetleyin ve şu programı yazın.

```
new
10 CLS
20 MOVE 20,20
30 DRAW 620,20
```

```
40 DRAW 310,380
50 DRAW 20,20
run
```

Ekranda bir üçgen göreceksiniz, grafik kursörünü aşağıdakini yazarak ekranın ortasına getiriniz.

```
move 320,200
```

PEN numarası 3'ü kullanarak FILL komutunu yazınız. Biz belirlenen PEN ile grafik kursörün durduğu yerden çizilmiş olan sınıra (üçgen) doğru ekranı doldurduk.

```
fill 3
```

Şimdi grafik kursörü üçgenden dışarı çıkartalım.

```
move 0,0
```

Şimdi işe aşağıdakini yazınca ne olduğuna bakalım.

```
fill 2
```

Bilgisayar ekranın kenarları ile çizilen çizgiler arasını PEN no 2'yi kullanarak doldurdu.

Programı, şu satırları ilave ederek değiştirelim.

```
50 draw 50,50
60 move 320,200
70 fill 3
run
```

buradaki boşluk kalemin mürekkebinin o tarafa sızıntı yapmasına neden olur.

Bu konuyu, önce işaretlenmiş çemberin içini doldurup sonra çizerken bir daha izleyelim.

```
new
10 CLS
20 FOR a=1 TO 360
30 ORIGIN 320,200
40 PLOT 190*COS (a), 190*SIN (a)
50 NEXT
60 MOVE-188,0
70 FILL 3
run
```

Şimdi şunu deneyiniz.

```
new
10 MODE 1
20 ORIGIN 320,200
30 DEG
40 MOVE 0,190
50 FOR d=0 TO 360 STEP 10
60 DRAW 190*SIN (d), 190*COS (d)
70 NEXT
80 MOVE -188,0
90 FILL 3
run
```

Biz doldurulmuş dairenin sınır çizgisini, pen ink'ini (kalemin mürekkebini) paper ink (sayfa mürekkebi) ile aynı yaparak görünmez hale sokarız. Örneğin;

```
45 GRAPHICS PEN: INK 2,1
run
```

GRAPHICS PEN komutu, ekrana çizilen grafiğin kalemini seçer. INK komutu da bu kalemin mürekkep rengini belirler. Yukarıdaki örnekte ise sayfa rengi ile aynı idi.

Son olarak şu gösteri programını yazalım.

```
new
10 MODE 0:BORDER 13
20 MOVE 0,200: DRAW 640,200
30 FOR x =80 TO 560 STEP 80
40 MOVE x,0:DRAW x,400
50 NEXT:MOVE -40,300
60 FOR c=0 TO 7
70 MOVER 80,0:FILL c
80 MOVER 0,-200:FILL c+8
90 MOVER 0,200:NEXT
100 GOTO 100
run
```

dolu alanların rengi, doldurulduktan sonra değişebilir. Örneğin:

```
100 SPEED INK 30,30
110 BORDER RND*26,RND*26
120 INK RND*15,RND*26,RND*26
130 FOR t=1 TO 500:NEXT: GOTO 110
run
```

İleri Detaylar

CPC 6128'de grafik için detaylı bilgileri, "Boş Vaktinizde" adlı konunun "Grafiksel Konuşma" bölümünde bulabilirsiniz.

Bu bölümü tamamlamak ve irdelemek için, sizin anlayabileceğiniz birçok komut ve terimlerden oluşan birkaç grafik gösteri programı vereceğiz. Her program devamlı şekiller çizecektir.

```
new
10 BORDER 0:GRAPHICS PEN 1
20 m = CINT (RND * 2):MODE M
30 i1 = RND * 26:i1 = RND * 26
40 IF ABS (i1-i2)  10 THEN 30
50 INK 0, i1:INK 1, i2
60 s = RND * 5 + 3:ORIGIN 320, -100
70 x = -100 TO 0 STEP S
80 MOVE 0,0:DRAW x,300:DRAW 0,600
90 MOVE 0,0:DRAW -x 300:DRAW 0,600
100 NEXT: FOR t = 1 TO 2000:NEXT:GOTO 20
run

10MODE 1:BORDER 0:PAPER 0
20 GRAPHICS PEN 2:INK 0,0:i = 14
30 EVERY 2200 GOSUB 150
40 flag = 0:CLG
50 Ink 2,14 + RND * 12
60 b% = RND * 5 + 1
70 c% = RND * 5 + 1
80 ORIGIN 320,200
90 FOR a =  0 TO 100 STEP PI/30
100 x% = 100 x COS (a)
110 MOVE x%, y%
120 DRAW 200 * COS (a/b%), 200 * SIN (a/c%)
130 IF flag = 1 THEN 40
140 NEXT
150 flag = 1:RETURN
run

10 MODE 1:BORDER 0:DEG
20 PRINT "Lütfen bekleyiniz"
30 FOR n = 1 TO 3
40 INK 0,0:INK 1,26 2,6:INK 3,18
50 IF n = 1 THEN sa = 120
60 IF n = 2 THEN sa = 135
70 IF n = 3 THEN sa = 150
80 IF n = 1 THEN ORIGIN 0,-50,0,650,0,400 ELSE ORIGIN 0,0,0,640,0,400
90 DIM c x (5), cy(5), r(5), Ic(5)
100 DIM np (5)
110 DIM px%(5,81), py%(5,81)
120 st = 1:cx(1) = 320:cy(1) = 200:r(1) = 80
130 FOR st = 1 TO 4
140 r(st + 1) = r(st)/2
```

```
150 NEXT st
160 FOR st = 1 TO 5
170 lc(st) = 0:np(st) = 0
180 np(st) = np(st) + 1
190 px%(st,np(st)) = r(st) * SIN(lc(st))
200 py%(st,np(st)) = r(st) * COS(lc(st))
210 lc(st) = 1c(st) + 360/r(st)
220 lc(st) 360 THEN 180
230 px%(st,np(st) + 1) = px%(st,1)
240 py%(st,np(st) + 1) = py%(st,1)
250 NEXT st
260 CLS:cj = REMAIN (1):cj = REMAIN (2)
270 cj = REMAIN (3):INK 1,2:st = 1
280 GOSUB 350
290 LOCATE 1,1
300 EVERY 25,1 GOSUB 510
310 EVERY 15,2 GOSUB 550
320 EVERY 5,3 GOSUB 590
330 ERASE cx,cy,r,lc,np,px%,py%:NEXT
340 GOTO 340
350 cx% = cx(st):cy% = cy(st):lc(st) = 0
360 FOR x% = 1 TO np(st)
370 MOVE cx%,cy%
380 DRAW cx% + 'x%(st,x%),cy% + py%(st,x%),1 + (st MOD 3)
390 DRAW cx% + px%(st,x% + 1),cy% + py%(st,x% + 1),1 + (st MOD 3)
400 NEXT x%
410 IF st = 5 THEN RETURN
420 lc(st) = 0
430 cx(st + 1) = cx(st) + 1.5 × r(st) * SIN(sa + lc(st))
440 cy(st + 1) = cy(st) + 1.5 × r(st) * CIS(sa + lc(st))
450 st = st + 1
460 GOSUB 350
470 st = st-1
480 lc(st) = lc(st) + 2 * sa
490 IF (ls)(st) MOD 360        0 THEN 430
500 RETURN
510 ik(1) = 1 + RND * 25
520 IF (ik(1) = (2) OR ik(1) = ik(3) THEN 510
530 INK 1,ik(1)
540 RETURN
550 ik(2) = 1 + RND * 25
560 IF ik(2) = ik(1) OR ik(2) = ik(3) THEN 550
570 INK 2, ik(2)
580 RETURN
590 ik(3) = 1 + RND * 25
600 IF ik(3) = ik(1) OR ik(3) = ik(2) THEN 590
610 INK 3, ik(3)
620 RETURN
```

Bölüm 9: Sesi Kullanmak

Ses bilgisayardaki haporlörden çıkar. Eğer MP2 modülatör/güç kaynağı ve bir televizyon kullanıyorsanız TV'nizin ses kontrolunu en aza getirin.

Ses seviyesi ayarını bilgisayarın sağ tarafındaki VOLUME kontrolünden ayarlayabilirsiniz. Ses ayrıca stereo setinizin auxiliary input soketine; bilgisayarınızın STEREO yazılı soketi kullanarak verilebilir. Bu bilgisayarınızdan çıkan sesi hi-fi hoparlör veya kulaklıktan dinleme olanağı sağlar. Bilgisayarın STEREO soketinin bağlantılarıyla ilgili bilgileri bu konunun 2 nci bölümünde bulabilirsiniz.

Ses Komutu (SOUND)

SOUND komutu 7 kısımdan (parametreden) oluşmuştur. İlk ikisi mutlaka kullanılır, diğerleri isteğe bağlıdır. Komutun tipi şöyledir:

SOUND < kanal durumu >, < ton periodu >,< süre,><ses şiddeti >, < ses şiddeti envelopu >, < ton envelopu,>, < gürültü periodu >

Biraz karışık görünmekle beraber her parametreyi incelersek çok geçmeden hepsini kavrarız. Parametreleri tek tek inceleyelim.

Kanal Durumu

Bu anda işi basit tutmak için kanal durumunu ses kanalının referans durumu diye ele alacağız. 3 ses kanalı vardır ve şimdi biz 1 nolu kanal durumunu kullanacağız.

Ton Periodu

Ton periodu sesin saniyedeki salınım sayısının teknik ifadesidir. Diğer bir değişle hangi nota olduğudur (do,re,mi,vs.). Her notanın bir ayar numasarı vardır; ve bu numara ton periodudur. "Referansınız için" adlı konuda orta (do)nun ton periodunun 239 olduğunu göreceksiniz.
Şimdi bilgisayarı ilk duruma getirin [CONTROL] [SHIFT] [ESC] ve şunu yazınız.

```
10 sound 1,239
run
```

Siz 0.2 saniye süren bir orta do notası duyacaksınız.

Her hiç bir şey duymadıysanız bilgisayarın VOLUME kontrol düğmesini kontrol ederek tekrar RUN yazınız.

Süre

Bu parametre sesin uzunluğunu ayarlar; diğer bir değişle sesin ne kadar süreceğini ayarlar. Parametre saniyenin 0.01 olan birimlerle çalışır ve parametre belirtilmediği zaman parametreyi 20 olarak kabul eder. Bizim notayı 0.2 saniye duymamızın sebebi budur. $0.01 \times 20 = 0.2$ saniye

Nota süresinin 1 saniye olması için süre 100 olmalıdır; 2 saniye olması için 200 kullanılmalıdır.

```
10 sound 1,239,200
run
```

Orta do sesini 2 saniye duyacaksınız.

Ses Şiddeti

Bu parametre notanın başlama ses şiddetini belirler. Numara Ø ile 15 arasındadır. Ø en az, 15 en fazladır. Eğer hiç bir numara kullanılmamışsa 12 kabul edilir.

```
10 sound 1,239,200,5
run
```

Bu sesin şiddetine dikkat edin. Şimdi daha büyük bir ses numarası kullanalım.

```
10 sound 1,239,200,15
run
```

Şimdi daha yüksek bir ses duyacaksınız.

Ses Şiddeti Zarfı

Notanın süresinde ses şiddetini değiştirmek için bir ses şiddeti zarfı tanımlamanız gerekir. Bu ayrı bir ENV komutu kullanarak yapılabilir. SOUND komutu gibi her birinin kendi referans numarası olan değişik sayıda ses şeddeti zarfı yapabilirsiniz. Şayet referans numarası 1 olan ses zarfı SOUND komutunda kullanmak istiyorsanız ses şiddeti zarf ile belirlenen yere 1 yazınız. Ses şiddeti zarfı yaratma daha sonra kısaca açıklanacaktır.

Ton Zarfı

Notanın çalma süresinde tonunun veya perdesinin değişmesini istiyorsanız ayrı bir ENT komutu kullanarak ton zarfı tanımlayabilirsiniz. SOUND komutunda oldu-

ğu gibi her birinin kendi referans numarası olan değişik sayıda ton zarfı vardır. Şayet referans numarası 1 olan ton zarfını SOUND komutunda kullanmak istiyorsanız ton zarfı ile belirlenen yere 1 yazınız. Ton zarfı yaratmak daha sonra kısaca açıklanacaktır.

Gürültü

SOUND komutunun son parametresi gürültüdür. Gürültünün elde edilebileceği alan gürültü parametresinin 1 ve 31 arasında değiştirilmesi ile sağlanır. SOUND komutunun sonundaki gürültü 2 yazın ve sonucu izleyin. Sonra gürültü parametresini 27 yapın ve aradaki farkı izleyin.

```
10 sound 1,239,200,15,,,2
```

Yukarıdaki örnekte iki boş parametre (,,,) gürültü parametresi 2'den önce yazılmıştır. Bunun nedeni ise bizim ses zarfı ve ton zarfını belirlemediğimizdendir.

Ses Şiddeti Zarfı Meydana Getirme

Ses şiddeti zarfı komutu ENV'dir. Basit şekilde komut 4 parametreden oluşmuştur ve şöyledir.

ENV< zarf numarası>, <basamak sayısı>, <basamak genişliği>, <basamak zamanı>

Daha önce yaptığımız gibi parametreleri tek tek inceleyelim.

Zarf Numarası

SOUND komutunda volume zarfının çağrılabilmesi için 0 ve 15 arasında belirlenmiş referans numarasıdır.

Basamak Sayısı

Bu parametre ses sürerken kaç değişik volume basamağı istediğinizi belirler. Örneğin 10 saniye süren bir notada her biri 1 saniye olan 10 volume basamağı istediniz. Bu durumda basamak sayısı parametresi 10 olarak kullanılacaktır.
Basamak sayısı 1 ile 127 arasında değişir.

Basamak Genişliği

Her basamak ses genliği bakımından esas basamağa göre 0-15 seviyesi arası değişebilir. 15 ses seviyesi SOUND emrindeki seviyelerle aynıdır. Bunun beraber basamak genliği -128 + 127 arasında ayarlanabilir. Böylece yalnız ses gücünü aşağı yukarı ayarlamakla kalmayıp volume seviyesi her 15'den sonra tekrar 0'lanır ve çok değişik sesler elde edilebilir.

Basamak Zamanı

Bu parametre basamaklar arasındaki zamanı 0.01 saniye bazına göre belirler. Basamak zamanı numaraları 0'dan 255'ye kadar değişen sıralamadır. Böylece basamaklar arasındaki en uzun zaman 2.56 saniyedir. (0 rakkamı 256 diye kabul edilir.)

Şunu unutmayınız ki basamak sayısı parametresiyle basamak zamanı çarpıldığında SOUND komutundaki süre parametresinden büyük olamaz, aksi takdirde ses bütün ses şiddeti basamaklarını geçemeden biter. Böyle bir durumda ses şiddeti zarfının geri kalan kısım atılır.

Aynı şekilde eğer süre parametresi SOUND komutundaki basamak sayısı ile basamak zamanının çarpımından büyük ise bütün ses şiddeti basamakları geçtikten sonra son seviyede sabit kalır. Ses şiddeti zarfını incelemek için aşağıdaki programı yazınız.

```
10 ENV 1,10,100
20 SOUND 1,142,100,1,1
run
```

20'nci satır 10 saniye süren ses şiddeti, 1'den başlayan ses şiddeti zarf numarası 1 olan 10 basamaklı her 1 saniyede her basamakta ses şiddeti 1 artan ton periodu 142 (la) olan bir sesi tanımlar.
10'ncu satırı değiştirerek zarf değişmesini etkisini görelim.

```
10 ENV 1,100,1,10
10 ENV 1,1002,10
10 ENV 1,100,4,10
10 ENV 1,100,20,20
10 ENV 1,50,2,20
10 ENV 1,50,15,30
```

Son olarakda sunu deneyin

```
10 ENV 1,50,2,10
```

Sesin yarı zamanında seviye aynı kaldı. Çünkü basamak sayısı 50 idi ve her basamak arasındaki zaman 0.1 saniye idi. Bu nedenle ses şiddeti sadece 5 saniye değişti. Fakat SOUND komutundaki süre parametresi 1000 olarak verilmişti. (10 saniye)

Ne tür sesler meydana getirebileceğini deneyin.

Şayet daha karışık bir ses şiddeti zarfı meydana getirmek istiyorsanız 3 parametreyi basamak sayısı , basamak genliği , basamak zamanı ENV komutu sonunda sonunda aynı zarfın değişik bölümlerini belirlemek için 4 defa daha tekrar ediniz.

Ton Zarfı Meydana Getirmek

Ton zarfı komutu ENT'dir. En basit 4 parametreden oluşmuştur.

ENT <zarf numarası>, <basamak sayısı>, <basamakların ton periodu>, <basamak zamanı>

ZARF NUMARASI

Zarf numarası belli bir zarfı SOUND komutunda belirlemek için 0 ile 15 arasında yer alan bir referans numarasıdır.

Basamak Sayısı

Bu parametre ses devam ederken kaç değişik ton basamağı istediğinizi belirtir. Örneğin 10 saniye süren bir nota, her biri 1 saniye olan 10 ton basamağı istersek (basamak sayısında 10 parametresi kullanmalıyız. Basamak sayısı 0 ile 239 arasında değişir.

Ton Periyodunun Basamakları

Bu parametrenin ton periodu -128 ile +127 arasında değişir. Negatif basamaklar notaların frekansını arttırır (tizleştirir). En kısa ton periodu 0'dır. Ton zarfını hesaplarken bu akılda tutulmalıdır. "Sizin bilginiz için" adlı bölümde ton periodlarının bütün kapsamı gösterilmiştir.

Basamak Zamanı

Bu parametre basamaklar arasındaki 0.01 saniyelik birimler seçilen zamanı belirler. Basamak zamanı numaraları 0 ile 255 arasındadır. Bu, basamaklar arası en uzun zaman 2.56 saniye demektir. (0 rakkamı 256 kabul edilir.)

Şunu unutmamak gerekir: basamak sayısı parametresi ile basamak zamanı parametresinin çarpımı SOUND komutundaki süre parametresinden büyük olması gerekir. Çünkü bütün ton basamakları geçmeden ses biter. Bu durumda ton zarfının geri kalan kısmı çıkartılır.

Aynı şekilde SOUND komutunun süre parametresi basamak sayısı ile basamak zamanı çarpımından uzun olduğu zaman ses, bütün ton basamaklarını geçip gittiği halde devam eder ve son ton frekansında sabit kalır. Ton zarfı ile deney yapmak için şu programı yazınız.

```
10 ent 1,100,2,2
20 sound 1,142,200,15,1
run
```

20'nci satırdaki ton periodu 142 (la) olan 2 saniye devam eden ses şiddeti 15'den (max) başlayan ses şiddeti zarfı olmayan (boş parametre ile gösterilen) ve ton zarf numarası olan sesi tanımlar.

10'ncu satır ton zarf numarası 1 olan 100 basamaklı her 0.02 saniyede 2 ile yükselen ton periodlu ton zarfını belirliyor.

Şimdi 10'ncu satırı değiştirerek ton zarfının değişmesini izleyin.

```
10 ent 1 1,100,-2,2
10 ent 1,10,4,20
10 ent 1,10,-4,20
```

Durdurmak için iki defa [ESC] tuşuna basınız.

Şimdi de ses şiddeti zarfını, ton zarfını ve SOUND komutunu beraberce değişik sesler elde etmek için kullanalım. Başlanıç olarak;

```
10 env 1,100,1,3
20 ent 1,100,5,3
30 sound 1,142,300,1,1,1
run
```

Sonra 20'nci satırı aşağıdakiyle değiştirelim.

```
20 ent 1,100,-2,3
run
```

Şimdi de bütün satırları değiştirelim.

```
10 env 1,100,2,2
20 ent 1,100,-2,2
30 sound 1,142,200,1,1,1
run
```

Eğer daha karışık ton zarfı yaratmak istiyorsanız, üç parametreyi basamak sayısı, basamağın ton periodu, basamak zamanı ENT komutunun sonunda 4 defa daha tekrarlatınız. Böylelikle aynı zarfın değişik bir bölümünü tanımlamış olursunuz.

AMSTRAD CPC 6128 BASIC KOMUTLARI LİSTESİ adı altındaki bölümde bütün değişik ses komutlarını bulabilirsiniz. Eğer daha melodik bir şeyler isterseniz "Müziğin sesi" adlı konuya bakınız.

Bölüm10: AMSDOS ve CP/M

AMSDOS Nedir?....

Bilgisayar açıldığı zaman ve resetlendiği zaman, sistem işlemleri AMSDOS altında yürütür. AMSDOS, AMStrad Disc Operating System (Amstrad disk operasyon sistemi) kelimelerinin kısaltılmış şeklidir ve aşağıdaki dosyalama komutları ile fonksiyonlarını sunar:

```
LOAD "dosya adı"
RUN "dosya adı"
SAVE "dosya adı"
CHAIN "dosya adı"
MERGE "dosya adı"
OPENIN "dosya adı"
OPENIN "dosya adı"
OPENOUT "dosya adı"
CLOSEIN
CLOSEOUT
CAT
EOF
INPUT # 9
LINE INPUT # 9
LIST # 9
PRINT # 9
WRITE # 9
```

Bunlara ilaveten, AMSDOS disk idaresi için ilave komutlar içerir.

Bu komutlar, harici komutlar diye adlandırılır ve önlerine çubuk (bar) sembolü (|) konur çubuk sembolünü (SHIFT) @ tuşuna basarak elde edebilirsiniz.

Bazı çok kullanılan harici komutlar şunlardır.

```
| a
| b
| tape (ikiye ayrılır. | tape.in ve | tape.out)
| disc (ikiye ayrılır. | disc.in ve | disc.out)
```

| a ve | b komutları 2-sürücü sisteminde kullanılır ve bilgisayarın hangi sürücüyü çalıştıracağını belirtir.

Örnek:

```
la
load "FILENAME"
```

Bilgisayara belirtilmiş programı sürücü A'daki disk'den yüklemesini söyler. Eğer I a ve I b'den biri verilmemişse veya bilgisayar ilk konumuna getirilmişse sistem sürücü A ile çalışır.

Eğer sadece bilgisayardaki disk sürücünü kullanıyorsanız bu sürücü A'yı kullanacağı şeklinde yorumlanır ve I a ve I b komutlarına gerek yoktur. I b komutunu ilave disk sürücüsü yokken girerseniz, ekranda şu mesajı alırsınız.

```
Drive B: disc missing (disk yok)
Retry, Ignore or Cannel (tekrar dene, unut veya iptal)
```

Yapacağınız şey C (iptal) olmalıdır.

Kaset Kullanmak İsterseniz

ltape komutu bilgisayara bütün yükleme kaydetme vs.... işlemleri disk yerine harici kaset ünitesi ile gerçekleştirmesini emreder. ltape girilmediği zaman bilgisayar daima disk operasyonunu işleme sokar.
I tape belirtildikten sonra tekrar disk operasyonuna dönmek için aşağıdaki komutu yazmak gerekir.

```
I disc
```

Değişik bir işlem olarak kasetten yüklemeyi disk de saklamayı isterseniz şu komutu kullanın.

```
I tape.in
```

Bu komut bilgisayara girilecek veriyi kasetten oku fakat kaydedilecek veriyi disk'e yaz demektir. Aynı şekilde girilecek veriyi disk'ten, saklanacak veriyi kasetde saklamak için önce I disc.in komutunu, bir önceki I tape.in komutunu geri almak için yazmak gerekir. Bundan da anlaşılacağı gibi I tape.in ve tape.out komutları I disc.in ve I disc.out komutlarını geri almak için; veya I disc.in ve I disc.out komutları I tape.in ve I tape.out komutlarını geri almak için kullanılır. Disk ve kasetle ilgili detaylı bilgileri "disk ve kaset kullanmak", "AMDSDOS ve CP/M" adlı bölümlerin 4 ve 5 nci konularında bulabilirsiniz.

Disk ve Kasete Program Kopya Etme

Bu konunun 7. bölümünde, CP/M sistem disk paketinin yüzündeki DİSCKIT3 programını kullanarak yeni boş disketi formatlamayı öğrendiniz. Aynı zamanda bir diskden diğerine bir program kopya etmeyi de öğrendiniz. Şimdi aşağıdaki komutlarla A sürücüsüne veya B sürücüsüne (kaset ünitesine) yüklemeyi kaydetmeyi yapabilirsiniz.

```
I tape, I disk, I tape.in, I tape.out, I disc.in, I disc.out, Ia,Ib
```

Diğer komutlar;

```
I dir, I drive, I era, I ren, I user
```

Bunlar, konu 5'deki "AMSDOS ve CP/M" adlı bölümde anlatılacaktır.

Disk'in Tamamını Kopya Etmek

Bir diskin içeriğinin tamamı bir diğer disk'e CP/M sistem disk paketinin 1. yüzündeki DISCKIT3 programı ile kopya edilebilir.

Sistem disk paketinizin yüzünü bilgisayarın disk sürücüsüne yerleştirin ve şunu yazın;

```
I cpm
```

A > hatırlatmasından sonra şunu yazınız.

```
disckit3
```

Birkaç saniye sonra ekranın üst tarafından DISCKIT açılış mesajını göreceksiniz. Bunu takiben;

```
one drive found
```

Bu mesaj size DISCKIT programını çalıştırdığınızı ve bilgisayarın, sizin sadece bir disk sürücüsüyle çalıştığınızı (bilgisayar üzerindeki) belirtir.
Eğer sisteme ilave birdisk sürücüsü bağladıysanız mesaj şöyledir:

```
Two drives found
```

Ekranın altında şunu göreceksiniz:

Copy	7	
Format	4	
Verify	1	
Exit from program	Ø	

Bu DISCKIT ana seçeneğidir. Numaralar tuş takımının sağ tarafındaki fonksiyon tuşlarıdır; (f0, f1, f4, f7) ve bu tuşlardan birine basarak istediğiniz seçeneği seçersiniz. Şunu unutmayınız ki Ø nolu fonksiyon tuşuna basarsanız DISCKIT programından çıkıp CP/M direkt konsol moduna geri dönersiniz. (A>)

Şimdi biz diski kopya etmek istediğimizden 7 nolu fonksiyon tuşuna basacağız.

DİKKAT

DAHA ÖNCE KAYDEDİLMİŞ DİSKİN ÜZERİNE KOPYA ETMEK BU DİSKİN ESKİ MUHTEVİYATININ HEPSİNİ SİLER.

1- Sürücülü Sistemde Kopya

Sizin 1 sürücülü sistem kullandığınızı varsayarak ekranda şu mesajı görürsünüz.

```
Y  Copy
        Any other key to exit menu
```

Bu anda ana CP/M sistem diskinizi çıkartınız ve kopya çekeceğiniz diski (şayet CP/M sistem diskini kopya edecekseniz.) sürücüye yerleştiriniz.

(Y, evet demektir ve anlamı "diski kopya etmeye başla"dır)

Disk'in formatı bilgisayarı tarafından incelenir ve ekranın üst tarafında bilginize sunulur.

Biraz sonra şu mesajı görürsünüz:

Insert disc to WRITE
Press any key to continue

Bu noktada kopya edeceğiniz diski sürücüden çıkarınız; ve sürücüye kopya yapılacak diski sokup herhangi bir tuşa basınız.

Diskin formatı hakkındaki bilgiler (üzerine kopya yapılacak diskin) ekranın üst tarafında belirtilir.

Eğer üzerine kopya yapılacak disk doğru formatlanmamışsa (veya henüz formatlanmamışsa) bu kopya yapılırken dikkate alınır ve aşağıdaki mesajı görürsünüz:

Disc isn't formatted (or faulty)
Going to format while copying
Disc will be system format

Veya buna benzer mesajları kopya ederken alabilirsiniz. Bilgisayar diskten kopya almaya tekrar hazır olduğu zaman şu mesajı görürsünüz.

Insert disc to READ
Press any key to continue

Kopya alınacak diski tekrar sokmanız gerekir.

Bu işlemi birkaç defa yaptıktan sonra 1 nci diskin muhteviyatı 2 nci diske kopya edilmiş olur ve şu mesajı alırız:

Copy completed	(kopya tamamlanmıştır)
Remove disc	(diski çıkarınız)
Press any key to continue	(devam için herhangi bir tuşa basınız)

Ekrana gelen bu talimatlar size yeni bir diski kopya etme seçeneği (Y'ye basarak) veya ana DISCKIT menüsüne dönmeyi sağlar.

Yazı Koruma

Biliyorsunuz ki yazı koruma deliği açık olan disklerin üzerine kopya yapılamaz. Eğer yapılmaya kalkılırsa:

Disc write -potected	(disk yazı korunmalıdır)
Insert discto WRITE	(YAZI için disk sokunuz)
R-etry or C-ancel	(Tekrar dene veya İptal)

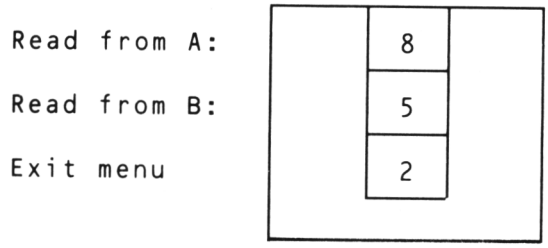
Yukarıda belirtildiği gibi iptal için C yazınız, diski çıkartınız ve yazı koruma deliği kapalı olan doğru diski kopya için koyunuz.

Saklamak istediğiniz programları ihtiva eden disklerin yazı koruma deliklerini kapatmayınız ve CP/M sistem disk paketindeki disklerinizin HİÇBİR ZAMAN yazı koruma deliklerini kapamayınız.

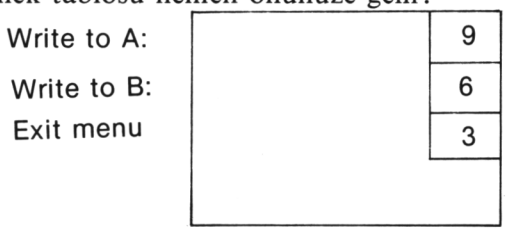
2- Sürücülü Sistemde Kopya

Yukarıdaki bilgilerin ışığında 1. yüzdeki DISCKIT programını yükleyin, sonra DISCKIT seçenek tablosundan f7 tuşuna basarak Copy seçeneğini seçiniz.

Bu anda daha başka bir seçenek tablosu elde edeceksiniz.



2- Sürücülü sistemde sürekli olarak kopyası çıkartılacak ve kopya edilecek disklerin sokma ve çıkarma işlemleri olmayacak demektir. Yukarıdaki seçenek tablosu kopyasını çıkartacağımız diskin sürücüsünü seçme imkanı tanır. Sürücü A'daki diskten kopya çekmek için f8 tuşuna basınız.
Üçüncü seçenek tablosu hemen önünüze gelir.



Bu seçenek tablosu üzerine kopya yapılacak diskin sürücüsünün seçinimini bize sunar. Siz arzu ederseniz aynı sürücüyü (kopya yapılacak diskin durduğu) seçebilirsiniz. Fakat bu sürekli olarak diskleri değiştirmeniz demektir ki 2-sürücülü sistemin avantajını kullanmamışsınız anlamına gelir.

Büyük bir kolaylık ve sürat için f6 tuşuna basınız. (sürücü B'de duran diske kopya etmek için)

Şimdi iki diski uygun sürücüye yerleştiriniz ve kopyaya başlamak için Y tuşuna basınız.

Bir kere daha her iki diskin format bilgileri ekranda görülecektir. Üzerine kopya yapılacak disk doğru olarak formatlanmışsa (veya henüz formatlanmamışsa) kopya çekilirken dikkate alınacaktır.

Kopya edildikten sonra aşağıdaki mesaj görüntüleri:

```
Copy completed
Remove both discs
Press any key to continue
```

Her iki diskide herhangi bir sonraki operasyondan önce çıkartınız. Daha sonra kopyalama işlemine Y yazarak devam edebilirsiniz; veya ana DISCKIT seçenek tablosuna herhangi bir tuşa basarak dönebilirsiniz.

Disklerin Tetkiki

DISCKIT3 programı aynı zamanda bir diskin tetkik edilme hizmetini size sunar.

Diskin formatı ile ilgili bilgiler görüntülenir; ve diskin üzerindeki bütün dosyalar okunur. Dosyalardaki herhangi bir hata ekran üzerinde tesbit edilir.

Bir diski tetkik için sistem disk paketinizin 1 nci yüzünü diske giriniz ve şunu yazınız.

```
lcpm
```

A > işleme hazır dan sonra aşağıdakini yazınız.

```
disckit3
```

Ana DISCKIT seçenek tablosunun tetkik kısmını f1 tuşuna basarak seçiniz Ekrandaki talimatları takip ediniz. Eğer iki sürücülü sistem ile çalışıyorsanız size sürücü A veya sürücü B'deki diskli tetkik için seçenek verilecektir. Tetkiki istediğiniz diski girdiğinizde Y yazarak tetkike başlatınız.

Tetkik bittiği zaman aşağıdaki mesaj görüntülenecektir:

```
Verify completed
Remove disc
Press any key to continue
```

Şayet siz 2-sürücülü sistemde çalışıyorsanız sizden her iki diski de çıkartmanız istenecektir ve herhangi bir işlem yapmadan önce bunu yapınız. Daha sonra tetkik işlemini Y yazarak tekrarlayabilir veya herhangi bir tuşa basarak ana DISCKIT seçenek tablosuna dönebilirsiniz.

DISCKIT Programını CP/M 2.2 de Kullanmak

CPC664'de veya CPC464 + DDII'de (her ikiside CP/M 2.2 kullanır) yapılmış veya yapılacak diskin format, kopya veya tetkikini isterseniz DISCKIT programı sistem disk paketinin 4 ncü yüzündedir. Programın adı DISCKIT2'dir. Aynı sebepten dolayı yukarıda bahis edilen DISCKIT3 gibi çalışır.

DISCKIT2'yi çalıştırmak için sistem disk paketinin 4. yüzünü bilgisayarınızın disk sürücüsüne sokunuz ve şunu yazınız:

```
I cpm
```

CP/M 2.2 işe başlama ve A > hazır mesajını aldıktan sonra şunu yazınız:

```
disckit2
```

Seçenek tablosundan tercihinizi yapınız ve daha önce bahis ettiğimiz DISCKIT3'deki gibi ekranda gözüken talimatları takip ediniz.

NOT: Kopya esnasında DISCKIT2 ekran hafızasını veri için kullanır. Böylece kopya yapılırken ekranda rast gele şekiller görebilirsiniz.

İlave Bilgiler

Ana CP/M sistem disk paketinizin sunduğu programların kullanılması hakkındaki ilave bilgileri 4 ve 9. konular arasında bulabilirsiniz.

Son olarak bu kitapçığın başında "ÖNEMLİ" başlığı altında verilen uyarıları bir kere daha gözden geçirin.

MONTAJ BİLGİLERİ 5,6,7
ÇALIŞTIRMA BİLGİLERİ 1,2,3,4,5,6,7,9

Bölüm 11: Bellek Bankası Yönetimi
İkinci 64K Belleği Kullanmak

CPC6128 64K'lık iki kısımdan meydana gelen 128'lık RAM (Random Access Memory) ihtiva eder. CP/M Plus bütün 128K'yı herzaman kullanır fakat BASIC normal olarak ikinci 64K'yı kullanmaz sadece birinci 64K'daki kullanılabilir belleği kullanır. BASIC ile programlama yaparken 64K'lık ekstrayı tamamen kullanmadan bırakmak çok üzücüdür. Bu nedenle program bu ekstra belleği özel olarak kullanabilecek gibi yapılmalıdır. Program 2. 64K'lık RAM'ın ya ekran görüntülerinin depolama yeri yada dizilerin depolama yeri olarak kullanılması için bağzı ekstra komutlar içerir.

Bu ekstra komutları içeren program "Bellek bankası yönetimi" diye bilinir. Burada banka teknikbir terim olup belleğin bir bölümü diye tarif edilebilir.

BELLEK BANKASI Yönetimini Ekran Görüntüleri İçin Kullanmak

6128 ekran görüntülerini her zaman ekranda yansıtır. (Bunu yapmak için her bir pixel in renk ve parlaklık bilgileri için) 16K'lık bir belleğe ihtiyaç vardır. 6128'in belleği her biri 16K'lık bloklarda 6 ekran görüntüsüne müsaittir ve bunların hepsi herhangi bir zamanda bilgisayarın belleğinde bulunabilir. BELLEK BANKASI YÖNETİMİ sizin için BASIC'den mümkün olan 6 ekran görüntüsünden 5'inin görüntüye getirmenizi; oynamanızı kolaylaştırır.

Bilgisayarı ilk açtığımızda ekran ilk 64K'da 16K blok bellek tarafından görüntülendirilir. (biz buna blok 1 deriz). Diğer 4 ekran 2'nci 64K'lık bellekte tutulur ki biz bunlara blok 2, blok 3, blok 4, ve blok 5 deriz.

Normal olarak sadece blok 1 ekranda görüntülenebilir. (ilk 64K'daki). Böylelikle 2'nci 64K'da saklananları (blok 2'den 5'e kadar) ekranda görmek için gereken ekranı blok 1'e götürmek gerekir. BELLEK BANKA YÖNETİMİ ISCREENCOPY gibi bir komut ile bir ekranı hali hazırda; görüntüdeki ekranın üzerine yazdırarak bir ekranı diğer ekrana transfer yapar, veya ISCREENSWAP komut ile iki ekranın muhteviyatlarını değiştirir.

AMSDOS komutlarında olduğu gibi BELLEK BANKA YÖNETİMİ de "harici komutlar" kullanır ve bu komutlar çubuk sembolü (|) ile başlar (SHIFT ve @ tuşuna basınca)

BELLEK BANKA Yönetiminin Kullanılması

Bilgisayarı [CONTROL] [SHIFT] [ESC] tuşlarına basarak reset ediniz ve sistem disk paketinin 1. yüzünü sokarak şunu yazınız:

```
RUN "BANKMAN"
```

Yükleme yöntemi 7. konunun 13. bölümünde RSX'de (Oturmuş sistem büyümesi) detaylı olarak anlatılmıştır. RSX hakkında bir bilginiz olması ve bu işlemi programda kullanmadan önce bellek yeri saklamanız hakkında da bir bilginiz olması tavsiye edilir. Bununla beraber aşağıdaki örnekleri denemeniz için yükleme işlemini anlamanıza gerek yoktur.

```
MODE 1

PRINT "THIS IS THE DEFAULT SCREEN"
ISCREENSWAP, 1,2
```

Metin kaybolacaktır. Şimdi siz ekran2 (blok 2'de) saklananı görüyorsunuz. Eğer makina yeni açılmışsa muhtemel olarak rastgele şekiller göreceksiniz. Bu şekilleri temizlemek için şunu yazınız:

```
MODE 1
PRINT "THIS IS SCREEN 2"
ISCREENSWAP, 1,2
```

Orijinal metin tekrar görünür. Şimdi ISCREENSWAP, 1,2 komutunu tekrarlayınız iki ekranın yer değiştirdiğini göreceksiniz. Siz 5 ekrandan herhangi muhteviyatının diğeriyle yer değiştirebilirsiniz fakat şunu aklınızda tutunuz ki siz sadece ekran 1'dekini yer değiştirirseniz görüntü değişir.

Diğer bir komuta ISCREENCOPY'dir. Bu komut size bir ekranı diğerinin üzerine yeni görüntü olarak yazar ve kopya etmeyi sağlar. Aşağıdakini yazınız.

```
MODE 1
PRINT "BU BİR EKRAN KOPYASIDIR"
ISCREENCOPY, 2,1
```

Ekran 1'in muhteviyatı ekran 2'ye kopya edildi. Şayet parametreleri aşağıdaki şekilde değiştirirsek:

```
MODE 1
ISCREENCOPY, 1,2
```

Ekranda görüntülenen içerik, ekran 2'den kopya edilen içeriktir. Birinci parametre üzerine kopya yapılacak ekrandır, ikinci ekran ise kopyası çıkartılacak ekrandır. Ekranları kopya yaparken eğer ekranların modları farklı ise veya en son mode komutundan dolayı ekran görüntüsü yuvarlanmışsa parçalanmış bir görüntü elde edersiniz. BELLEK BANKASI YÖNETİMİ'nin ekran komutları metin ekranlarından ziyade şekil ekranlarına yöneliktir, bu nedenle yuvarlanma daha az meydana gelir.

BELLEK BANKASI Yönetimi'ni Dizi Saklanmasında Kullanmak

BELLEK BANKASI YÖNETİMİ 4 adet daha komut ile 64K'lık ilave belleğin dizi değişkenlerinin dosyalama sisteminde kullanılmasını sağlar. Bir çok program iki kısma ayrılabilir. Birinci olarak normal program talimatları ve ikinci olarak da programın kullanacağı veriler. Veri temelli programa en güzel örnek adres rehberidir. Bu tür program insanların adlarını ve adreslerini saklayacak dizi alanlarını kullanır. Diğer 64K'lık bellekte birbiri peşisıra saklanabilir. Dizilerin içinde saklandığı bellek kayıtlar diye adlandırılan bölümlere ayrılabilir. BASIC'deki dizinin boyunun içeriği ile ilgili olmasına rağmen bir kayıt sadece 2 ve 255 karakter arasında sabit bir uzunluk olabilir. Kayıtların amacı standart büyüklükte bölümler sunmaktır. Bir kayıttan tekrar bulma veya saklama gibi her operasyonda otomatik olarak bir sonraki kayıda geçme takip eder, bir sonraki işlem için hazırdır. Bir sonraki operasyon için kullanılacak kayıt "geçerli kayıt" (current record) diye adlandırılır ve değişik bir kayıt tarif edilmemişse otomatik olarak kullanılır. Biz bu tip organizasyonuna "RAM disk" deriz. Bunun sebebi RAM kullanıldığı halde random access disk sistemi ile aynı usulde çalışmasıdır. Aşağıdaki tarifleri okuyunuz. Böylelikle değişik komutların her birinin ne işe yaradığını nasıl kullanıldığını anlayacaksınız, şayet anlamayacak olursanız örneklerle çalışmaya devam ediniz. ilk RAM disk komutu IBANKOPEN'dur. Bu komut her kayıtın ihtiva ettiği dizi bilgilerin karakter sayısını belirler. Yazılımı:

```
IBANKOPEN,n
```

IBANKWRITE bir diziyi şimdiki kayıtta saklar. Şimdiki kayıt, yeni yazıların arkasındaki noktaya yükseltilir ve bir sonraki operasyona hazırdır. Yazılımı şöyledir:

```
IBANKWRITE, @r%,a$
```

veya

```
IBANKWRITE, @r%,a$,n
```

r% bir tam sayı (integer) değişkenidir şöyleki operasyonla ilgili verilen bilgilere geri gönderen bir koddur. a$, kayıta yazılacak karakterleri ihtiva eden dizi değişkenidir. Birinci örnekde kayıt, şimdiki kayıttır. ikinci örnekten yazılacak kayıtı belirleyen parametredir.

IBANKREAD, şimdiki kayıtı inceler ve muhteviyatını diziye geri götürür. Şimdiki kayıt, henüz okunmuş kayıtın arkasındaki noktaya yükseltilir ve bir sonraki operasyona hazırdır. Kayıtın muhteviyatını okumak, onu değiştirmez, böylelikle kayıt tekrar tekrar okunabilir. Yazılımı ise:

IBANKWRITE, @r%,a$

veya

IBANKWRITE, @r%,a$,n

r% tamsayı değişkenidir ki operasyon ile ilgili verilen bilgilere geri döndüren bir koddur. a$ kayıt muhteviyatından okunan dizi değişkenidir. İlk örnekde kayıt şimdiki kayıttır. İkinci örnekte n parametresi hangi kayıtın okunacağını belirtir.

Son komut IBANKFIND'dır. Bu komut özel bir diziyi bulmak için kayıtları araştırır. Diziyi bulduğu kayıt nosuna geri getirir. Yazılımı ise şöyledir:

IBANKRAED, @r%,a$

veya

IBANKREAD, @r%,a$,n

veya

IBANKFIND, @r%,a$,n,m

r% bir tam sayı değişkenidir ki kayıt numarası (dizinin bulunduğu) veya kod bulunmadığını bildirdiği zaman geri döner. a$ aranan dizidir. Seçilebilir parametre n hangi kayıttan aranmaya başlandığını belirtir. Eğer n belirlenmemişse arama şimdiki kayıttan başlar. İkinci seçmeli parametre m bir kopya bulamasa dahi hangi kayıtta aramayı durduracağını belirtir. Eğer belirlenmemişse arama yazdığımız veri bitse bile bütün 64K boyunca devam eder. Bunu deneyin. Eğer sizin işletilen bir BELLEK BANKA YÖNETİM programınız varsa ekran değiş tokuş komutlarına bakınız; ve bilgisayarı resetlemeden sonra bellekte hali hazırda yerleşmiş ekstra komutlara da bakınız. Eğer bu programınız yoksa sistem disk paketinizin 1 nci yüzünü sokunuz ve şunu yazınız:

RUN "BANKMAN"

Arkasından aşağıdakini yazınız:

IBANKOPEN, 20

Bu uzunluğunu 20 karakter ayarlar ve şimdiki kayıtı Ø'a ayarlar. Şimdi de aşağıdakini yazınız.

@$ = "FIRST ENTRY" + SPACE$(9)

a$'ı 20 karakter uzunluğa ayarladı. Şimdi de şunu yazın

r% = Ø

r%'nin başlangıcını veriyor. Aşağıdakini yazınız.

IBANKWRITE, @r%,@$

Bu a$'ı 1. kayıda (kayıt Ø) yazar. Aşağıdakileri yazın.

```
d$ = SPACE$(20)
IBANKREAD, @r%,d$,Ø
PRINT d$
```

İlk komut D$'a 20'lik bir alan ayırdı. Bu kayıt okunduğunda kayıt muhteviyatını ihtiva edecek yeterli bir yerdir. İkinci komut Ø numaralı kayıdı okur ve sonucu d$'a yerleştirir. Şimdiki kayıt, kayıt 1'e atladı (daha önceki IBANKWRITE operasyonundan dolayı) ve kayıt Ø'ı okunacak kayıt olarak belirlemek gereklidir. Unutmayın ki hiçbir kayıt nosu belirtilmemişse şimdiki kayıt okunur. Son olarak, okuma operasyonunun sonucu yazılır. Bundan dolayı d$, "FRIST ENTRY"yi ve 9 boşluk ihtiva etmelidir.
Şimdi aşağıdakileri yazınız:

```
b$ = "TWO" + SPACE$(17)
c$ = "THERE" + SPACE$(15)
IBANKWRITE, @r%,b$,1
IBANKWRITE, @r%,c$
```

Bu b$ ve c$'ı kayıt 1 ve 2'ye koyar. Tekrar birinci IBANKWRITE komutundaki seçmeli parametre kullanılır ve kayıt 1'in b$'ın yerindeki kayıt olarak belirtir. Şimdiki kayıt numarası bundan sonra gelen kayıda ayarlanmış ve otomatik olarak kayıt 2'ye yerleştirilir.
Aşağıdakini yazınız:

```
PRINT r%
```

Yukarıdaki örnekten sonra r% yazmanın sonucu 2'dir. Bu rakkam son çalışan kayıt olarak veya şimdiki kayıt -1 olarak ele alınabilir. Yukarıdaki örnekde son çalışmış olan kayıt 2 idi ve sonraki kayıt 3 olacaktır.

Bu "geridönme kodu"nun amacı henüz tamamlanmış operasyon hakkında bilgi edinmektir. Başarılı bir operasyon kayıt nosunu belirten pozitif rakkamlar ile geri döner; başarısız bir işlem ise hata kodunu belirleyen bir negatif numara ile geri döner. IBANKWRITE ve IBANKREAD tarafından geri dönen iki hata vardır. Bunlar:

-1 Ulaşılan dosyanın sonunu gösterir. Bütün kayıtları kullandığı zaman veya belirtilmiş kayıt var olmadığı zaman bu durum meydana gelir.

-2 Banka anahtarlama başarısızlığı gösterir. Bu hiç bir zaman olmamalıdır.

Aşağıdaki örnekleri deneyin:

```
d$ = STRING$ (20,"X")
IBANKOPEN, 20
FOR n = 1 to 3:IBANKWRITE, @r%,d$:NEXT
```

Bu d$'ın 20×ihtiva etmesini sağlar. IBANKOPEN şimdiki kayıdın Ø olması için resetler ki IBANKWRITE komutu kayıt Ø,1 ve 2'nin muhteviyatının üzerine d$ yazmasını sağlar. Şimdi şunu yazınız.

```
a$ = "FRIST"
IBANKWRITE, @r%,a$,Ø
```

Bu "FIRST" kelimesini kayıt Ø'a bazı ×'lerin üzerine yazarak koyar. Şimdi d$'ı boşluk için resetleyin.

```
d$ = SPACE$(20)
```

Aşağıdaki komutu yazınız:

```
IBANKREAD, @r%,d$,Ø
```

Bu, d$'ın içine kayıt Ø'ı geri okur.
Bütün üç kayıt ×'leri ihtiva eder. Kayıt Ø'a "FIRST" kelimesi ilave edilmiştir. d$ boşluk ile doldurulmuştur. Son olarak kayıt Ø d$'ın içine geri okunmuştur. Şimdi aşağıdakini yazalım:

```
PRINT d$
```

Sonuç "FIRST× × × × × × × × × × × × × × × ×" olacaktır. Bu yukarıdaki komutları kullanırken önemli bir mütelayı tarif eder. Eğer bir kayıta konan dizi bu kayıtı tamamen doldurmuyorsa üzerine yazılmayan eski karakterler kayıtta kalır. Bu sebepten dolayı kayıda yeni bilgiler saklanmadan önce ya boşluk dizisi koymalı yada CHR$(32) konmalı. Bu içinde hâlâ bulunması istenmemiş karakterlerin bulunmasını ve dizinin geri okunmasını bertaraf eder. Aynı mütalaa diziye okunacak kayıt içinde yapılabilir. Eğer dizi kayıdın boyundan uzun ise dizinin sonundaki karakterler değişmeden kalır. Kayıt Ø'ın d$'a okunmadan önce d$'ın temizlenmesi (boşluk doldurulması) bu sebepten dolayıdır.

Bir diziyi bu dizinin tamamını alamayacak kadar kısa bir kayıtla yazmak da mümkündür. Bu durumda artan karakterler ihmal edilir.

Aynı şekilde uzun bir kayıdın içeriğini kısa bir diziye okumak da mümkündür. Yine artan karakterler (kayıttan okunan) ihmal edilecektir. Normal dizi işlemlerinde BASIC bütün karakterleri kabul etmek için harici komutlar kullanmak gerekir.

Son olarak IBANKFIND komutu. Bu komut kısmı bir girdi için kayıdın aranmasına izin verir. Örneğin eğer kayıt numarası 24 "FRED" kelimesi ile başlıyorsa bu aşağıdaki komut ile bulunabilir:

```
IBANKFIND,@r%,"FRED"
```

Bu işlem adres ve isim aranması gerektiren veri bazlı programlarda çok faydalı olur.

IBANKFIND şimdiki kayıttan başlar ve araştırılan dizi bulununcaya kadar kayıt da çalışmaya devam eder; veya kayıtların saklandığı ikinci 64K'nın sonuna kadar çalışır.

Komutun sonuna kayıt numarası ilave ederek nereden başlayacağını belirler.

Başlangıç kayıt numaralarından sonra konulan kayıt numarası ilave edilebilir; ve bu aramanın nerede sona ereceğini belirler.

IBANKFIND kayıdın başlangıcına yerleşmemiş diziyi aramakta da kullanılır. Bunun için CHR$(Ø) aranacak kanalı karakterden önceki karakter pozisyonunda kullanılır. CHR$(Ø) "wildcards" gibi ele alınır. (CP/M dosya isimlerindeki ? işaretleri gibi) anlamı herhangi bir karakter gibi sayılmalarıdır. Bir örnek olarak aşağıdakini verebiliriz:

```
a$ = STRING$(1Ø,Ø) + "FRED"
IBANKFIND, @r%,a$Ø
```

Bu "FRED" kelimesinin ilk gözüktüğü yeri bulur (11'nci de 14 ncü karakter pozisyonunda durmaktadır.) İlk 10 karakter telefon numaralarını veya diğer bilgileri ihtiva eder ki IBANKFIND bunu değerlendirmez.

Dizinin bulunduğu tam sayı değişkeni r%'ye yerleştirilmiş kayıt numarası geri dönüş kodu olarak kullanılır. (eğer "FRED" kelimesi bulunmuşsa) aksi takdirde geri dönüş kodu -3 ihtiva eder.

İlave Detaylar

BELLEK BANKASI YÖNETİMİ hakkında daha geniş bilgileri 8'nci konuda bulabilirsiniz ve 7'nci konunun 13'ncü ve 14'ncü bölümlerinde RSX'in kullanımı daha detaylı olarak anlatılmıştır.

ÇALIŞTIRMA BİLGİLERİ 1,2,3,9

CPC 6128'in 11 bölümlük teme kursunu bitirmiş bulunuyorsunuz. Şimdi siz tuştakımındaki birçok tuşun ne işe yaradığını, basit BASIC KOMUTLARININ NASIL kullanılacağını, yeni diskin kullanmaya hazırlanması için nasıl formatlanacağını bazı temel disk fonksiyonların gerçekleştirileceğini (örneğin LOAD, SAVE, CAT, vs.) biliyorsunuz.

Bundan sonraki konu AMSTRAD BASIC ve hesaplamalara daha özel bir bakış açısı kazandıracaktır.

AMSDOS ve CP/M bölümlerinden ünitenin disk sürücü bölümleri için derin bir başvuru bölümü hazırlanmıştır ve Digital araştırmadan Dr. LOGO adlı yeni bir bilgisayar lisanıyla tanışacaksınız.

KONU 2
Temel Kursun İlerisinde

Siz temel kursu okudunuz ve bilgisayarınız açık olarak önünüzde duruyor. FOR NEXT çevriminin nasıl yapıldığını ve IF THEN kullanarak şartları incelemeyi ve doğru ise herhangi bir şeyi yapmayı öğrendiniz.

Siz, bütün ekranda adınızı görmekten sıkılmış olabilirsiniz ve faydalı veya eğlendirici olan ciddi hesaplamalar yapmak isteyebilirsiniz. Bundan sonra AMSTRAD BASIC komutlarının listesini sizin idareniz için yazılmış tarifleriyle verecek ve hangi komutun ne için kullanıldığını tarif edeceğiz. Bu listeyle kuşanmış olarak bilgisayara yaptırabileceğiniz şeyler sadece sizin tasavvurunuzla sınırlanmıştır.

Eğer daha önce bilgisayar kullanmamışsanız, programcılık sizi ürkütebilir. Fakat bu sizin düşündüğünüzden daha kolaydır. Şimdi BASIC'in yeni bir lisan olmadığını fakat bazı şeylerin hızını yükseltmek için İNGİLİZCE'deki bazı sözcüklerin kısaltılmışı olduğunu düşünün. Diğer bir deyişle CLS'nin 3 harflik sihirli bir kod olmayıp sadece Clear Screan (ekranı temizle) olduğunu düşünün.

Nereden Başlayalım.

Yeni başlayanlar, için programa başlamak en zor kısımdır. Yapacağınız ilk şey programın ne yapmasını istediğiniz ve sonuçların size nasıl verilmesini istediğinize karar vermenizdir.

Buna karar verdikten sonra istediğiniz şartları yerine getiren programı yazmaya başlayabilirsiniz. Daima, programda enaz sayıda GOTO kullanarak, programın baştan sona pürüzsüz bir şekilde akmasını nasıl sağlayacağınızı düşünün. İyi bir program liste edildiği zaman kolaylıkla takip edilebilen ve hatadan ayıklarken (debug) veya hata bulurken sizi ümitsiz bir karışıklığa itmeyen programdır.

BASIC çok hoşgörülü bir dildir ve çoğunlukla size yanlışlık yaptığınızda ekranda hata mesajı çıkararak yardım eder. BASIC sizin sonradan düşündüklerinize de ilave imkanı tanır ve yeni satırları hali hazırda bulunan satırların arasına sokabilirsiniz.

Basit Bir Program Yazımı

Şimdi adları ve telefon numaralarını tutacak bir program yazalım. Programı "TELEFON REHBERİ" diye adlandıralım. Şimdi yukarıdaki kaideleri uygulayalım: "Program ne yapacak?" ve "Sonuçları bize nasıl sunulacak?"

Program 100 isim ve telefon numarası girmenize ve saklamanıza müsait olacaktır. Telefon numarasına ihtiyacınız olduğunda adı yazmakla numarayı geri alabileceksiniz. Buna ilave olarak isimlerden birin nasıl girdiğinizden emin olmadığınız zaman bütün bilgilerin ekranda görünmesini sağlayacağız. Dikkat ederseniz otomatik olarak sonuçların nasıl sunulacağı sorusuna da cevap vermiş olduk.

Başlangıç olarak programın ismini yazalım.

```
1Ø REM telefon rehberi
```

Daha sonra adları, (karakter dizisi) değişken adı NAME$ olan değişkenlere gireceğiz. (INPUT) Aynı şekide telefon numaralarını TEL$ adlı değişkenlere gireceğiz.

```
2Ø INPUT "ismi giriniz"; NAME$
3Ø "telefon numarasını giriniz"; TEL$
```

örneğin; Ahmet adını ve telefon numarasını da 1602948 diye girdiniz.

Program bilgileri sakladı fakat ekranda herhangi bir değişiklik olmadı. Bu bilgileri bulup getirecek ve yazacak NAME$ ve TEL$ değerlerini elde etmek için aşağıdaki komutu kullanınız.

```
PRINT NAME$..... ve ..... PRINT TEL$
```

Biz 100'er adet isim ve telefon numarasını programda saklamak istiyorduk. Tabii ki programa 100 adet INPUT komutu yazmayacağız. Bilgisayar bu durum için bir dizi tedarik eder ve bu dizi herhangi bir boyut numarası olan bir değişkeni (NAME$) kullanmamıza müsaade eder. Değişkenin içeriğini istediğiniz zaman, değişkenin adını takip eden parantez içinde bir referans numarası kullanırsınız. Bu referans numarasının adı (subscript) "altyazı"dır ve NAME$(27) ifadesi altyazılı değişkendir. Eğer biz sayı değişkenini, altyazı olarak kullanırsak (örneğin: NAME$(×)) bir FOR NEXT çevrimi ile 1'den 100'e kadar, × değerini değiştirerek bütün değişkenlerin listesini yazdırabiliriz. × değeri 1 artınca altyazı değeride değişecek ve NAME$ dizisinde değişik bir isim olacaktır.

İki dizi istediğimizden (NAME$ ve TEL$) her birinin boyutu 100 eleman olacaktır. Diziyi kullanmadan önce DIM komutunu koymanız gerekir.

```
2Ø DIM NAME$ (1ØØ)
3Ø DIM TEL$ (1ØØ)
```

Değişkenlerimizi belirledikten sonra öncelikle adları ve numaraları bu diziye koyacak bir program yazalım.

```
40 FOR x = 1 TO 100
50 INPUT; "Adı Soyadı";NAME$(x)
60 INPUT "Telefon numarası";TEL$(x)
70 NEXT
run
```

Bütün bunlar yapıldı fakat biz tüm isimleri bir seferde girmek istemeyebiliriz. Bundan başka programın ekrandaki görüntüsü göze pek hoş gelmiyor, ekrandaki fazlalıklardan kurtulmalıyız. Bunun için şu satırı ekleyin.

```
45 CLS
```

Şimdi bilgisayarın bizim giriş bilgilerimizi bitirip bitirmediğimizi anında anlamasını nasıl sağlayacağız? [ESC] tuşuna basarak programı durdurabiliriz fakat RUN yazdığımızda verdiğimiz değişkenlerin tamamı kaybolur.
Yukarıda bahis ettiğimiz yöntem yerine, program yeni isimleri girdi olarak alırken biz herhangi birşeyin, NAME$(x)in değeri yerine yazılıp yazılmadığını kontrol eden bir program yazabiliriz. Diğer tabiriyle NAME$(x) değerinin boş dizi olup olmadığını kontrol edeceğiz. Bunu nasıl yapacağımız hakkında bir ipucu yakaladınız mı? Şimdi aşağıdakileri yazınız.

```
55 IF NAME$(x) = "" THEN 80
80 PRINT "Başka girdi yok"
```

Program size girişleri nasıl durdurabileceğinizi de söyleyebilir. Şöyleki;

```
47 PRINT "girişleri durdurmak için (RETURN) basın)"
```

Şimdi sakladığınız bilgileri "PRINT" edecek bir programa geçelim.

```
 90 FOR x = 1 TO 100
100 PRINT NAME$(x);TEL$(x)
110 NEXT
```

Yine program 100. elemana gelmeden nerede duracağını bilmiyor, şunu ekleyelim:

```
 95 IF NAME$(x) = "" THEN 120
120 PRINT "Liste bitti"
```

95. satır NAME$(x) in boş dizi olup olmadığını kontrol eder ve IF THEN komutuyla 100 ve 110. satırı atlayarak yazmaya son verir.

Bundan sonraki ihtiyacımız, girilmiş herhangi bir ismi arayacak olan programdır. Aşağıdakini yazınız.

```
130 INPUT "bul";SEARCH$
140 FOR x = 1 TO 100
150 IF INSTR (NAME$) (x),SEARCH$) = Ø THEN 180
160 PRINT NAME$(x);" ";TEL$(x)
170 END
180 NEXT
190 PRINT "ad bulunamadı"
run
```

150. satırdaki INSTR komutu bilgisayara 2. dizi ifadesinin ilk görüldüğünü bulmak için 1. dizi ifadesini sorguya çek der. Diğer bir deyişle SEARCH$ için NAME$'ı araştırır. (SEARCH$ değişkeni 130. satırdadır ve aranan adı içerir) Eğer INSTR aradığını bulamazsa Ø değerini verir ve programda 100. satıra geçer, x'in bir sonraki değeriyle bir daha dener. Program x'in 100'e kadar olan değerlerinin hepsinden geçmişse 190. satıra devam eder. INSTR, Ø değerini meydana getirmezse program 150. satıra geçer buradan 160'a adı ve telefon numarasını yazar, 170. satırda son bulur.

Gördüğünüz gibi programımız oldukça hızlı gelişiyor fakat daha yapılacak çok iş var. Yaptığımız işleri bir daha hatırlayalım. Önce bilgiyi veriyoruz, sonra listeyi geri alıyoruz ve belirttiğimiz bir ismi arıyoruz.

Eğer bunu bir sırada yapmak istemiyorsak, eskiden kaydedilmiş bir ismi arayacak olursak, listeye başka isimler ilave edeceksek ne olacak?. Bütün bu görüşleri düşünmeli ve çözümlerini bulmalısınız.

Bu programdaki diğer bir problem de bilgisayarın hafızasında saklanan verilerin tekrar RUN yapılmasıyla silinmesidir. Bu da istenmeyen bir durumdur.

Çözüm

Programı ne şekilde yazmak gerekir ki çalıştığı zaman değişik fonksiyonlardan seçim yapılınca bu seçimi çalıştırabilsin. Bu tip programlara "seçenek tablosu sürücüsü" (menü) denir ve seçim için ekranda gösterilir. Şimdi yukarıdaki programa seçenek tablosu ekleyelim.

```
 32 PRINT "1. bilgiyi gir"
 33 PRINT "2. bilgi listesi"
 34 PRINT "3. arama"
 35 PRINT "4. bilgi saklanması"
 36 PRINT "5. bilgiyi yükle"
 37 INPUT "seçeneğinizi giriniz";ms
```

```
 38 ON ms GOSUB 40,90,130
 85 RETURN
125 RETURN
170 RETURN
200 RETURN
```

Gördüğünüz gibi programa seçenek tablosu eklendi. Bu durumda ms değişkenine seçiminizi girebilirsiniz. 38. satırda ON ms GOSUB komutu programa eğer ms = 1 ise ilk alt program satır numarasına (40.satır), ms = 2 ise 90 satır numaralı alt programına gitmesini söyler ve böylece bu iş ms'in alacağı değerlere göre devam eder.

Şimdi her fonksiyonun bulunduğu alt program ON ms GOSUB ile çağrıldığından hepsine birer RETURN komutu ilave etmemiz gerekir.

RETURN komutunun ne işe yaradığını hatırlıyormusunuz? Alt programdan sonra hemen GOSUB komutunu izleyen satıra geri döner. Yukarıdaki örnekte 38. satırdan sonra gelen 40. satıra geri döner ve proram buradan çalışmaya devam eder. Biz bunu istemediğimiz için şu satırı programa ilave edeceğiz.

```
39 GOTO 32
```

Bu durumda program geriye döner ve ekrana menüyü çıkartır. Şimdi programı tekrar çalıştıralım ve ne kadar ilerlediğimizi görelim.
Daha önce programın listesini alalım ve program hâlâ çalışıyorsa [ESC] tuşuna iki defa basarak durduralım. Program durunca şunu yazınız.

```
LIST
```

```
 10 REM telefon rehberi
 20 DIM NAME$(100)
 30 DIM TEL$(100)
 32 PRINT "1. Bilgiyi gir"
 33 PRINT "2. Bilgi listesi"
 34 PRINT "3. Arama"
 35 PRINT "4. Bilgi saklanması"
 36 PRINT "5. Bilgiyi yükle"
 37 INPUT "seçenek tercihinizi girin";ms
 38 ON ms GOSUB 40,90,130
 39 GOTO 32
 40 FOR x = 1 TO 100
 45 CLS
 47 PRINT "girişleri durdurmak için RETURN'e basın"
 50 INPUT; "Adı Soyadı"; NAME(x)
```

```
 55 IF NAME$(×)="" THEN 8Ø
 6Ø INPUT "Telefon numarası";TEL$(×)
 7Ø NEXT
 8Ø PRINT "Başka girdi yok"
 85 RETURN
 9Ø FOR × =1 TO 1ØØ
 95 IF NAME$(×)="" THEN 12Ø
1ØØ PRINT NAME$(×);" ";TEL$(×)
11Ø NEXT
12Ø PRINT "Liste bitti"
125 RETURN
13Ø INPUT "bul";SEARCH$
14Ø FOR × =1 TO 1ØØ
15Ø IF INSTR (NAME$(×), SEARCH$)=Ø THEN 18Ø
16Ø PRINT NAME$(×);" ";TEL$(×)
17Ø RETURN
18Ø NEXT
19Ø PRINT "ad bulunamadı"
2ØØ RETURN
```

Programın bazı yerlerine yeni komutlar sıkıştırdık. RENUM komutu ile satırları tekrar numaralandıralım ve sıkışık kısımları açarak yeniden düzenleyelim. Aşağıdaki komutu yazınız.

```
RENUM
LIST
```

Bilgisayar listeyi şöyle verecektir.

```
 1Ø REM telefon rehberi
 2Ø DIM NAME$(1ØØ)
 3Ø DIM TEL$(1ØØ)
 4Ø PRINT "1. Bilgiyi gir"
 5Ø PRINT "2. Bilgi listesi"
 6Ø PRINT "3. Arama"
 7Ø PRINT "4. Bilgi saklanması"
 8Ø PRINT "5. bilgiyi yükle"
 9Ø INPUT "seçenek tercihizi girin";ms
1ØØ ON ms GOSUB 12Ø,21Ø,27Ø
11Ø GOTO 4Ø
12Ø FOR × =1 TO 1ØØ
13Ø CLS
14Ø PRINT "girişleri durdurmak için RETURN'e basın"
15Ø INPUT "Adı soyadı";NAME$(×)
16Ø IF NAME$(×)="" THEN 19Ø
```

```
170 INPUT "Telefon numarası";TEL$(x)
180 NEXT
190 PRINT "Başka girdi yok"
200 RETURN
210 FOR x = 1 TO 100
220 IF NAME$(x) = "" THEN 250
230 PRINT NAME$(x):" ";TEL$(x)
240 NEXT
250 PRINT "Liste bitti"
260 RETURN
270 INPUT "bul"; SEARCH$
280 FOR x = 1 TO 100
290 IF INSTR (NAME$(x),SEARCH$) = 0 THEN 320
300 PRINT NAME$(x);" ";TEL$(x)
310 RETURN
320 NEXT
330 PRINT "ad bulunamadı"
340 RETURN
```

Program şimdi daha iyi bir şekle girmiştir fakat bir komut ilave etmemiz gerekmektedir. Bu komut herhangi bir yeni isim ve telefon girdiğimizde, bilgisayara bunu boş bulduğu dizinin ilk elementi olarak yerleştirip hali hazırda bulunan kayıta ilave ettirecektir. Bunun için LEN komutunu (dizinin boyunu bildir) kullanacağız.

Eğer ad değişkenlerinin uzunluğu sıfırdan büyükse, yani eleman dizisinde bir kayıt varsa 180. satıra gitmesi lazımdır.

```
135 IF LEN (NAME$(x))>0 THEN 180
```

Daima, programlama ihtiyaçlarınıza yardımcı olacak enaz bir komut vardır ve ne kadar çok program yaparsanız çözümler o kadar kolaylaşacaktır.

Bu sefer program çalıştırıldığı zaman değişkenlerin muhteviyatının geri yüklenebilmesi problemi ortaya çıkmıştır. Temel kursun 7. bölümünde SAVE komutunu kullanarak programı nasıl saklayacağımızı görmüştük. Programda değişken değerleri (tuş takımından) girilir ve ekrandan geri alınır. Yani programı SAVE komutu ile sakladığınız zaman girdiğiniz değişken değerleri saklanmaz. Bundan dolayı öyle bir program bölümü yazmalıyız ki değişken değerlerini diskin üzekinde saklayabilsin. Bunuda ayrı bir "veri dosyası" yaratarak sağlarız.

Önce bir çıkış (OUTPUT) dosyası açarız (OPEN) ve bir isimle (örneğin "data") belirleriz. Daha sonra değişkenlerin (NAME$(x) ve TEL$(x)) değerlerini 1'den 100'e kadar dosyaya yazarız (WRITE) ve son olarak çıkış (OUTPUT) dosyasını kapatıp (CLOSE) seçenek tablosuna geri (RETURN) döneriz. Bu programı 350.

satırdan itibaren ilave edelim. Her yeni satır numarasını yazmamak için aşağıdaki komutu kullanılım.

```
AUTO 350
```

Bu komut satırları otomatik olarak 350'den itibaren numaralandırır.

```
350 OPENOUT "data"
360 FOR x = 1 TO 100
370 WRITE # 9, NAME$(x), TEL$(x)
380 NEXT
390 CLOSEOUT
400 PRINT "data saklandı"
410 RETURN
```

410. satırı yazdıktan sonra [ESC] tuşuna iki defa basıp otomatik numuralandırmayı durdurunuz.
Seçenek tablosuna birkaç tercih ilave ettiğimiz için 100. satırdaki ON ms GOSUB komutuna ilgili satır numaralarını eklemeliyiz.

```
100 ON ms GOSUB 120,210,270,350
```

Seçenek tablosundan 4. maddeyi seçtiğiniz zaman bilgisayara girdiğiniz tüm bilgiler saklanacaktır.

370. satırda WRITE komutundan sonra # 9 ifadesi kullanıldı. Bu ifade bilgisayara hangi akıntının "data"yı götüreceğini belirtir. Bilgisayarın 10 akıntısı vardır. Verileri 0 ve 7 akıntılarına yöneltmenin sonucu, verilerin ekrana çıkartılmasıdır. Çünkü # 0 ve # 7'ye kadar olan ekran akıntısı WINDOW'dur.

Verileri #8 akıntısına gönderirseniz, printeriniz çalışır.

Son olarak verileri #9 akıntısına gönderirseniz, 370. satırda olduğu gibi bilgiler disk sürücüsüne gider.

Eğer satır numaralarını hariç bırakacaksak aşağıdakini yazınız.

```
AUTO
```

Bilgisayar her [RETURN] tuşuna basıldığında 10'dan başlayarak ve 10'ar arttırarak satırları numaralar. Daha önce 10,20,30 vs. satırlarını kullandıysanız [RETURN] tuşuna her basışınızda bu satırlar ekranda görülür. Her satır ekranda görününce, satırlarda çabuk ve devamlı düzeltme yapmak için [RETURN] tuşuna basılmadan bu işlem yapılmalıdır.

Programa Geri Dönüş

Daha evvel bilgileri diskde saklamayı yapmıştık. Şimdi ise yapacağımız verilerin diskden geri yüklenmesidir. Bu nedenle 100. satırdaki komuta yeni bir ilave yapacağız. Bunun için 100. satırı şöyle düzeltelim.

```
100 ON ms GOSUB 120,210,270,350,420
```

Önce diskden "data" ismi ile bir giriş dosyası açalım. Daha sonra tüm değişken değerlerini ihtiva eden girişleri #9 akıntısı ile alalım, son olarak giriş dosyalarını kapatalım ve menüye geri dönelim.

```
420 OPENIN "data"
430 FOR x = 1 TO 100
440 INPUT # 9, NAME$(x),TEL$(x)
450 NEXT
460 CLOSEIN
470 PRINT "data yüklendi"
480 RETURN
```

Başlangıcın Sonu

Böylelikle biz gerekli bütün şartlara haiz ve ne yapması gerektiğini seçebileceğimiz bir program yazdık. Bütün geri kalan ise neticeleri ekrana çıkartmaktır.

Sonun Başlangıcı

Programın görüntüsünü düzenlemek için bazı eklemeler yapalım.

```
34 MODE 1
```

Bu ekran modunu sağlar ve programın başında ekranı temizler. Şimdi de şu satırı girelim.

```
36 WINDOW # 1,13,30,10,14
```

Burada yapılan, menüyü içine yerleştirmek için ekranda küçük bir pencere açmaktır. #1, pencerenin akıntı numarasını belirler. Akıntı numarası belirtilmediği takdirde ekrandaki tüm görüntüler #0 ekran akıntısı kullanılarak yapılmıştır. #1 ifadesini dört adet rakam takip etmektedir. Bunlar bilgisayara sırası ile sol, sağ, üst ve alt kenar olarak pencere boyutlarını bildirir. Bu komutlar aynı LOCATE komutunda olduğu gibi satır ve kolon numaralarıdır. Örnekte; pencerenin sol kenarı 13. kolondan başlıyor, sağ kenarı 30. kolonda sona eriyor, üst kenar 10. sıradan başlıyor ve 14. satırda sona eriyor.

Şimdi bizim seçenek tablosunu #1 Akıntısını yazalım. Bunun için 40,50,60,70 ve 80. satırları değiştirelim.

```
4Ø PRINT #1, "1.Bilgiyi gir"
5Ø PRINT #1, "2. Bilgi listesi"
6Ø PRINT #1, "3. Arama
7Ø PRINT #1, "4. Bilgi saklanması"
8Ø PRINT #1, "5. Bilgiyi yükle"
```

Bir de şu satırı ekleyelim.

```
85 LOCATE 7,25
```

Bu seçenek tablosu giriş cümlesini 90. satıra yerleştirir.

Seçenek tablosuna geri dönüldüğünde ekranın temizlenmesi için aşağıdaki ilaveyi yazalım.

```
11Ø GOTO 34
```

Seçenek tablosundan tercih yapıldığında ekranın temizlenmesi için şu satırı ekleyelim.

```
95 CLS
```

Son olarak aşağıdaki üç satırı seçenek tablosuna geri dönmeden programı beklemeye almak için ilave edelim.

```
1Ø3 LOCATE 9,25
1Ø5 PRINT "menü için herhangi bir tuşa basın"
1Ø7 IF INKEY$ = "" THEN 1Ø7
```

103. satır bilgisayara 105. satırdaki mesajı nereye yazacağını belirtir. 107. satır tuş takımına basılıp basılmadığını kontrol eder. Bu yöntem programa ara vermek için kullanışlı bir yoldur. Bu sayede BASIC bir tuşa basılıncaya kadar bir sonraki satıra geçmez.

Programımız tamamen bitmiş durumdadır. Siz programı daha iyi bir hale sokabilirsiniz. Adları ve telefon numaralarını silebilir, alfabetik sıraya sokup yazıcıdan liste alabilir, bilgisayarın otomatik olarak telefon numarasını çevirmesini sağlayan bir program yazabilirsiniz. Programı burada bitiriyoruz. Son olarak satırların düzenlenmesi için şu komutu yazınız.

```
RENUM
```

Program aşağıdaki hali almıştır.

FİNAL LİSTESİ

```
10 REM Telefon rehberi
20 DIM NAME$(100)
30 DIM (TEL$(100)
40 MODE 1
50 WINDOW # 1,13,30,10,14
60 PRINT # 1, "1. Bilgiyi gir"
70 PRINT # 1, 2. Bilgi listesi"
80 PRINT # 1, "3. Arama"
90 PRINT # 1, "4. Bilgi saklanması"
100 PRINT # 1, "5. Bilgiyi yükle"
110 LOCATE 7,25
120 INPUT "seçenek tercihinizi girin";ms
130 CLS
140 ON ms GOSUB 190,290,350,430,500
150 LOCATE 9,25
160 PRINT "menü için herhangi bir tuşa basın"
170 IF INKEY$ = "" THEN 170
180 GOTO 40
190 FOR x = 1 TO 100
200 CLS
210 IF LEN(NAME$(x)) = 0 THEN 260
220 PRINT "girişleri durdurmak için RETURN'e basın
230 INPUT; "Adı soyadı";NAME$(x)
260 NEXT
270 PRINT "Başka girdi yok"
280 RETURN
290 FOR x = 1 TO 100
300 IF NAME$(x) = "" THEN 330
310 PRINT NAME$(x);" ";TEL$(x)
320 next
330 PRINT "Liste bitti"
340 RETURN
350 INPUT "bul";SEARCH$
360 FOR x = 1 TO 100
370 IF INSTR(NAME$),SEARCH$( = 0 THEN 400
380 PRINT NAME$(x);" ";TEL$(x)
390 RETURN
400 NEXT
410 PRINT "ad bulunamadı
420 RETURN
430 OPENOUT "data"
440 FOR x = 1 TO 100
```

```
450 WRITE # 9, NAME$(x),TEL$(x)
460 NEXT
470 CLOSEOUT
480 PRINT "data saklandı"
490 RETURN
500 OPENIN "data"
510 FOR x = 1 TO 100
520 INPUT # 9, NAME$(x),TEL$(x)
530 NEXT
540 CLOSEIN
550 PRINT "data yüklendi"
560 RETURN
run
```

Konu 3
AMSTRAD CPC 6128 BASIC Komutlarının Komple Listesi

ÖNEMLİ

Bu konuda kullandığımız terimleri ve notasyonları anlamanız çok önemlidir. Birkaç çeşit parantez göreceksiniz ki bir komutu izah ederken kullanılmıştır. Bu tip parantezlerin her birinin ayrı anlamı vardır,

Herhangi bir komut parantezli olarak gösterilmiyorsa aynen kullanılmalıdır. Örneğin;
END

komutu gibi.

Eğer bir yazı köşeli parantez içindeyse, örneğin;

< satır numarası >

kelimeleri ve satırları yazmak gerekmez. Yukarıdaki örnekte komut için gerekli tip veri gösterilmiştir. Örnek

EDIT < satır numarası >

gördüğümüzde

```
EDIT 100
```

yazılmalıdır.

Yuvarlak parantez mutlaka yazılmalıdır. Örneğin;

COS (<rakamsal ifade>)

parantezin içine cosinüsün gerektirdiği açı değeri yazılır, mesela;

```
PRINT COS (45)
```

Köşeli parantez ise isteğe bağlı ifadeleri belirler.

RUN [<satır numaras>]

RUN komutunu bir parametrenin takip etmesi gerekmez, fakat siz bir parametre koyarak bu komutu genişletebilirsiniz. Komut aşağıdaki gibi iki şekilde de ifade edilebilir.

RUN .... veya .... RUN 100

Özel Karakterler

& veya &H Hexadecimal sabite işareti
&× Binary sabite işareti
Kanal yönlendirici işareti

Veri Tipleri

Diziler Ø'dan 255 karakter uzunluğunda olabilir. Bir dizi ifadesi dizi çeşidinin değerini meydana çıkaran bir ifadedir. Diziler + operatörünü kullanarak birbirlerine ilave edilebilirler. Sonuç olarak çıkan dizi 255 karakterden uzun olamaz.

Nümerik veri reel veya tam sayı olabilir. Tam sayı verisi -372768 ile 32767 arasında olabilir. Reel sayılar ise ± 1.7E + 38 ve sıfırın üstündeki en küçük değer 2.9E-39 dur.

Bir nümerik ifade sonucu nümerik olan herhangi bir ifade olabilir. Sadece rakamlar oluşabilir veya nümerik değişken olabilir yada değişkenler tarafından çalıştırılan rakamlar olabilir.

Bir akıntı ifadesi nümerik ifade ile ilgilidir. Şöyleki, konunun nereye gönderilmesi gerekiyorsa (ekran penceresi, yazıcı veya disk gibi) bunu belirler.

Bir konu listesi bir virgülle ayrılan konu listesini belirleyen parametredir. Liste bir yada daha fazla konudan olabilir.

Bazı işaretler:

% Tam sayı
! reel sayı (gerçek)
$ Dizi

AMSTRAD CPC6128 BASIC komutları burada şu formu kullanarak sıralanmıştır.

Anahtar Kelime

Söz dizimi
Örnek
Tanımlama
Bağlantılı komutlar
Anahtar kelimeler:

KOMUTLAR : doğrudan yapılan işlemler
FONKSİYONLAR: bir ifadede girdi olarak istenen işlemler
OPERATÖR matematiksel girdilerdir.

BASIC küçük harf girilen tüm komutları program LIST yapılırken büyük harfe çevirir. Bu konudaki örnekler büyük harf olarak verilmiştir. bu program LIST yapıldığı zamanki görüntüsüdür. Eğer siz küçük harfler ile girerseniz hatalarınızı (yanlış yazılmış komutları) LİST yapıldığında küçük harf olarak kalacağıdan kolaylıkla fark edersiniz.

Anahtar Kelimeler

ABS

ABS (nümerik ifade)

```
PRINT ABS (-67.98)
67.98
```

FONKSİYON: Verilen ifadenin mutlak değerini geri verir. Bu, negatif sayıların pozitife çevrilmesi anlamındadır.

Bağlantılı komut: SGN

AFTER

AFTER < zamanlayıcı gecikmesi >, [< zamanlayıcı numarası >] GOSUB < satır numarası >

```
10 AFTER 250 GOSUB 60:CLS
20 PRINT "5 saniye içinde bir harf tahmin edin"
30 a$ = INKEY$:IF flag = 1 THEN END
40 IF a$ >< CHR$(INT(RND * 26 * 97)) THEN 30
50 PRINT a$; "doğru. Siz kazandınız"
60 PRINT "çok geç. Ben kazandım"
70 SOUND 1,2000:flag = 1:RETURN
run
```

KOMUT: Verilen bir zaman dönemi geçtikten sonra bir alt programı çağırır. zamanlayıcı gecikmesi parametresi 0.02 saniye birimleriyle gecikme süresini belirtir. Zamanlayıcı numarası (0 ile 3 arasında) eldeki dört geciktirme zamanlamasından hangisinin kullanılması gerektiğini belirtir. Zamanlayıcı 3 en yüksek önceliklidir. Zamanlayıcı 0 (önceden tespit edilmiş zamanlayıcı) en düşük önceliklidir.

Zamanlayıcıların her birinin onunla ilgili bir altprogram olması gerekir.

Ara vericiler ile ilgili diğer bilgileri "Boş Vaktinizde" adlı konunun 2. bölümünde bulabilirsiniz.

Bağlantılı komutlar: EVERY, REMAIN, RETURN

AND

argüman AND argüman

```
IF "ali"<"can" AND "köpek">"kedi" THEN PRINT "doğru" ELSE PRINT "yanlış" doğru.
IF "can"<"ali" AND "kedi">"köpek" THEN PRINT "doğru" ELSE PRINT "yanlış" yanlış
IF "ali"<"can" AND "kedi">"köpek" THEN PRINT "doğru" ELSE PRINT "yanlış" yanlış
PRINT 1 AND 1
1
PRINT 0 AND 0
0
PRINT 1 AND 0
0
```

OPERATÖR: Tam sayılarda ikili tabana çevirip kontrol eder.
Bağlantılı komutlar: OR, NOT, XOR

ASC

ASC(<karakter dizisi>)

```
PRINT ASC ("x")
120
```

FONKSİYON: Karakter dizisinin ilk karakterinin kod numarasını verir.

İlgili anahtar kelime: CHR$

ATN

ATN (<sayısal ifade>)

```
PRINT ATN (1)
0.7853G8163
```

FONKSİYON: Sayısal ifadenin Ark-TaNjantını hesaplar.

DEG VE RAD kullanarak hesapları ona göre yapar.

Bağlantılı komutlar: COS, DEG, RAD, SIN, TAN

AUTO

AUTO [<satır numarası>] [,<artış>]

```
AUTO 100,50
```

KOMUT: Satır numaralarını otomatik olarak çıkartmak içindir. İsteğe bağlı satır numarası parametresi programın belirli bir noktasından itibaren satır numarası çıkartmaya başlamak içindir. Eğer parametre konmazsa, satır numarası 10'dan itibaren başlar.

Tercihe bağlı artış yeni satırın numarasını belirler. Eğer parametre konulmazsa her satır numarası 10 artarak yazılır.

Üretilen satır numarası halen kullanılıyorsa, satırın muhteviyatı ekranda görüntülenir ve gerekiyorsa düzeltme yapılabilir. Görüntülenen satır (RETURN) tuşuna basılırsa hafızadakinin yerine geçer.

Otomatik sıfır numaralama [ESC] tuşuna basılarak durdurulabilir.

Bağlantılı komut: Yok

BIN$

BIN$ (<işaretsiz tamsayı ifadesi> [,<tamsayı ifadesi>]

```
PRINT BIN$(64,8)
01000000
```

FONKSİYON: işaretsiz tam sayı ifadesinin değerini BINary (2 tabanına göre) olarak gösterir. Burada ikinci tam sayı ifadesi binary hanesinin sayısını belirtir. Bu da Ø ile 16 arasındadır. Eğer belirtilen hame sayısı daha büyükse sonuç ifadenin ön tarafı sıfırlarla dolar. Belirtilen hane sayısı daha küçükse sonuç belirtilen hane sayısına göre kısaltılmaz fakat ne kadar gerekliyse o kadar yazılır.

işaretsiz tam sayı ifadesi -32768 ile 65535 değerleri arasında binary forma çevrilebilir.

Bağlantılı komutlar: DEC$,HEX$,STR$

BORDER

BORDER < renk> [,<renk>]

```
1Ø REM 729 border kombinasyonu
2Ø SPEED INK 5,5
3Ø FOR a=Ø TO 26
4Ø FOR b=Ø TO 26
5Ø BORDER a,b:CLS:LOCATE 14,13
6Ø PRINT "border";a;",";b
7Ø FOR t=1 TO 5ØØ
8Ø NEXT t,b,a
run
```

KOMUT: Ekrandaki border (çerçeve) rengini değiştirir. Eğer iki renk belirlenmişse, border SPEED INK komutunda belirtilen hızda iki renk arasında değişir. Border renkleri 0 ile 26 arasında değişir.

Bağlantılı komut: SPEED INK

BREAK

ON BREAK CONT, ON BREAK GOSUB, ON BREAK STOP komutlarına bakınız.

CALL

CALL < adres ifadesi > [,<parametre listesi>]

```
CALL Ø
```

KOMUT: Harici olarak geliştirilmiş bir alt programı BASIC'den istenmesini sağlar. Yukarıdaki call, bilgisayarı tamamen resetler.

Tecrübesi olmayanların kullanacağı bir komut değildir.

Bağlantılı komut: UNT

CAT

```
CAT
CAT
```

KOMUT: Diskin kataloglanmasını sağlar. Bulunan tüm dosyaların (her dosyanın uzunluğu en yakın üst Kbayt cinsinden belirterek) tam adlarını görüntüler. Diskdeki kullanıma serbest olarak kalan alan, sürücü ve kullanıcı tanımlarıyla birlikte görüntülenir.

Kataloglama hafızada hazır bulunan programı etkilemez.

Bağlantılı komutlar: LOAD, RUN, SAVE

CHAIN

CHAIN < dosya adı > [,<satır numarası>]

```
CHAIN "testprog.bas",35Ø
```

KOMUT: Hafızada hazır bulunan program yerine diskden bir program yükler. Yeni program ya başlangıcından yada satır numarası belirlenmişse o satırdan çalışmaya başlar.

Korunmuş dosyalar (SAVE,p) CHAIN ile yüklenip çalıştırılabilirler.

Bağlantılı komutlar: CHAIN MERGE, LOAD, MERGE

CHAIN MERGE

CHAIN MERGE < dosya adı > [,< satır numarası >] [,DELETE < satır numarası sınırı >]

```
CHAIN MERGE 'newrun.bas", 750, DELETE 400-680
```

KOMUT: Diskden yüklediği programı hafızada halen hazır bulunan programa birleştirir sonra meydana gelen programı seçmekte olan satır numarası belirtilmişse belirtilen satırdan aksi halde baştan çalıştırmaya başlar. Eğer program birleştirilmeden önce orjinal programın bir kısmı çıkarılmak isteniyorsa DELETE satır numarası sınırı tercihi belirtilmelidir.
Yeni programda bulunan eski program satır numaralır birleştirme yapıldıktan sonra yeni program satırları olarak değişir. (yeni satır içerikleri eski satırların üzerine yazılır)
Korunmuş dosyalar CHAIN MERGE ile yüklenip çalıştırılmalıdır.

Bağlantılı komutlar: CHAIN, DELETE, LOAD, MERGE

CHR$

CHR$ (< tam sayı ifadesi >)

```
10 FOR x = 32 TO 255
20 PRINT x;CHR$(x)
30 NEXT
run
```

FONKSİYON: Bir tam sayı ifadesini (0'dan 255'e kadar kadar) karşılığı olan karaktere çevirir. 0'dan 31'e kadar olan karakterler ise kontrol karakterleridir.

Bağlantılı komut: ASC

CINT

CINT (< sayısal ifade >)

```
10 n = 1.9999
20 PRINT CINT (n)
run
2
```

FONKSİYON: Sayısal ifadenin değerini -32768 ile 32767 arasında yuvarlanmış halde gösterir.

Bağlantılı komutlar: CREAL, FIX, INT, ROUND, UNT

CLEAR

```
CLEAR
CLEAR
```

KOMUT: Bütün değişkenleri sıfır veya boş yaparak temizler. Tüm açık dosyalar bırakılır, kullanma fonksiyonları ve diziler silinir ve BASIC için radyan moduna geçilir.

İlgili anahtar kelime: Yok

CLEAR INPUT

```
CLEAR INPUT

10 CLS
20 PRINT "şimdi harfi yazın"
30 FOR t = 1 TO 3000
40 NEXT
50 CLEAR INPUT
run
```

KOMUT: Daha önce tuş takımından yazılmış girdileri atar. Bu komutun etkisini görmek için yukarıdaki programı RUN yapın ve sorduğu zaman harfleri yazın. 50. satırı silin ve tekrar RUN yaparak aradaki farkı görün.

Bağlantılı komutlar: INKEY, INKEY$, JOY

CLG

```
CLG [<ink>]
LOCATE 1,20
CLG 3
```

KOMUT: Ekrandaki grafikleri temizler, eğer ink belirlenmişse grafik kağıdı (paper) o renge ayarlanır.

Bağlantılı komutlar: CLS, GRAPHICS PAPER, INK,ORIGIN

CLOSEIN

```
CLOSEIN
CLOSEIN
```

KOMUT: Diskden giriş dosyalarını kapatır. (OPENIN'e bak)

Bağlantılı komutlar: EOF, OPENIN

CLOSEOUT

```
CLOSEOUT
      CLOSEOUT
```

KOMUT: Diske çıkış dosyalarını kapatır. (OPENOUT'a bak)

Bağlantılı komut: OPENOUT

CLS

CLS [# <akıntı (kanal) numarası>]

```
10 PAPER #2,3
20 CLS #2
run
```

KOMUT: Verilen ekran penceresini kendi kağıt renginde temizler. Diğer kanal numarası belirtilmemiş ise o ekran penceresini temizler.

Bağlantılı komutlar: CLG, INK, PAPER, WINDOW

CONT

```
CONT
      CONT
```

KOMUT: [ESC] tuşuna iki defa basıldığında veya programda STOP komutundan sonra programa devam eder. CONT, program değişikliğe uğratılmadığı ve korunmuş program olmadığı takdirde programı uygulamaya devam eder.

CONT komutundan önce direkt komutlar girilebilir.

Bağlantılı komut: STOP

COPYCHR$

COPYCHR$ (# <kanal numarası>)

```
10 CLS
20 PRINT "üst köşe"
30 LOCATE 1,1
40 a$ = COPYCHRS$ ( #0)
50 LOCATE 1,20
60 PRINT a$
run
```

FONKSİYON: Kanaldaki hareket halindeki karakterleri kopya eder. Yukarıdaki program 1,1 konumundaki karakteri (üst solda) kopya eder ve 1,20 konumunda tekrar meydana çıkartır.
Eğer karakter okunması tanımlanamazsa, boş bir dizi geri döner.

Bağlantılı komut: LOCATE

COS

COS (<sayısal ifade>)

```
DEG
PRINT COS (45)
0,707106781
```

FONKSİYON: Sayısal ifadenin COSinüsünü hesaplar. DEG ve RAD, hesaplamalarda derece radyan olarak belirlenmesi için kullanılırlar.

Bağlantılı komutlar: ATN, DEG, RAD, SIN

CREAL

CREAL (<sayısal ifade>)

```
10 a = PI
20 PRINT CINT (a)
30 PRINT CREAL (a)
run
3
3.14159265
```

FONKSİYON: Sayısal ifadenin değerini reel sayıya çevirir.
Bağlantılı komut: CINT

CURSOR

CURSOR [<sistem anahtarı>], [,<kullanıcı anahtarı>]

```
10 CURSOR 1
20 PRINT "soru?";
30 a$ = INKEY$:IF a$ = "" THEN 30
40 PRINT a$
50 CURSOR 0
run
```

KOMUT: Sistem veya kullanıcı anahtarını kursörle on veya off diye ayarlar. O zaman bu anahtarlar 0 (off) veya 1 (on) parametrelerini alırlar. Yukarıdaki INKEY$ komutunda kursör gözükmektedir. Sistem anahtarının 10. satırda kursörü 1'e ayarlamasıyla on pozisyonuna getirilir.
Kursör, sistem ve kullanıcı anahtarların herikiside on (1) pozisyonunda olursa görünür. Sistem anahtarı INPUT komutu ile otomatik olarak on durumana gelir fakat INKEY$ ile off (0) pozisyonuna döner.
Anahtar parametrelerinden biri kullanılmayabilir (ancak ikiside değil). Bir anahtar parametresi hariç tutulsa bile hafızadaki anahtar durumu değişmez.

Bağlantılı komut: LOCATE

DATA

DATA <sabit listesi>

```
10 FOR x = 1 TO 4
20 READ adı$,soyadı$
30 PRINT adı$," ";soyadı$
40 NEXT
50 DATA Ali, Öztürk, Hakan, Özmen
60 DATA Semih, Özman, Ali, Mutlu
run
```

KOMUT: Programda kullanmak için sabit verileri bulundurur. READ komutuyla data değişkene okunur, daha sonra data listesinde ikinci veri okunur. RESTORE komutu istenilen DATA pozisyonuna göstergeyi gönderir.
Daha geniş bilgi için "Boş vaktiniz için" adlı konunun ikinci bölümüne bakınız.

Bağlantılı komut: READ, RESTORE

DEC$

DEC$ (< sayısal ifade > ,< format kalıbı >)

```
PRINT DEC$ (10↑7, "ŁŁ # # # # # # # # #,.# #"
      £10,000,000.00
```

FONKSİYON: Sayısal ifadeyi, belirlenmiş bir format kalıbı kullanarak, ondalık gösterişe çevirir.

Format kalıbı sadece aşağıdaki karakterlerden oluşur:

```
+ - £ $ * # ,↑
```

Bu format alanı belirleyicilerinin kullanımı PRINT USING komutunda anlatılacaktır.

Bağlantılı komutlar: BIN$,HEX$,PRINT USING,STR$

DEF FN

DEF FEN < fonksiyon ismi > [(< fonksiyon parametresi >)] = < ifade

```
10 TIME/300
20 DEF FNsaat = INT (TIME/300-t)
30 EVERY 100 GOSUB 50
40 GOTO 40
50 PRINT "program saniyeleri";
60 PRINT FNsaat; "önce çalıştı"
70 RETURN
run
```

KOMUT: Bir fonksiyonu tarif eder. BASIC, basit değerli, geri dönme fonksiyonlarının kullanılıman ve tarifine izin verir. DEF FN bu mekanizmanın tarif kısmı-

dır ve program içinde çalışan özel fonksiyonlu bir parametre meydana getirir (COS gibi.)

(Yukardaki örnekte FNsaat [ESC] tuşuna bir kere basılarak beklemeye alınsa, iki defa basılarak durdurulsa bile sonra CONT ile devam edilsede dahi devamlı olarak yenilenir.)

Bağlantılı komutlar: Yok

DEFINT

DEFINT <harf sıralaması>

```
10 DEFINT N
20 sayı = 123.456
30 PRINT sayı
run
123
```

KOMUT: Değişkeni tamsayı olarak yazmak için önceden belirler. Bir değişken belli bir karakter (! % $) olmadan karşılaşıldığında önceden belirlenmiş tip kabul edilir. Bu komut değişkenin tam sayı yazması için kullanır. İlk harfler şöyle olabilir:

```
DEFINT a,b,c
```

veya daha kapsamlı olarak şöyle yazılabilir:

```
DEFINT a-z
```

Bağlantılı komutlar: DEFREAL,DEFSTR

DEFREAL

DEFREAL <harf sıralaması>

```
DEFREAL x,a-f
```

KOMUT: Değişkeni reel sayı olarak yazmak için önceden ayarlanır. Bir değişken belirli bir karakter (! % $) olmadan karşılaşıldığında önceden belirlenmiş tip kabul edilir. Bu komut değişkenin reel sayı olarak yazılması için tanımlanmış ilk harfini önceden tespit etmek için kullanılır. İlk harfler şöyle olabilir:

```
DEFREAL a,b,c
```

veya daha kapsamlı olarak:

```
DEFREAL a-z
```

Bağlantılı komutlar: DEFINT, DEFSTR

DEFSTR

DEFSTR <harf sıralaması>

```
10 DEFSTR i
20 isim = "Amstrad"
30 PRINT isim
run
Amstrad
```

KOMUT: Değişkeni dizi tipinde daha önceden belirler. Değişkenle belirli bir karakter olmadan karşılaşıldığında (! % $) önceden belirlenmiş tip kabul edilir. Bu komut değişkenin diziye tanımlanmış ilk harfi (harfleri)ni yazmak için kullanılır. İlk harflerin listesi şöyle olabilir:

```
DEFSTR a,b,c
```

Ya da geliştirilmiş olarak şöyle olabilir:

```
DEFSTR a-z
```

Bağlantılı komutlar: DEFINT, DEFREAL

DEG

```
DEG
    DEG
```

KOMUT: Hesaplarda derece modunu seçer. SIN,COS,TAN ve ATN daha önceden belirlenmiş şartlarda radyan'dır. "DEG" RAD, NEW, CLEAR, LOAD, RUN, vs. gibi komutlarla değiştirilmediği sürece BASIC'i dereceye ayarlar.

Bağlantılı komutlar: ATN, COS, RAD, SIN, TAN

DELETE

DELETE <satır numara aralığı>

```
DELETE 100-200
```

KOMUT: programdan belirlenen satır numarası aralığı kısmını çıkartır.

İlk veya son numara (satır numara aralığı) "programın başından itibaren" yahut "programın sonuna kadar" diye belirterek de çıkartılabilir.

```
DELETE -200:Baştan 200.satıra kadar
```

veya

```
DELETE 50-:50:satırdan sona kadar
```

veya

```
DELETE :Tüm programı siler.
```

Bağlantılı komutlar: CHAIN, MERGE, RENUM

DERR

```
DERR
LOAD "xyz.abc"

XYZ.ABC not found
Ready
PRINT DERR
146
```

FONKSİYON: Disk dosyalama sisteminden geri çevrilen en son hata kodunu rapor eder. DERR meydana gelen disk hatasını tetkik etmek için kullanılır. "Bilginiz için" adlı konuda hata mesajları listesine bakınız.

Bağlantılı komutlar: ERL, ERROR, ERROR, ON ERROR GO TO, RESUME

DI

```
DI

10 CLS:TAG:EVERY 10 GOSUB 90
20 X1 = RND * 320:X2 RND * 320
30 Y = 200 + RND * 200:C$ = CHR$(RND * 255)
40 FORX = 320-X1 TO 320 + X2 STEP 4
50 DI
```

```
60 MOVE 320,0,1,:MOVE x-2,Y:MOVE X,Y
70 PRINT " ";C$:FRAME
80 EI:NEXT:GOTO 20
GO MOVE 320,0:DRAW x + 8,Y-16,0:RETURN
run
```

KOMUT: kesintileri (interrupts)dolaylı kesinti alt programında RETURN tarafından dolaylı veya EI tarafından olarak terar harekete geçene kadar etkisiz kılar (ESC kesintisi hariç).

Bir kesme alt programa girdiğinde eşit veya daha düşük öncelikteki kesintileri etkisiz hale getirir.

Komut kelimenin tam anlamı ile programın kesintisiz devam etmesi için kullanılır. Örneğin kaynakların kullanılması için iki alt programın yarışması sırasında. Yukardaki örnekte ana program ve alt program grafik görüntüyü kullanmak için yarışmaktadır.

Kesintelir için geniş bilgi "Boş vaktiniz için" adlı konunun 2. bölümündedir.

Bağlantılı komutlar: AFTER, EI, EVERY, REMAIN

DIM

DIM < değişken listesi >

```
10 CLS
20 DIM arkadaş$(5),telefon$(5)
30 FOR n = 1 TO 5
40 PRINT "Arkadaş numarası";n
50 INPUT "İsim girin"; arkadaş$(n)
60 INPUT "Tel no'sunu girin"; telefon$(n)
70 PRINT
80 NEXT
90 FOR n = 1 TO 5
100 PRINT n; arkadaş$(n), telefon$(n)
110 NEXT
run
```

KOMUT: matrislerin boyutlarını belirler. DIM matrisler için yer tahsis eder ve maksimum değişken sayısını belirler. BASIC'e bir matris için ayrılacak yer hakkında bilgi verilmelidir, aksi takdirde 10 olarak belirler.

Bir matris, 1 değişken adı kullanan, alt numaralı ve matrisdeki her elemanın kendi değeri olan bir değişken listesi tarafından tanımlanır. Matrisin kontrolu FOR NEXT döngüsü tarafından her eleman ele alınarak yapılabilir.

Matrisler çok boyutlu olabilir ve bu tür matrislerin her elemanı matrisin sınırları içindeki pozisyonundan tanımlanır. Örneğin matris şu şekilde boyutlandırılsın:

```
DIM A$ (20,20,20)
```

Matrisin bir elemanı şöyle tanımlanabilir:

```
a$(4,5,6)
```

Bağlantılı komut: ERASE

DRAW

DRAW < x-koordinatı >, < y-koordinatı > [,[<ink>] [,<ink mode>]]

```
10 MODE 0:BORDER 0:PAPER 0:INK 0,0
20 X = RND x 640:Y = RND x 400:Z = RND x 15
30 DRAW X,Y,Z
40 GOTO 20
run
```

KOMUT: Grafik kursörünün şimdiki pozisyonundan x,y koordinatları belirlenmiş bir pozisyona çizgi çizer. Çizginin rengi (ink) ile belirlenmelidir (0 ile 15 arası).

Tercihe bırakılan (ink mode) yazılan ink'in grafik ekrandaki reaksiyonunu belirler. Bu 4 ink mode şöyledir.

0: Normal
1: XOR (eXclusive OR)
2: AND
3: OR

Bağlantılı komutlar: DRAWR, GRAPHICS PEN, MASK

DRAWR

DRAWR < x-uzaklığı >,<-uzaklığı > [,[<ink>][,<ink mode>]]

```
10 CLS:PRINT "Yukarı geliyor musunuz?"
20 MOVE 0,350: FOR n = 1 TO 18
30 DRAWR 20,0,1
40 DRAWR 0,-20,1
```

```
5Ø NEXT: MOVE 348,Ø:FILL 3
6Ø GOTO 6Ø
run
```

KOMUT: Ekranda grafik kursörünün şimdiki pozisyonundan x ve y uzaklıkları belirtilmiş pozisyona çizgi çizer. Çizelecek çizginin rengi (ink) belirlenmilidir (Ø ile 15 arası)

Tercihe bırakılmış (ink mode) yazılan ink'in grafik ekrandaki tersini belirler. Bu 4 ink modu şöyledir:

Ø: Normal
1: XOR
2: AND
3: OR

Bağlantılı komutlar: DRAW, GRAPHICS PEN, MASK

EDIT

EDIT < satır numarası >

```
EDIT 2Ø
```

Komut: Belirtilen satırdaki yanlışı veya eksiği düzeltmek için satırı kursör ile birlikte ekrana getirir.

Bağlantılı komutlar: AUTO, LIST

EI

```
EI
    EI
```

KOMUT: DI komutuyla kaldırılmış kesintiyi tekrar geçerli hale getirir.

Eğer bir kesme alt programında kesinti etkisiz bırakılmışsa alt programın sonundaki RETURN komutuyla otomatik olarak tekrar harekete geçirilir.

Daha fazla bilgi için "Boş vaktiniz için" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: AFTER, DI, EVERY, REMAIN

ELSE

(IF komutuna bakınız)

END

```
END
    END
```

KOMUT: Program işleyişini sona erdirir ve direkt mod'a döner. END komutu programda birkaç tane olabilir ve bir tanesi programın en son satırından sonra diye kabul edilir.

Bağlantılı komut: STOP

ENT

ENT < zarf numarası > [,< zarf bölümü >][,< zarf bölümü >][,< zarf bölümü >][,< zarf bölümü>][,< zarf bölümü >]

```
10 ENT 1,10,-50,10,10,50,10
20 SOUND 1,500,200,10,,1
run
```

KOMUT: Zarf numarasında belirlenmiş ton zarf ayarlar (1 ile 15 arası) ve SOUND komutu ile birleşik olarak kullanılır. Eğer zarf numarası negatif (—) ise (-1 ile -15 arası) SOUND komutunun bitişine kadar zarf tekrarlanır.

Her zarf bölümü 2 veya 3 parametre içerir. Eğer 3 parametre kullanılıyorsa bunlar:

<basamak sayısı >,< basamak boyutu >,< bekleme zamanı >

PARAMETRE 1: < basamak sayısı >

Bu parametre zarf bölümünde istediğiniz seste kaç değişik ton basamağı olduğunu belirtir. Örneğin 10 saniye süren bir notanın bölümünde her 1 saniye 10 ton basamağı isteyebilirsiniz. Bu durumda basamak sayısı parametresi kullanılmalıdır.

Kullanılabilen basamak sayısı 0'dan 239'a kadardır.

PARAMETRE 2: < basamak boyutu >

Bu parametre -128 ile +127 arasında yer alır. Negatifler notanın tonunu yükseltir; pozitifler ise alçaltır. En kısa ton periodu Ø'dır. Ton periodları hakkında geniş bilgiyi "Bilginiz için" adlı konuda bulabilirsiniz.

PARAMETRE 3: < bekleme zamanı >

Bu parametre basamaklar arasındaki zamanı saniyenin 1/100'lük birimler esasına göre belirtir. Bekleme zamanı 0 ile 255 (Ø sayısı 256 diye ele alınır) arasındaki sayılar olabilir yani en uzun zaman 2,56 saniyedir.
Eğer iki parametre kullanılırsa, bunlar:
ton periodu, bekleme zamanıdır.

PARAMETRE 1: <ton periyodu >

Bu parametre ton periyodu için yeni ayarlamayı verir. (SOUND komutundaki 2. parametreye bakınız)

PARAMETRE 2: < bekleme zamanı >

Bu parametre bekleme zananı 0,01 saniyelik birimlerde belirler. Bekleme zamanı numaralar 0 ile 255 arasındadır (0 256 diye kullanılır) ve en uzun zaman 2.56 saniyedir.

GENEL OLARAK

Bekleme zamanının toplam uzunluğu SOUND komutundaki süre parametresinden büyük olmamalıdır, aksi takdirde ses bütün ton basamakları geçmeden biter.
Aynı şekilde SOUND komutundaki süre parametresi bekleme zamanı ndan daha büyük olursa ton basamakları geçip gittikten sonrada ses devam eder ve son ton yüksekliğinde sabit kalır.
ENT komutunda 5 değişik zarf bölümü vardır (her biri yukarda anlatılan 2 veya 3 parametrenin kullanılması ile oluşur).
Ton zarfının ilk basamağı hemen yerine getirilir. Verilen ton zarfı her seferinde ayarlanır. Daha önceki değeri kaybolur.
Zarfı numarasını belirtmek ve zarfa bölümünün olmaması önceki ayarlamayı iptal eder.
Bağlantılı komutlar: ENV, SOUND

ENV

ENV <env. no'su>[,<env.bölümü>][,<env.bölümü>]
[,<env.bölümü>][,<env.bölümü>]
[,<env.bölümü>]

```
10 ENV 1,15,-1,10,15,1,10
20 SOUND 1,200,300,15,1
run
```

KOMUT: SOUND komutu ile bağlantılı olarak kullanılan zarf numarasında belirtilen (1 ile 15 arası) ses şiddetini ayarlar.

Her bir zarf bölümü 2 veya 3 parametre içerir. Eğer 3 parametre kullanılıyorsa bunlar: <basamak sayısı> , <basamak boyutu>, <bekleme zamanı>

PARAMETRE 1:<basamak sayısı>

Bu parametre zarf bölümünde geçmesini istediğiniz sesin kaç değişik şiddet basamağı olduğunu belirler. Örnek olarak 10 saniye süren bir nota bölümünde her biri 1 saniyelik 10 ses şiddeti basamağı isteyebilirsiniz. Bu durumda basamak sayısı parametresi 10 olmalıdır.

Kullanılabilen basamak sayısı 0'dan 127'ye kadardır.

PARAMETRE 2: <basamak boyutu>

Her basamak bir öncekine göre ses seviyesi olarak 0'dan 15'e kadar değişir. 16 değişik ses seviyesi SOUND komutunda duyulanla aynıdır. Bununla beraber basamak boyutu parametresi -128 ile 127 arasında kullanılır (ses seviyesi 0 ile 15 arasındayken).

PARAMETRE 3: <bekleme zamanı>

Bu parametre basamaklar arasını 0.01 saniyelik birimler de belirler. Bekleme zamanı numarası 0 ile 255 arasında değişir (0 256 diye ele alınır). Basamaklar arasında en uzun zaman 2,56 saniyedir.

Eğer iki parametre kullanılıyorsa bunlar:

< donanım (hardware) zarfı> , zarf periodu>

PARAMETRE 1:<donanım zarfı.>

Bu parametre ses entegresinin envelop belirleme kütüğüne gönderilecek değeri belirtir.

PARAMETRE 2: < zarf periyodu >

Bu parametre ses entegresinin zarf period kütüğüne gönderilecek değeri belirtir.

Donanım zarfını kullanırken donanım bilginiz olduğu var sayılır. Böyle bir bilginiz yoksa uygun bir bekleme zamanı parametresi için yazılım zarfı kullanmanız tavsiye edilir.

GENEL OLARAK

Bekleme zamanının toplam uzunluğu SOUND komutundaki süre parametresinden büyük olmalıdır aksi takdirde bütün ses basamakları geçmeden önce es biter (bu durumda ses zarfının geri kalan içeriği iptal edilir).

Aynı şekilde SOUND komutunun süre parametresi bekleme zamanı parametresinden büyük olursa bütün ses basamakları geçtikten sonra da ses devam eder ve son seviyesinde sabit kalır.

ENV komutunda 5 değişik zarf bölümü kullanılabilir (yukarda bahsedilen 2 ve 3 parametreler)

Ses zarfının ilk basamağı hemen işleme sokulur.

Verilen ses zarfı her seferinde ayarlanır ve bir önceki değer kaybolur.

Envelop bölümü olmadan belirlenen envelop numarası önceki ayarı iptal eder.

Bu konuda daha geniş bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakın.

Bağlantılı komutlar: ENT, SOUND

EOF

```
EOF
10 OPENIN "ex1.bas"
20 WHILE NOT EOF
30 LINE INPUT # 9,a$
40 PRINT a$
50 WEND
60 CLOSEIN
run
```

FONKSİYON: Disk inputunun dosyanın sonunda olup olmadığına bakar. Sonda olduğu doğruysa veya herhangi bir dosya açılmamışsa -1; aksi durumda Ø (yanlış) verir.

Bağlantılı komutlar: OPENIN, CLOSEIN

ERASE

ERASE < değişken adı listesi >

```
1Ø DIM a(1ØØ),b$(1ØØ)
ERASE a,b$
```

KOMUT: Artık ihtiyaç kalmayan matrislerin içeriklerini, belleği diğer kullanımlara elverişli hale sokmak için, siler.

Bağlantılı komutlar: DIM

ERL

```
ERL
1Ø ON ERROR GOTO 3Ø
2Ø GOTO 1ØØØ
3Ø PRINT "hatalı satır";ERL
4Ø END
run
```

FONKSİYON: Hatanın görüldüğü satırı rapor eder. Yukardaki örnekte 20. satırda hata var ve ERL fonksiyonu tarafından bildirilecektir.

Bağlantılı komutlar: DERR, ERR, ERROR, ON ERROR GOTO, RESUME

ERR

```
ERR
GOTO 5ØØ
Line does not exist
Ready
PRINT ERR
8
```

FONKSIYON: Enson görülen hata numarasını bildirir. Hata mesajları listesini "Bilginiz için" adlı konuda bulabilirsiniz. Yukardaki örnekte hata no 8 "satır mevcut değil" (line does not exist) hatasıdır.

Bağlantılı komutlar: DERR, ERL, ERROR, ON ERROR GOTO, RESUME

ERROR

ERROR <tamsayı>

```
10 IF INKEY$ = "" THEN 10 ELSE ERROR 17
run
```

KOMUT: Verilen bir hata numarasıyla hata tedbiri alır. Hata mesajları listesi, 1'den 32'ye kadar, "Bilginiz için" adlı konuda verilmiştir. Hata, BASIC tarafından gerçek olduğu meydana çıkarılmış gibi muamele edilip ERR ve ERL'nin uygun değerleri rapor edilirken bir hata ele alma bölümüne atlar.

ERROR'e 33 ile 255 arasında ve hata mesajı meydana getirmek için kullanılan bir tamsayı ifadesi eşlik eder örneğin:

```
10 ON ERROR GOTO 100
20 INPUT "Bir karakter girin";a$
30 IF LEN(a$)< >1 THEN ERROR 100
40 GOTO 20
100 IF ERR = 100 THEN 110 ELSE 130
110 PRINT CHR$(7)
120 PRINT "Bir karakter demiştim"
130 RESUME 20
run
```

Bağlantılı komutlar: ERL, ERR, ON ERROR GOTO, RESUME

EVERY

EVERY < zaman periyodu > [,< sayıcı numarası >] GOSUB < satır numarası >

```
10 EVERY 50,1 GOSUB 30
20 GOTO 20
30 SOUND 1,20
40 RETURN
run
```

KOMUT: BASIC alt programı sabit zaman aralıklarında çağırır. Zaman periyodu, zaman aralığını 0.02 saniyelik birimlerle belirtir. "Sayıcı numarası" (Ø ile 3 arasındadır) dört geciktirili sayaçlardan hangisinin kullanılacağını belirler. Sayıcı 3 en yüksek önceliklidir, sayıcı Ø (önceden belirlenmiş sayıcıdır) en düşük önceliklidir. Daha fazla bilgi için "Boş zamanınız için" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: AFTER, REMAIN

EXP

EXP (<sayısal ifade>)

```
PRINT EXP (6.876)
968.743625
```

FONKSİYON: E'nin verilen sayısal ifade değeri kadar üssünü alır, E'nin değeri 2.7182818'dir ve 1'in tabii logaritmasıdır (Fn).

Bağlantılı komutlar: LOG

FILL

FILL <ink>

```
1Ø MODE Ø
2Ø FOR n = 1 TO 5ØØ
3Ø PRINT "0";
4Ø NEXT
5Ø pencolour = 2 + RND x 13
6Ø FILL pencolour
7Ø GOTO 5Ø
run
```

KOMUT: Grafik ekranda herhangi bir alanı doldurur. Alanın kenarları hali hazırdaki grafik kalemin ink'i ile veya doldurmak için kullanılan ink ile çizgi çizilerek belirlenir. Doldurma şimdiki grafik kursörün durduğu yerden başlar. Eğer bu yer kenarın üzerinde yer alıyorsa hiçbir yer doldurulmaz.
Bağlantılı komutlar: GRAPHICS PEN

FIX

FIX (<sayısal ifade>)

```
PRINT FIX (9.99999)
9
```

FONKSİYON: Sayısal ifadede tam sayıdan sonrasını çıkartarak sıfırlar.
Bağlantılı komutlar: CINT, INT, ROUND

FN

(DEF FN komutuna bakınız)

FOR

FOR < basit değişken >= < başla >TO< son >[STEP < artış miktarı >]

```
10 FOR n=2 TO 8 STEP 2
20 PRINT n;
30 NEXT n
40 PRINT "kime teşekkür edeceğiz?"
run
```

KOMUT: Programın bir bölümünde FOR ve NEXT komutları arasında, kontrol değişkeninin başlangıç ve son değerleri arasında ilerlemesini sağlar. Eğer STEP artış belirlenmemiş ise bunu 1 olarak kabul eder.

Artış miktarı negatif olarak belirlenebilir ancak başlangıç parametresi son parametresinden büyük olmalıdır aksi takdirde kontrol değişkeni ilerlemez.

FOR NEXT döngüleri birbirlerinin içinde olabilir.

NEXT komutuna değişkenin adını koymak isteğe bağlıdır. BASIC isimsiz NEXT'lerin hangi FOR komutuna bağlı olduğunu otomatik olarak bulur.

Bağlantılı komutlar: NEXT, STEP. TO

FRAME

```
FRAME
10 MODE 0
20 PRINT "FRAME KAPALI"
30 TAG
40 MOVE 0,200
50 FOR x=0 TO 500 STEP 4
60 IF f=1 THEN FRAME
```

```
70 MOVE x,200
80 PRINT " ";CHR$(143);
90 NEXT
100 IF f=1 THEN RUN
110 CLS
120 TAGOFF
130 PRINT "FRAME AÇIK"
140 f=1
150 GOTO 30
run
```

KOMUT: Ekrandaki grafik çizimleri ekran geri dönüş sinyali (frame flyback) ile senkronize eder. Bütün etkisi karakter veya grafik karakterlerinin titremeye veya kopmaya uğramadan yumuşak olmasıdır.

Bağlantılı komutlar: TAG, TAGOFF

FRE

FRE (<sayısal ifade>)
FRE (<dizi ifadesi>)

```
PRINT FRE (0)
PRINT FRE ("")
```

FONKSİYON: BASIC tarafından kullanılmamış ne kadar serbest bellek olduğunu belirtir.

FRE (" ") uygun bir boşluk için bir değere dönmeden hafızayı gereksiz şeyleri temizlemeye zorlar.

Bağlantılı komutlar: HIMEM, MEMORY

GOSUB

GOSUB <satır numarası>

GOSUB 12Ø

KOMUT: Belirtilen satır numarasına giderek BASIC alt programına atlar. Alt programın sonundaki RETURN komutu ile GOSUB komutundan sonraki talimattan itibaren program çalışmaya devam eder.

Bağlantılı komutlar: RETURN

GOTO

GOTO <satır numarası>

GOTO 9Ø

KOMUT: Belirtilen satır numarasına gider.

Bağlantılı komutlar: Yok

GRAPHICS PAPER

GRAPHICS PAPER < ink >

```
1Ø MODE Ø
2Ø MASK 15
3Ø GRAPHICS PAPER 3
4Ø DRAW 64Ø,Ø
run
```

KOMUT: Grafik kağıdının ink'ini ayarlar. Örneğin ekrana çizilen grafiğin arka fonu. Devamlı çizgiler çizildiğinde grafik kağıdı görülmez. Yukardaki örnekte MASK komutu kesikli bir çizgi çizilmesini sağlar ve grafik kağıdı görülür.

Grafik kağıdının ink'in (Ø ile 15 arasındadır) TAG işlemdeyken karakterlerinin yazıldığı kağıt alanı için kullanılır ve CLG kullanılarak grafik pencere temizlendiğinde ilk konumuna gelir.

Bağlantılı komutlar. CLG, GRAPHICS PEN, INK, MASK, TAG, TAGOFF

GRAPHICS PEN

GRAPHICS PEN [<ink>] [,<fon modu>]

```
10 MODE 0
20 GRAPHICS PEN 15
30 MOVE 200,0
40 DRAW 200,400
50 MOVE 639,0
60 FILL 15
run
```

KOMUT: Çizilecek çizgilerde ve işaretlenerek noktalarda kullanılacak ink 'i (0-15 arasında) ayarlar. Grafik fon modu 'da ayarlanabilir.
0: Mat fon
1: Şeffat fon
(Şeffaf ton karakterleri, TAG çalışırken, grafik sayfasını etkiler ve noktalı çizgilerde boşluklar olur.) Parametrelerden biri çıkartılabilir fakat herikiside değil. Eğer parametrelerden biri çıkartılırsa kısmi ayarlama değişmez.

Bağlantılı komutlar: GRAPHICS PAPER, INK, MASK, TAG, TAGOFF

HEX$

HEX$ (<işaretsiz tam sayı>[,<alan genişliği>]

```
PRINT HEX$ (255,4)
00FF
```

FONKSİYON: İşaretsiz tam sayıyı haksadesimal (16'lı taban) dizi haline çevirir. Bu çevirmeyi yaparken (alan genişliği)nde bildirilen haneyi (0 ile 16 arasındadır) kullanır. Eğer hane sayısı sonuçta çıkan ifadeden büyük ise ön tarafına 0 koyar. Eğer bildirilen hane sayısı küçükse sonuç kısaltılmaz. Ne kadar hane gerekiyorsa o kadar üretilir.

İşaretsiz tamsayı -32768'den 65535'e kadar değerde heksadesimal forma çevrilebilir.

Bağlantılı komutlar: BIN$, DEC$, STR$, UNT

HIMEM

```
HIMEM
PRINT HIMEM
42619
```

FONKSİYON: BASIC tarafından kullanılan belleğin en yüksek byte adresini verir (MEMORY ile değiştirilebilir)

Bağlantılı komutlar: FRE, MEMORY, SYMBOL, SYMBOL AFTER

IF

If < mantıksal ifade > THEN < seçenek kısmı > [ELSE < seç-kısmı >]

```
10 MODE 1
20 x = CINT (RND x 100)
30 PRINT "Tuttuğum sayısı (0-100 arası) bul"
40 INPUT N
50 IF n < x THEN PRINT n; "çok küçük"
60 IF n > x THEN PRINT n; "çok büyük"
70 IF n = x THEN 80 ELSE C = C + 1:GOTO 40
80 PRINT "Çok iyi bildiniz";
90 PRINT c + 1;"yaklaştınız"
run
```

KOMUT: Mantıksal ifadenin doğruluğunu araştırır ve doğru ise seçenek kısmını işleme koyar. Eğer mantıksal ifade yanlış ise ve ELSE de bir seçenek belirlenmiş ise 0, aksi takdirde hemen altındaki satır işleme konur. IF THEN komutları herhangi bir derinlikte olabilir ve satırın bitmesi ile sona erer. Bu nedenle aynı satırda IF THEN komutlarına bağlı olmayan bir başka cümle olması imkansızdır.

Mantıksal ifadenin sonucu herhangi bir satıra atla komutu aşağıdaki şekillerde olabilir:

```
IF a = 1 THEN 100
IF a = 1 GOTO 100
I1 a = 1 THEN GOTO 100
```

Bağlantılı komutlar: ELSE, GOTO, THEN

INK

INK < ink >, < renk > [,< renk >]

```
10 MODE 1:PAPER 0:PEN 1
20 FOR p = 0 TO 1
30 FOR i = 0 TO 26
40 INK p,i
```

```
5Ø LOCATE 16,12:PRINT "INK";p;".";i
6Ø FOR t=1 TO 4ØØ:NEXT t,i,p
7Ø INK Ø,1: INK 1,24:CLS
run
```

KOMUT: Verilen ink'e göre rengi ayarlar. INK parametresi Ø-15 arası olmalıdır. PEN veya PAPER komutuna ait olan referansı tanımlar. İlk renk parametresi bir tam sayı ifadesi olan renk değeridir ve Ø ile 26 arasında değişir. Seçmeli olan ikinci renk ise iki renk arasında değişeceğini belirtir ve değişme hızı SPEED INK komutu ile hesaplanır.

Bağlantılı komutlar: GRAPHICS PAPER, GRAPHICS PEN, PEN, PAPER, SPEED INK

INKEY

INKEY (<tamsayı ifadesi>)

```
1Ø IF INKEY (55) <> 32 THEN 1Ø
2ØPRINT "SHIFT ve V tuşuna basın"
3Ø CLEAR INPUT
run
```

FONKSİYON: Tuş takımını hangi tuşlara basıldığını bildirmek için sorguya çeker. Tuş takımı her 0.02 saniyede bir tetkik edilir.

Fonksiyon herhangi bir tuşun basılı kalıp kalmadığını -1 değerinin geri dönmesiyle araştırır.

Yukardaki örnek SHIFT ve V tuşunun beraberce basılıp basılmadığını araştırır ve sona erer.

Tuş numaralarını bilgisayarınızın sağ üst köşesinde bulabilirsiniz aynı zamanda "Bilginiz için" adlı konuda açıklama yapılmıştır.

SHIFT CONTROL ve (tam sayı)'da belirlenen tuş bileşimi şu şekilde tanımlanır:

Geri dönen	[SHIFT]	[CONTROL]	Belirlenen tuş
-1	aşağı/yukarı	aşağı/yukarı	yukarı
0	yukarı	yukarı	aşağı
32	aşağı	yukarı	aşağı
128	yukarı	aşağı	aşağı
160	aşağı	aşağı	aşağı

Bağlantılı komutlar: CLEAR INPUT, INKEY$, JOY

INKEY$

```
INKEY$
10 CLS
20 PRINT "Evet veya Hayır'ı (E/H) seçiniz"
30 a$ = INKEY$
40 IF a$ = "" THEN 30
50 IF a$ = "e" OR a$ = "E" THEN 80
60 IF a$ = "h" OR a$ = "H" THEN 90
70 GOTO 30
80 PRINT "Siz Evet'i seçtiniz":END
90 PRINT "Siz Hayır'ı seçtiniz"
run
```

FONKSİYON: Tuş takımını araştırır, herhangi bir tuş basılmışsa bulunan diziye döner. Eğer hiç bir tuşa basılmamışsa INKEY$ boş bir diziye döner. Yukardaki örnekte 40. satır ve 70. satır programa tuş takımı dizisini sorguladıktan sonra 30. satıra geri dönmesini söyler.

Bağlantılı komutlar: CLEAR INPUT, INKEY

INP

INP (<kapı numarası>)

```
PRINT INP (& FF77)
255
```

FONKSİYON: Kapı numarası'nda belirtilen I/O adresinden INPUT değerini geri getirir.

Bağlantılı komutlar: OUT, WAIT

INPUT

INPUT [# <kanal ifadesi>,][;] [<tırnak içine yazılacak ifade ayırıcı>] <değişken listesi>

```
1Ø MODE 1
2Ø INPUT "Çarpma için iki sayı ver, (virgülle, ayrılmış olsun)";a,b
3Ø PRINT a; "çarpı"; b;"eşittir";a x b
4Ø GOTO 2Ø
run
```

KOMUT: Bildirilen kanaldan verileri okur (kanal numarası verilmemişse Ø kanalından okur.)

İlk ";" komut işleme girdikten sonra yeni satıra gitmesini önler.

(Ayırıcı) virgül veya noktalı virgül olabilir. Noktalı virgül soru işareti görünmesine neden olur; virgül ise soru işaretini önler.

Eğer giriş yanlış yapılmışsa, yani yerine Ø harfi girilmişse o zaman BASIC şöyle uyarır:

? Redo from start

Tuş takımından gelecek bütün cevaplar RETURN tuşuna basıncaya kadar bitirilmelidir.

Bağlantılı komut: LINE INPUT

INSTR

INSTR ([<başlama pozisyonu>,]<araştırılan dizi>,<araştırılacak dizi>)

```
10 CLS:FOR n = 1 TO 26
20 alfabe$ = alfabe$ + CHR$(n + 64)
30 NEXT
50 b$ = UPPER$(a$)
60 b$;"bir sayıdır";
70 PRINT INSTR (alfabe$,b$);
80 PRINT "alfabede.":PRINT
90 GOTO 40
run
```

FONKSİYON: Araştırılacak dizi ifadesini bulmak için araştırılan dizi yi araştırır ve bunun ilk bulunduğu yeri belirtir. Eğer araştırılacak dizi araştırılan dizi de bulunamazsa Ø bildirilir.

Araştırmaya hangi pozisyondan başlanacağı, seçimi tercihe bağlı olan başlama pozisyonu parametresinde 1'den 255'e kadar olan birtam sayı ile belirlenir. Aksi takdirde arama araştırılacak dizi nin ilk karakterinden başlar.

Bağlantılı komut: Yok

INT

INT (<sayısal ifade>)

```
PRINT INT (-1.995)
-2
```

FONKSİYON: Virgülden sonraki haneleri atarak en yakın küçük tam sayıya yuvarlar. FIX ile aynı işlemin sadece pozitif sayılarla olan kısmı yapılabilir.

Bağlantılı komutlar: CINT, FIX, ROUND

JOY

JOY (<tam sayı ifadesi>)

```
10 PRINT "Programı durdurmak için -";
20 PRINT "Joystick ile çalış"
```

```
30 IF JOY (Ø)< > Ø THEN END
40 GOTO 10
run
```

FONKSİYON: tam sayı ifadesi nde Joystick'ten bit seviyesinde (0 veya 1) belirtilen sonuçları okur.

Bit	Desimal (ondalık)
Ø: Yukarı	1
1: Aşağı	2
2: Sola	4
3: Sağa	8
4: Ateş 2	16
5: Ateş 1	32

Eğer joystick sola hareket ettirilip üzerindeki "ateş" butonuna basılırsa JOY (0) fonksiyonu ondalık değeri 20'ye getirir: 16 (ateş 2) + 4(sola) = 20
Joystick ile ilgili geniş bilgiyi "Bilginiz için" adlı konuda bulabilirsiniz.

Bağlantılı komutar: CLEAR INPUT, INKEY

KEY

KEY < genişleme işaret sayısı >, < dizi ifadesi >

KEY 11, "border 13:paper Ø:pen 1:ink Ø,13:ink 1,Ø mode 2:list" + CHR$(13) Şimdide ENTER'e basın...

KOMUT: Dizi ifadesini genişleme işaret sayısındaki belirtilmiş tuşa ayarlar. Ø ile 31 arasında 32 genişleme işareti vardır. Bunlar 128'den 159'a kadar tuş değerlerinde tutulurlar. 128 tuştan (sayısal tuş takımında Ø) 140'a (CONTROL ENTER) kadar olan tuşlar 0'dan 9'a kadar sayıları yazmayı ihmal edebilirler. Ondalık noktası, RETURN ve RUN"RETURN, daha önceden belirlenmiştir, fakat diğer dizi ifadesine tekrar ayarlanabilir. Genişleme işareti 13'den 31'e kadar (141'den 159'a kadarki tuş değeri) daha önceden belirlenmiş boş dizidir. Fakat sonraki örnekte göreceğiniz gibi genişletilebilir. KEY komutuyla verilen genişleme işareti sayısı Ø'dan

31'e kadardır veya tercihe bağlı olarak 128'den 159'a kadar tuş değerlerine yansıtılabilir ("Bilginiz için adlı konuda tuş takımlarına bakınız.) Toplam 120 karakter dizi ifadesinde genişletilebilir. Aşırı genişlemede "I'm p r o p e r a r g u m e n t" adlı hata mesajı ekrana gelir.

Bağlantılı komut: KEY DEF

KEY DEF

KEY DEF <tuş numarası>, <tekrar>,[,<normal>[,<shiftli>[,<kontrol]]]

```
KEY 159, "Bu TAB tuşudur"
KEY DEF 68,1,159
```

Şimdi[TAB]tuşuna basın...

KOMUT: Ø ile 79 arasında değişen, belirlenmiş tuş numarasından dönen tuş değerini belirler (açıklama için konu 7 bölüm 4'e bakınız). Normal, shiftli ve kontrol parametreleri, tuş yalnız, SHIFT ile veya CONTROL ile birlikte basıldığında, geri dönerken alacakları değerleri içermelidir. Bu parametrelerin her biri tercihe bağlıdır.

Tekrar parametresi tuşu otomatik tekrar fonksiyonuna ayarlar (Ø off, 1 on). Otomatik tekrarın hızı SPEED KEY komutuyla ayarlanabilir.

Yukardaki örnekte 159 no'lu tuş (genişleme işareti 31'in eşdeğeri) öne genişleme dizisine tahsis edilir, sonra KEY DEF komutu tuş 68'i (TAB tuşu) otomatik tekrar (1) olarak tanımlar ve yalnız basılınca normal değeri 159'a döner.

Yukardaki örnekte normal fonksiyona geri dönmek için:

```
KEY DEF 68,Ø,9
```

yazılmalıdır. Burada 9 TAB'ın ASC 11 değeridir.

Bağlantılı komutlar: KEY, SPEED KEY

LEFT$

LEFT$(<dizi ifadesi>, <gerekli uzunluk>)

```
10 CLS
20 a$ = "AMSTRAD"
30 FOR n = 1 TO 7
40 PRINT LEFT$(a$,n)
50 NEXT
run
```

FONKSİYON: dizi ifadesinden, soldan itibaren gerekli uzunluk parametresinde belirtilen kadar karakter çıkartılarak getirilir. Eğer dizi ifadesi, gerekli uzunluk-dan daha kısa ise tüm dizi ifadesi getirilir.

Bağlantılı komutlar: MID$, RIGHT$

LEN

LEN (<dizi ifadesi>)

```
10 LINE UNPIT "Bir cümle yazın";a$
20 PRINT "Cümle";
30 PRINT LEN(a$);"karakter uzunluğundadır."
run
```

FONKSİYON: dizi ifadesinin toplam karakter sayısını verir.

Bağlantılı komut: Yok

LET

LET <değişken> = <ifade>

```
LET x = 100
```

KOMUT: bir değeri bir değişkene tahsis eder. Eski BASIC'den bir kalıntıdır. Eski BASIC yetiştirme el kitaplarında uyum sağlamanın dışında AMSTRAD BASIC'-de bir yararı yoktur. Yukardaki örnek şöylede yazılabilir:

```
x = 100
```

Bağlantılı komut: Yok

LINE INPUT

LINE INPUT [#<kanal ifadesi>,][;][<tırnak içine yazılacak ifade ayırıcı>] <değişken dizisi>

```
10 LINE INPUT "Metinden bir satır yazın";a$
20 CLS
30 PRINT "Değişken şimdi şuna eşittir:-"
40 PRINT a$
run
```

KOMUT: Belirtilen kanaldan metnin tam bir satırını giriş olarak kabul eder (eğer kanal belirtilmemişse Ø kanalından kabul eder.) İlk seçmeli noktalı virgül satır başını önler.

Ayırıcı noktalı virgül veya virgül olmalıdır. Noktalı virgül soru işaretinin görünmesine sebep olur; virgül soru işaretini önler.

Tuş takımından girdiler RETURN tuşuna basıldığında son bulur.
Disk (veya kaset) kanal 9'dan girişler alınan bir carriage return ile sona erdirilir yada 255 karakterlik bir dizi değişkeni tarafından sona erdirilir.

Bağlantılı komutlar: INPUT

LIST

LIST < satır numara aralığı >, [, # < akıntı ifadesi >]

```
LIST 100-1000, # 1
```

KOMUT: Verilen bir kanaldan program satırlarını listeler. Kanal # Ø daha önceden belirlenmiş ekran kanalıdır ve # 8 ise yazıcı (printer) kanalıdır. Listeleme ESC'e bir defa basmak suretiyle beklemeye alınabilir ve ara çubuğuna (space bar) basarak tekrar başlatılır. ESC'e iki kere basmak listelemeyi tamamen durdurup direkt mod'a geçer. satır numara aralığındaki ilk ve son rakkam listelemeye başlanacak satırı veya listelemenin son satırını belirtir.
Şayet tek rakkam varsa, ya proramın başından itibaren araya kadar, veya oradan programın sonuna kadar olan satırlar ekrana gelir.

```
LIST -200
```

veya

```
LIST 50-
```

Bütün programın listesini isterseniz
LIST yazmanız yeterlidir.
Bağlantılı komut: Yok

LOAD

LOAD <dosya adı> [,<adres ifadesi>]

```
LOAD "discfile.xyz",&2AF8
```

KOMUT: BASIC programı diskten belleğe yükler, bunu yaparken bellekte herhangi bir program varsa yüklediği bu programın yerine geçer. Seçenekli adres ifadesi belirlenmişse bir binary dosyayı o adrese yükler.

Korunmuş bir BASIC programı LOAD komutunu kullanarak yüklenmemelidir çünkü bellek tarafından bozulabilir. Bunun yerine RUN veya CHAIN komutlarını kullanın.

Bağlantılı komutlar: CHAIN, CAHIN MERGE, MERGE, RUN, SAVE

LOCATE

LOCATE [<kanal numarası>,] <x-koordinatı>,<y-koordinatı>

```
10 MODE 1
20 FOR n = 1 TO 20
30 LOCATE n,n
40 PRINT CHR$ (143);"yer"
50 PRINT n;",";n
60 NEXT
run
```

KOMUT: Yazı kursörünü, belirtilen kanaldan, x ve y koordinatları belirtilmiş pozisyona yerleştirir. Kanal #0 daha önceden belirlenmiş kanaldır.

Bağlantılı komut: WINDOW

LOG

LOG (<sayısal ifade>)

```
PRINT LOG (9999)
9.21024037
```

FONKSİYON: Sıfırdan büyük sayısal ifade nin tabii logaritmasını hesaplar.

Bağlantılı komutlar: EXP,LOG10

LOG10

LOG10 (<sayısal ifade>)

```
PRINT LOG10(9999)
3.99995657
```

FONKSİYON: Sıfırdan büyük sayısal ifade'nin 10 tabanına göre logaritmasını hesaplar.

Bağlantılı komutlar: EXP, LOG

LOWER$

LOWER$ (<dizi ifadesi>)

```
10 a$ = "HARFLERİN NASIL DEĞİŞTİĞİNİ GÖRÜN"
20 PRINT LOWER$ (a$ + "LOWER CASE")
```

FONKSİYON: Belirlenmiş dizi ifadesini büyük harfler, küçük harf olacak şekilde kopye eder ve yeni diziye götürür. Fonksiyon karışık gelen (büyük ve küçük harf) girdilerin düzenlenmesinde faydalı bir yoldur.

Bağlantılı komut: UPPER$

MASK

MASK (<tam sayı ifadesi>][,<ilk nokta ayarlaması>]

```
10 MODE 0:INK 5,21:INK 8,16
20 MOVE -100*RND,400*RND
30 WHILE XPOS    640
40 FOR x = 1 TO 8
50 MASK 2↑ (8-x)
60 DRAWR 32,0,x,1:MOVER -32,0
70 NEXT
80 MOVER 34,0
90 WEND: GOTO 20
run
```

KOMUT: Çizgi çizmede kullanılan şablon veya maskeyi ayarlar. Tam sayı ifade-

sinin Ø ile 255 arası olan değeri birbirlerine bitiş her 8 pikselik grubun bitlerinin ON (1) veya OFF (Ø) yapar.

İlk nokta ayarlaması çizginin ilk noktasının çizilip (1), çizilmeyeceğini (Ø) hesaplar.

Bu iki parametreden birisi ihmal edilebilir, fakat ikisi birlikte ihmal edilemez. Eğer bir parametre kullanılmazsa kısmi ayarlama değişmez.

Bağlantılı komutlar: DRAW, DRAWR, GRAPHICS PAPER, GRAPHICS PEN

MAX

MAX (<sayısal ifade listesi>)

```
1Ø n=66
2Ø PRINT MAX (1,n,3,6,4,3)
run
66
```

FONKSİYON: Sayısal ifadedeki en yüksek değeri alır.

Bağlantılı komutlar: MIN

MEMORY

MEMORY < adres ifadesi >

```
MEMORY &20AA
```

KOMUT: Basic hafıza parametrelerini en yüksek byte adresine ayarlayarak eldeki hafıza miktarını değiştirmek yeniden ayarlamakta kullanılır.

Bağlantılı komutlar: FRE, HIMEM, SYMBOL, SYMBOL AFTER

MERGE

MERGE < dosya ismi >

KOMUT: Diskten program yükler ve bellekteki halihazırdaki programa ekler

Yeni programdaki mevcut satır numaraları ile birleştirilen eski programın aynı numaralı program satırlarının üzerine yeni programın satırı yazılır.

Korunmaya alınmış (SAVE " ",p) dosyaları halihazırdaki program ile birleştirilemezler.

Bağlantılı komutlar: CHAIN, CHAIN MERGE, LOAD

MID$

MID$ (<dizi ifadesi>, < başama pozisyonu > [,< altdizi uzunluğu >])

```
10 MODE 1: ZONE 3
20 a$ = "ANSIKLOPEDI"
30 PRINT "Bana nasıl hecelendiğini göster";a$
40 PRINT "TAMAM....":PRINT
50 FOR n = 1 TO LEN (a$)
60 PRINT MID$ (a$,n,1)
70 FOR t = 1 TO 700:NEPT t,n
80 PRINT:PRINT
90 INPUT "Şimdi başka bir harf girin";a$
100 GOTO 50
run
```

FONKSİYON: Dizi ifadesinin < başlama pozisyonu >ndan başlayarak < alt dizi uzunluğu boyunda yeni alt diziyi belirler. Eğer < alt dizi uzunluğu > parametresi belirlenmemişse < başlangıç pozisyonu > ndan sonra < dizi ifadesi > nden kalan kısım belirtilir.

Eğer başlangıç poisyonu, dizi ifadesinin toplam uzunluğundan büyükse boş dizi belirtilir. Başlangıç pozisyonunun uzunluğu 1'den 255'e; alt dizi uzunluğunun boyu da 0'dan 255'e kadardır.

Bağlantılı komutlar: LEFT$, RIGHT$

MID$

MID$ (<dizi değişkeni >,< yerleşme pozisyonu >[,< yeni dizi boyu >]= < yeni dizi ifadesi >

```
10 a$ = "hello"
20 MID$(a$,3,2) = "XX"
```

```
30 PRINT a$
run
hexxo
```

KOMUT: Yeni dizi ifadesi, dizi değişkeni tarafından tanımlanan diziye yerleştirir ve yeni dizi boyu kadar karakteri doldurur.

MID$ bir komut olarak kullanıldığında a$ gibi bir dizi değişkeni kullanılmalı ancak "hello" gibi bir dizi sabiti kullanılmamalıdır.

Bağlantılı komutlar: LEFT$, RIHGT$

MIN

MIN(<sayısal ifade listesi >)

```
PRINT MIN (3,6,2.999,8,9)
2.999
```

FONKSİYON: Sayısal ifade listesindeki en küçük değeri belirtir.

Bağlantılı komut: MAX

MOD

< Argüman > MOD < argüman >

```
PRINT 10 MOD 3
1
PRINT 10 MOD 5
0
```

İŞLEM: Birinci argümanin ikinci argümana böldükten sonra kalanı bildirir.

Bağlantılı komut: Yok.

MODE

MODE <tam sayı ifadesi >

```
10 m = m + 1:IF m > 2 THEN m = 0
20 MODE m
30 PRINT "Bu, mode";m
40 PRINT "Şimdi bir tuşa basın"
50 IF INKEY$ = "" THEN 50 ELSE 10
run
```

KOMUT: Ekran modunu, 0,1,2 olarak değiştirir ve ekran ink'ini 0'a döndürür (bu kullanılan paper ink'i olmamalıdır). Bütün yazılar, grafik penceleri sıfırlanır ve yazı ile grafik kursörleri kendi orjinal yerlerine döner.

Bağlantılı komutlar: ORIGIN, WINDOW

MOVE

MOVE <x koordinatı>,<y koordinatı>,[,<ink>][<ink modu>)]]

```
10 MODE 1:TAG
20 x = RND x 800-100:y = RND x 430
30 MOVE x,y
40 PRINT "Buraya hareket ediyorum";
50 GOTO 20
run
```

KOMUT: Grafik kursörünü x ve y koordinatları ile belirtilen mutlak noktaya götürür. Seçmeli ink parametresi grafik pen inkini 0 ile 15 arası değiştirmek için kullanılır. Seçmeli ink modu bir sonra yazılacak ink'i hesaplar. 4 ink modu şöyledir.

0: Normal
1: XOR (exclusiv OR)
2: AND
3: OR

Bağlantılı komutlar: MOVER, ORGIN, XPOS, YPOS

MOVER

MOVER <x uzaklığı>,<y uzaklığı> [<ink><ink modu>]

```
10 MOD 1:TAG:MOVE 0,16
20 PRINT "Hayatta bazen";
```

```
30 FOR N=1 TO 10
40 NOVER -32,16
50 PRINT "yukardasındır";:NEXT:PRINT "bazende";
60 FOR n=1 TO 10
70 MOVER -64, -16
80 PRINT "aşağıda";:NEXT
run
```

KOMUT: Grafik kursörü o anki pozisyonundan izafi bir pozisyona götürür. Bu izafi pozisyon x-uzaklığı ve y-uzaklığı ile belirtilir. Seçmeli ink parametresi grafik pen ink'ini Ø ile 15 arasında değiştirmek için kullanılır. Seçmeli ink modu bir sonra yazılacak ink'i belirler. 4 ink modu yukardaki gibidir.

Bağlantılı komutlar: MOVE, ORIGIN, XPOS, YPOS

NEW

```
NEW
NEW
```

KOMUT: Bellekteki programı ve değişkenleri siler. Tuş tanımları kaybolmaz, görüntü karakteristkileri (mode, pen, paper, ink, vs) değişmez ve ekran temizlenmez.

Bağlantılı komut: Yok

NEXT

NEXT [<değişken listesi>]

```
10 FOR a=1 TO 3
20 FOR b=Ø TO 26
30 MODE 1
40 PEN a:BORDER b
50 PRINT "pen";a;"border";b
60 FOR c=1 TO 500
70 NEXT c,b,a
run
```

KOMUT: FOR döngüsünün sonunu belirtir. NEXT ortak veya ilgili FOR için olabilir. Yukardaki örnekte değişken listesi eş kullandıkları FOR komutlarının sırasıyla ters sıralı olarak yazılmıştır, böylece birbirinin içine yerleşmiş döngüler biribirleriyle kesişmezler.

Bağantılı komutlar: FOR, STEP, TO

NOT

NOT < argüman >

```
IF NOT "ahmet"<"mehmet" THEN PRINT "doğru" ELSE
PRINT "yanlış"
yanlış
```

İŞLEYİŞ: Tam sayılar üzerinde bit seviyesinde işlem yapar. Argümandaki her bit'i ters çevirir.
Mantığı ile ilgili daha fazla bilgi için "Boş vaktiniz için" adlı konunun 2. bölümüne bakın.

Bağlantılı komutlar: AND, OR, XOR

ON BREAK CONT

```
ON BREAK CONT
10 ON BREAK CONT
20 PRINT "Program ESC tuşuna basıp *Break* çıktığında devam edecektir":PRINT
30 FOR t=1 to 1000:NEXT:GOTO 20
run
```

KOMUT: Programı durdurmak için kullanılan [ESC] tuşunun işlevini iptal eder ve program durmaz, bu yüzden programı çalıştırmadan önce SAVE edin.

ON BREAK CONT, Program içinde ON BREAK STOP tarafından etkisiz hale getirilebilir.

Bağlantılı komutlar: ON BREAK GOSUB, ON BREAK STOP

ON BREAK GOSUB

ON BREAK GOSUB < satır numarası >

```
10 ON BREAK GOSUB 40
20 PRINT "program çalışıyor"
30 GOTO 20
40 CLS:PRINT "ESC tuşuna iki defa basın";
50 PRINT "ki GOSUB çalışsın"
```

```
60 FOR t = 1 to 2000:NEXT
70 RETURN
run
```

KOMUT: [ESC] tuşuna iki defa basılınca BASIC'i satır numarasında belirtilmiş alt-programa atlamasını bildirir.

Bağlantılı komutlar: ON BREAK CONT, ON BREAK STOP, RETURN

ON BREAK STOP

```
ON BREAK STOP
10 ON BREAK GOSUB 40
20 PRINT "Program işliyor"
30 GOTO 20
40 CLS:PRINT"ESC tuşuna iki defa basınız"
50 PRINT"iki defa basmak GOSUB alt programı çağırır"
60 FOR t = 1 TO 2000:NEXT
70 ON BREAK STOP
80 RETURN
run
```

KOMUT: ON BREAK CONT ve ON BREAK GOSUB komutlarını etkisiz hale getirir ve ESC tuşu programı durdurabilir. Yukardaki örnekte ON BREAK GOSUB komutu bir kere çalışabiliyor ve ON BREAK alt programının 70. satırında etkisiz hale getiriliyor.

Bağlantılı komutlar: ON BREAK CONT, ON BREAK GOSUB

ON ERROR GOTO

ON ERROR GOTO < satır numarası >

```
10 ON ERROR GOTO 60
20 CLS:PRINT "Eğer hata bulunursa,";
30 PRINT "programın listesi çıkar"
4 FOR t = 1 TO 4000:NEXT
50 GOTO 100
60 PRINT "Hatanın göründüğü satır";

70 PRINT ERL: PRINT:LIST
run
```

KOMUT: Programda bir hata gözlendiği zaman belirtilen satır numarasına atlar.

ON ERROR GOTO Ø komutu hata satırını çalıştırmaz ve BASIC normal hata çalışmasına geçer.

Bir de RESUME komutuna bakın.

Bağlantılı komutlar: DERR, ERL, ERR, ERROR, RESUME

ON <ifade> GOSUB

ON < seçici > GOSUB < satır numara listesi >

```
1Ø PAPER Ø: PEN 1: INK Ø,1
2Ø CLS: PRINT "SEÇENEK TABLOSU":PRINT
3ØPRINT "1- Border değiştirir":PRINT
4ØPRINT "2- Pen değiştirir":PRINT
5ØPRINT "3- Mode değiştirir":PRINT
6ØINPUT "Tercihinizi girin";x
7Ø ON xGOSUB 9Ø, 11Ø, 13Ø
8Ø GOTO 2Ø
9Ø b=b-1:IF b<Ø THEN b=26
1ØØBORDER b: RETURN
11Ø p=p-1:IF p<2 THEN p=26
12Ø INK 1,p:RETURN
13Ø m=m-1:IF m<Ø THEN m=2
14Ø MODE m:RETURN
run
```

KOMUT: seçicinin değerine göre (0'dan 255'e kadar pozitif tam sayı) alt program satırına atlar. Seçici değeri satır numara listesinden seçeceği satır numarasının değeri satır numara listesinden seçeceği satır numarasının değerini belirtir. Yukardaki örnekte 1'i seçmek BASIC'i 90'ıncı, 2'yi seçmek BASIC'i 110'uncu ve 3'ü seçmek BASIC'i 130'uncu satıra atlatır.

Eğer seçici değeri Ø veya satır numara listesindeki satır numaraları sayısından yüksekse hiçbir alt satır seçilmez.

Bağlantılı komutlar: RETURN

ON <ifade> GOTO

ON<seçici>GOTO<satır numarası listesi>

```
1Ø CLS:PRINT "SEÇENEK TABLOSU":PRINT
2Ø PRINT "1- Program listesi":PRINT
3Ø PRINT "2- İlave ve düzeltme":PRINT
4Ø PRINT "3- Disk kataloğu":PRINT
```

```
50 INPUT "Seçiminizi giriniz";n
60 ON n GOTO 80, 90, 100
70 GOTO 10
80 LIST
90 AUTO
100 CAT
run
```

KOMUT: seçicinin değerine göre (0 ile 255 arası tam sayılar) atlayacağı satırı seçer. Seçici değeri satır numarası listesindeki satır namaralarının sırasını belirler. Yukardaki örnekte 1-8'inci, 2-90'ıncı, 3-100'üncü satıra atlatır.

Seçicinin değeri 0 veya satır numarası listesindeki numara sayısından büyük olursa hiçbir satır seçilmez.

Bağlantılı komut: YOK

ON SQ GOSUB

ON SQ (< kanal >) GOSUB < satır numarası >

```
10 ENV 1, 15, -1, 1
20 ON SQ(1) GOSUB 60
30 MODE 0: ORIGIN 0, 0, 200, 440 100, 300
40 FOR x = 1 TO 13: FRAME:MOVE 330, 200, x
50 FILL x:NEXT:GOTO 40
60 READ s:IF s = 0 THEN RESTORE: GOTO 60
70 SOUND 1, s, 25, 15, 1
80 ON SQ (1) GOSUB 60:RETURN
90 DATA 50, 60, 90, 100, 35, 200, 24, 500, 0
run
```

KOMUT: Verilen ses dizisinde boş bir yarık varsa BASIC alt programa gider. Kanal aşağıdaki tam sayılardan biri olabilir.

1:A kanalı için
2:B kanalı için
4:C kanalı için

Ses ile ilgili daha fazla bilgi için "Boş vaktiniz için adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: RETURN, SOUND, SQ

OPENIN

OPENIN <dosya adı>

```
10 REM Diskten bir giriş dosyası aç
20 OPENIN "Veri dosyası": INPUT #9,a,a$
30 CLOSEIN: PRINT "İki değeri:"
40 PRINT:PRINT a,a$
run
```

KOMUT: Hali hazırda kullanılan programda kullanılmak için diskten bir giriş dosyası açar. Açılacak giriş dosyası ASCII dosyası olmalıdır.

Yukardaki örnek ancak aşağıda OPENOUT başlığı altında, yaratılan dosyadan sonra çalışır.

Bağlantılı komutlar: CLOSEIN, EOF

OPENOUT

OPENOUT <dosya adı>

```
10 REM Diske bir çıkış dosyası aç
20 INPUT "Bir sayı değişkeni verin";a
30 INPUT "Bir dizi değişkeni verin';a$
40 OPENOUT "Veri dosyası"
50 WRITE #9, a, a$
60 CLOSEOUT:PRINT "Veri diske yüklendi"
run
```

KOMUT: Diske bir çıkış dosyası açar.

Bağlantılı komut: CLOSEOUT

OR

<argüman> OR <argüman>

```
IF "ahmet"<"mehmet" OR "köpek">"kedi" THEN PRINT "doğru" ELSE PRINT "yanlış" doğru
IF "ahmet"<"mehmet" OR "kedi">"köpek" THEN PRINT "doğru" ELSE PRINT "yanlış" yanlış
IF "ahmet"<"mehmet" OR "kedi">"köpek" THEN PRINT "doğru" ELSE PRINT "yanlış" doğru
.....
PRINT 1 OR 1
1
PRINT 0 OR 0
0
PRINT 1 OR 0
1
```

İŞLEM: Tam sayılar üstünde "boolean" ile işlem yapar. Her iki argüman Ø olmadıkça sonuç 1'dir.

Bağlantılı komutlar: AND, NOT, XOR

ORIGIN

ORGIN <x>, <y>[,<sol>,<sağ>, <üst>, <alt>]

```
10 MODE 1:BORDER 13:TAG
```

```
20 ORIGIN 0, 0, 100, 540, 300, 100
30 GRAPHICS PAPER 3:CLG
40 FOR x = 550 TO -310 STEP -10
50 MOVE x,206
60 PRINT "bu bir grafik penceredir";
70 FRAME:NEXT:GOTO 40
run
```

KOMUT: Grafiğin başlangıç noktasını belirtilen x ve y koordinatlarına göre saptar.
Grafik penceresinin boyutları seçmeli dört parametre tarafından belirlenir. Eğer grafik pencere için belirlenen koordinatları ekran kenarından dışarıda kalıyorsa o zaman grafik pencerenin kenarları ekranın kenarları olarak ele alınır.

Bağlantılı komut: CLG

OUT

OUT <kapı numarası>, <tam sayı ifadesi>

```
OUT & F8F4, & FF
```

KOMUT: tam sayı ifadesi değerini (0-255 arasında) kapı numarasında belirtilen adrese gönderir.

Yeni başlayanlar için kullanılacak bir komut değildir.

Bağlantılı komutlar: INP, WAIT

PAPER

PAPER [# <kanal ifadesi>,]<ink>

```
10 MODE 0:PEN 0:INK 0,13
20 FOR p = 1 TO 15
30 PAPER p:CLS
40 LOCATE 7, 12:PRINT "paper";p
50 FOR t = 1 TO 500:NEXT t,p
run
```

KOMUT: Karakterlerin arka zeminlerinin ink'ini ayarlar. Karakterler ekrana yazılırken, karakter yazılmadan önce karakter hücresi paper (ink) ile doldurulur (0-15 arası) (eğer saydam mod seçilmemişse).

Eğer kanal numarası konulmamışsa PAPER ink için kanal 0 kabul edilir.

Değişik PAPER ink sayısı ekran MODE'una bağlıdır.

Bağlantılı komutlar: GRAPHICS PAPER, INK, PEN

PEEK

PEEK (<adres ifadesi>)

```
10 MODE 1:ZONE 7
20 WINDOW 1, 40, 1, 2:WINDOW # 1, 1, 40, 3, 25
30 PRINT "bellek adresi"
40 LOCATE 20, 1:PRINT "bellek içeriği"
50 FOR n=0 TO 655335
60 P= PEEK (n)
70 PRINT # 1, n, "(&"; HEX$ (n);")";
80 PRINT # 1, TAB (20);p, "(&";HEX$(p);") "
90 NEXT
run
```

FONKSİYON: Adres ifadesinde belirtilen bellek adresinde ne olduğunu belirtir (adres ifadesi &0000 ile &FFFF arasında değişir, yani 0 ile 65635 arası). Bütün durumlarda PEEK, RAM'daki adresi belirlenmiş değerine döner (ROM'a değil) ve &00 ile &FF (0-255) arasındadır.

Bağlantılı komut: POKE

PEN

PEN [# <kanal ifadesi> ,][<ink>][,< arka zemin modu>]

```
10 MODE 0:PAPER 0:INK 0, 13
20 FOR p=1 TO 15
30 PEN p:PRINT SPACE$(47); "pen";p
40 FOR t=1 TO 500: NEXT t,p:GOTO 20
run
```

KOMUT: Verilen ekran kanalını yazarken kullanılacak (ink)i ayarlar (0-15 arası). Arka zemin modu parametresi şeffat (1) veya mat (0) hale getirilebilir.

Son iki parametreden biri ihmal edilebilir ancak ikisi birden edilemez. Eğer bir parametre çıkarılmışsa ayarlama değişmez.

Bağlantılı komut: PAPER

PI

PI

```
PRINT PI
3.14159265
```

FONKSİYON: Daire çevresinin çapına bölümünden çıkan sabiteyi verir.

Bağlantılı komutlar: DEG, RAD

PLOT

PLOT < x-koordinatı >,< y-koordinatı >[,[< ink >] [,< ink modu >]]

```
10 MODE 1:BORDER 0:PAPER 0:PEN 1
20 INK 0,0:INK 1,26:INK 2,13,26:DEG
30 FOR x=1 TO 360:ORIGIN 320,200
40 DRAW 50*COS(x),50*SIN(x),1
50 PLOT 100*COS(x),25*SIN(x):NEXT
60 ORIGIN 0,0:t=TIME+700:WHILE TIME<t
70 PLOT RND*640,RND*400:WEND
80 PLOT RND*640,RND*400,2
90 GOTO 90
run
```

KOMUT: x,y koordinatlarıyla belirlenmiş grafik ekrandaki mutlak pozisyona bir nokta çizer. Noktayı işaretleyen (ink) belirtilmelidir (0-15 arası)

Seçmeli ink modu yazılacak ink'in hali hazırdaki grafik ekrandakine tesirini hesaplar.

4 (ink modu) şöyledir:

0: Normal
1: XOR
2: AND
3: OR

Bağlantılı komutlar: GRAPHICS PEN, PLOTR

PLOTR

PLOTR < x uzaklığı > , < y uzaklığı > [,[< ink >][,< ink modu >]]

```
10 REM kursör tuşunu çizgi çizmek için kullanın
```

```
20 BORDER 0: GRAPHICS PEN 1
30 MODE 1:PLOT 320,200
40 IG INKEY (0) = 0 THEN PLOTR 0,1
50 IF INKEY (1) = 0 THEN PLOTR 1,0
60 IF INKEY (2) = 0 THEN PLOTR 0,-1
70 IF INKEY (8) = 0 THEN PLOTR -1,0
80 IF INKEY 49) = 0 THEN 30: REM copy = clear
90 GOTO 40
run
```

KOMUT: Şimdiki grafik kürsör pozisyonuna bağlantılı olarak x uzaklığı ve y uzaklığı değerleriyle grafik ekranda belirlenmiş pozisyona bir nokta işaretler. Seçmeli ink modu yazılacak ink'in grafik ekranındakine tesirini hesaplar. 4 (ink modu) şöyledir:

0: Normal
1: XOR
2: AND
3: OR

Bağlantılı komutlar: GRAPHICS PEN, PLOT

POKE

POKE < adres ifadesi>, <tam sayı ifadesi>

```
10 FOR m = 49152 TO 65535
20 POKE m, 100
30 NEXT
run
```

KOMUT: Tam sayı ifadesinin değerini (0-255 arası) direkt olarak adres ifadesinde belirlenmiş Z80 belleğine (RAM) yazar.

Yeni başlayanların kullanacağı bir komut değildir.

Bağlantılı komut: PEEK

POS

POS (# <kanal ifadesi>)

```
10 MODE 1:BORDER 0:LOCATE 8,2
20 PRINT "kursörün sol/sağ tuşlarını kullan"
30 WINDOW 1, 40, 12, 12: CURSOR 1,1
40 FOR n = 1 TO 19:PRINT CHR$(9);:NEXT
50 IF INKEY (1)<>-1 THEN PRINT CHR$(9);
60 IF INKEY (8)<>-1 THEN PRINT CHR$(8);
```

```
7Ø LOCATE #1, 2, 24
8Ø PRINT #1, "yazı kürsörü";
9Ø PRINT #1, "yatay pozisyon = ";
1ØØ PRINT #1, POS (#0): GOTO 50
run
```

FONKSİYON: Yazı kursörünün, yazı penceresinin sol kenarına göre yatay pozisyonunu bildirir. Kanal ifadesi mutlaka belirtilmelidir ve #Ø olarak daha önce belirlenmiştir.

POS (#8), yazıcı için yatay taşıyıcı pozisyonunu bildirir ki 1 en sol kenardır.

POS (#9) disk dosya kanalındaki mantıksal pozisyonu bildirir, örneğin son "carrige return" sırasında kanala gönderilen yazılacak karakter sayısı.

Bağlantılı komutlar: VPOS, WINDOW

PRINT

PRINT [#<kanal ifadesi>,][<yazılacak konu listesi>]

```
1Ø a$ = "küçük"
2Ø b$ = "Bu büyük bir dizidir"
3Ø PRINT a$;a$
4Ø PRINT a$,a$
5Ø PRINT
6Ø PRINT b$;b$
7Ø PRINT b$,b$
run
```

KOMUT: Verilen kanala yazılacak konu listesini yazar (eğer kanal numarası belirtilmemişse, #Ø dan önceden belirlenmiş kanal numarasıdır.)

Noktalı virgül (;) bilgisayara yeni yazılacak yazıyı, henüz yazılmış yazının yanına yazmasını belirtir. BASIC önce yeni yazılacak konuyu aynı satıra sığıp sığmayacağını kontrol eder, eğer sığmıyorsa noktalı virgüle bakmadan yeni satıra yazar.

Virgül kullanıldığı zaman bilgisayara yeni yazılacak yazıyı bir sonraki yazı bölgesine yazması belirtilmiş olur. BASIC önce yeni yazılacak yazının yazı bölgesine sığıp sığmayacağını kontrol eder, eğer sığıyorsa yazar, aksi takdirde virgüle bakmadan yeni bir satıra yazar.

PRINT SPC
PRINT TAB

PRINT [# < kanal ifadesi >,][<yazılacak konu listesi>][;]
[SPC (<tam sayı ifadesi >)][<yazılacak konu listesi>]

PRINT [#<kanal ifadesi >,][<yazılacak konu listesi >][;]
[TAB(<tam sayı ifadesi>)][<yazılacak konu listesi >]

```
10 PRINT "Bu spc fonksiyonudur"
20 FOR x =6 TO 15
30 PRINT SPC (5) "a"; SPC (x) "b"
40 NEXT
50 PRINT "Bu tab fonksiyonudur"
60 FOR x =6 TO 15
70 PRINT TAB (5) "a"; TAB (x) "b"
80 NEXT
run
```

SPC, tam sayı ifadesinde belirtilen değer kadar boşluk bıraktıktan sonra yazılacak konu listesini bu boşluğun hemen yanına yazar (yazılacak konu uzunluğu satıra sığacağını kabul edersek). SPC'yi noktalı virgül ile bitirmek gerekli değildir.

TAB, yazı penceresinin son kenarına göre boşluk bırakıp yazılacak konu listesinin hemen yanına yazar. Eğer geçerli pozisyon gerekli pozisyondan büyükse "carriage return" uygulanır ve bir sonraki satırda gerekli pozisyona yazılır.

PRINT USING

PRINT [#<kanal ifadesi>][<yazılacak liste>][;]
[USING < format kalıbı >][<ayırıcı> <ifade>]

```
10 FOR x =1 TO 10
20 n = 100000*(RND ↑ 5)
30 PRINT "güzel"; USING  "########,.##";n
40 NEXT
run
```

PRINIT USING, PRINT komutuyla yazılacak ifadenin yazı formatının belirtilmesine imkan tanır. Bu format kalıbının belirlenmesiyle mümkün olur. Ayırıcı, bir virgül veya noktalı virgül olabilir. Format kalıbı aşağıdaki (format tanımlama alanı) kullanılarak yapılmış bir dizi ifadesidir.

Sayısal Formatlar

Her # hane pozisyonunu tanımlar. Örnek kalıp: **######**

. Ondalık noktanın yerini belirle. Örnek kalıp: **######.##**

, Ondalık noktanın solunda gözükür. Her üç hanede bir (binler için) ayrılacağını belirler. Örnek kalıp: **######,.##**

Sayıların çevresinde:

££ (İki hane pozisyonunu belirler) £ işaretinin, ilk hanenin veya ondalık noktanın önüne yazılmasını belirler. £, hane pozisyonlarından birini doldurur. Örnek kalıp: **££######,.##**

****** (İki hane pozisyonu belirler) Baştaki herhangi bir boşluğu * ile doldurmasını belirtir. Örnek kalıp: ****######,.##**

****£** (Üç hane pozisyonu belirler) ** ve ££ seçeneklerinin birleştirilmiş halidir. Örnek kalıp: ****£######,.##**

$$ (İki hane pozisyonu belirler) $ işaretinin, ilk hanenin veya ondalık noktasının hemen önüne yazılacağını belirtir. $ hane pozisyonlarından birini doldurur. Örnek kalıp: $$ **######,.##**

****$** (Üç hane pozisyonunu belirler) ** ve $$ seçeneklerinin birleştirilmiş halidir. Örnek kalıp: ****$######,.##**

+ Bu + veya - işaretlerinin uygun şekilde yazılacağını belirtir. Eğer + kalıbın başında yer alıyorsa, + işareti sayının baş tarafına yazılır. Eğer + kalıbın sonunda yer alıyorsa, işaret sayının sonuna (ve herhangi bir üs alma işleminden sonra) yazılır. Örnek kalıp: + **####.####**

- Bu - işareti sadece kalıbın sonunda görülebilir. Herhangi bir negatif sayıdan sonra yazılacağını belirtir. Eğer sayı pozitif ise bir boşluk bırakır. "-" işareti eğer bu kalıp kullanılmıyorsa daha önceden belirlendiği gibi sayının önüne konur. Örnek kalıp: **####.####-**

↑↑↑↑ Yazılacak sayının üstsel bir şekilde yazılacağını belirtir. Bu hane pozisyonlarından sonra fakat + ve - işaretlerinden önce görülmelidir. Örnek kalıp: **#.####** ↑↑↑↑ +

Format kalıbı (sayılar için) 20 karakteri aşmamalıdır.

Eğer Format kalıbı giriş ifadesi için çok küçük ise örneğin:

```
PRINT USING  "####";  12345678
```

yazılacak sonuç kalıba uyması için kısaltılmaz, bütünlüğü içinde yazılır ancak format yanlışlığını belirtmek için önüne % işareti konur.

Dizi Formatlar

```
10 CLS: a$ = "abcdefghijklmnopqrst"
20 PRINT "giriş ifadesi =   ";a$
30 PRINT:PRINT "! belirtici =   ";
40 PRINT USING "!";a$
50 PRINT:PRINT " \ aralık \ belirtici =   ";
60 PRINT USING " \          \"; a$
70 PRINT:PRINT "&belirtici =   ";
80 PRINT USING "&";a$
90 GOTO 90
run
```

! Dizinin ilk karakterinin yazılacağını belirtir. Örnek kalıp: !

\<aralık> \

Dizinin ilk × karakterinin yazılacağını belirtir. Karakter sayısı olan × kalıptaki \ işaretleri arasında kalan uzunluğa eşittir. Örnek kalıp: \ \

& Bütün dizinin olduğu gibi yazılacağını belirtir. Örnek kalıp: &

Bir dizi için format kalıbı 255 karakteri geçemez.

Sayısal ve dizi format kalıpları, dizi değişkenleri ile gösterilebilir. Örneğin:

```
10 a$ =  "ff#######,.##"
20 b$ = "!"
30 PRINT USING a$;12345.6789;
40 PRINT USING b$;"kuruş"
run
```

Yazı formatları ile ilgili diğer bilgileri "Boş vaktiniz için" adlı konunun 2. bölümünde bulabilirsiniz.

Bağlantılı komutlar: SPC, TAB, USING, ZONE

RAD

RAD

```
RAD
```

KOMUT: Radyan ile ilgili hesaplamaları ayarlar. BASIC, bilgisayar açıldığında, reset'lendiğinde veya NEW, CLEAR , RUN, LOAD, vs. komutlar kullanıldığında, önceden radyan durumuna ayarlanmıştır.
Bağlantılı komutlar: ATN, COS, DEG, SIN, TAN

RANDOMİZE

RANDOMİZE [< sayısal ifade >]

```
RANDOMİZE 123.456
PRINT RND
Ø.258852139
```

KOMUT: Sayısal ifadede belirtilmiş sayı kökünden rastgele sayı üretir. BASIC rastgele sayı üreticisi her sayının bir önceki sayıya bağlı olduğu bir rastgele sıra meydana getirir. RANDOMİZE, rastgele sayı üreticisine yeni bir başlangıç değeri verir, bu değer ya kullanıcı tarafından sayısal ifadede verilen değerdir ya da belirtilen bir değerdir.

RAMDONIZE TIME tekrarlaması zor bir sıra yapar.

Bağlantılı komut: RND

READ

READ <değişken listesi>

```
1Ø FOR n=1 TO 8
2Ø READ a$, c
3Ø PRINT a$;"  ";:SOUND 1,c:NEXT
4Ø DATA işte, 478, burada, 426,8,379,nota
5Ø DATA 358, müzikal, 319,bir,284,ölçü,253,için,239
run
```

KOMUT: DATA cümlesinden verileri okur ve değişkenlere verir. Bunu yaparken otomatik olarak bir sonraki veriye geçer. RESTORE komutu okuma noktasını DA-

TA cümlesinin başına döndürür.
Daha geniş bilgi için "Boş vaktinizde" adlı konuya bakın. Bağlantılı komutlar: DATA, RESTORE

RELEASE

RELEASE <ses kanalı>

```
10 SOUND 65, 1000, 100
20 PRINT "Sesi bırakmak için R tuşuna basın"
30 IF INKEY$(50) = -1 THEN 30
40 RELEASE 1
run
```

KOMUT: SOUND komutundaki tutulmuş durumdaki ses kanallarını serbest bırakır.

Ses kanalı parametresi 1 ile 7 arasındaki tam sayı değerleridir ve şöyle çalışır:

1: A kanalını serbest bırakır.
2: B kanalını serbest bırakır.
3: A ve B kanalını serbest bırakır.
4: C kanalını serbest bırakır.
5: A ve C kanalını serbest bırakır.
6: B ve C kanalını serbest bırakır.
7: A, B ve C kanalını serbest bırakır.

Daha fazla bilgi için "Boş vaktinizde "adlı konunun 2. bölümüne bakınız.

Bağlantılı komut: SOUND

REM

REM <satırın geri kalanı>

```
10 REM Intergalactic Hyperspace Mega-Monster Invaders Deathchase by AMSOFT
20 REM Copyright AMSOFT 1985
```

KOMUT: Programın içine hatırlama ekler. Satırın geri kalanı BASIC tarafından yok sayılır (iki nokta üstüste dahil).

Tek tırnak işareti REM yerine kullanılır sadece DATA cümlesinde ' karakter dizisinin başladığını gösterdiğinden kullanılmaz.

Bağlantılı komut: Yok

REMAIN

REMAIN (<zamanlayıcı numarası>)

```
10 AFTER 500, 1 GOSUB 40
20 AFTER 100, 2 GOSUB 50
30 PRINT "Bu program çalışıyor": GOTO 30
40 REM Bu GOSUB programı 80. satırda etkisiz kılındığından çağrılmayacaktır.
50 PRINT:PRINT "Zamanlayıcı şimdi 1";
60 PRINT "REMAIN tarafından etkisiz hale konuldu"
70 PRINT "Geri kalan zaman birim = ";
80 PRINT REMAIN (1)
run
```

FONKSİYON: Zamanlayıcı numarasında belirtilen gecikme zamanlayıcısından kalan zaman okur ve etkisiz hale getirir. Zamanlayıcı numarası 0 ile 3 arasındadır. Kesiciler ile ilgili fazla bilgi için "Boş vaktiniz için" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: AFTER, DI, EI, EVERY

RENUM

RENUM [<yeni satır numarası>][,[<eski satır numarası>][,<artma>]]

```
10 CLS
20 REM Bu satır 123 olacak
30 REM Bu satır 124 olacak
40 REM Bu satır 125 olacak
RENUM 123, 20, 1
LIST
```

KOMUT: Program satırlarını yeniden numaralar. Eski satır numarası yeniden numaralamaya başlanacak satırı belirti. Eğer eski satır numarası belirtilmemişse, numaralama 1. satırdan başlar. Yeni satır numarası parametresi numaralanacak satırın yeni satır numarasını belirler, eğer belirtilmemişse numaralama 10'dan başlar. Artma parametresi numaralanacak satırların artma miktarıdır, eğer belirtilmemişse 10 kabul edilir. RENUM bütün GOSUB, GOTO gibi satır çağırma komutlarını da içine alır. Bununla beraber KEY komutunda olduğu gibi dizi ifadelerindeki satır numarası referanslarını değiştirmez (aynı şekilde REM, CHAIN, CHAIN MERGE). Satır numaraları 1'den 65535'e kadar geçerlidir.

Bağlantılı komutlar: DELETE, LIST

RESTORE

RESTORE [< satır numarası>]

```
10 READ a$:PRINT a$;" ";
20 RESTORE 50
30 FOR t = 1 TO 500:NEXT:GOTO 10
40 DATA restore'dan sonra data tekrar
50 DATA tekrar okunabilir
run
```

KOMUT: Okuma noktasını ilk DATA cümlesinin sonundan satır numarasında belirtilen DATA satırının başına götürür. Bu parametreyi koymazsak okuma noktasını birinci DATA satırının başlangıcına iade eder. Daha fazla bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: DATA, READ

RESUME

RESUME [<satır numarası>]

```
10 ON ERROR GOTO 60
20 FOR X = 10 TO 0 STEP-1:PRINT 1/X:NEXT
30 END
40 PRINT "Hatadan sonra buraya git"
50 END
60 PRINT "Hata numarası:";ERR" satırdadır";ERL
70 RESUME 40
run
```

KOMUT: ON ERROR GOTO komutu ile bir hata yakalandığı ve işleme konulduğu zaman normal program yürütülmesine devam eder. Eğer devam edeceği satır numarası belirtilmezse hatanın bulunduğu satırdan program tekrar başlar. Yukardaki örnekte satır numarasını çıkartıp tekrar deneyin.

```
70 RESUME
run
```

Bağlantılı Komutlar: DERR, ERL, ERR, ERROR, ON ERROR GOTO, RESUME NEXT

RESUME NEXT

```
RESUME NEXT
10 ON ERROR GOTO 90
```

```
2Ø PRINT "Her seferinde RETURN tuşuna basın"
3Ø INPUT "1";a
4Ø INPUT "2";a
5Ø inpot "3";a :REM syntax error!
6Ø INPUT "4";a
7 INPUT "5";a
8Ø END
9Ø PRINT "Hata";ERR;"satırdadır";ERL
1ØØ RESUME NEXT
run
```

KOMUT: ON ERROR GOTO komutu ile yakalanan ve işleme konulan hatadan sonra programı normal çalışmasına devam ettirir.

Bağlantılı komutlar: DERR, ERL, ERR, ERROR, ON ERROR GOTO, RESUME

RETURN

RETURN

```
1Ø GOSUB 5Ø:PRINT "GOSUB'dan sonra": END
5Ø FOR n = 1 TO 2Ø
6Ø PRINT "alt program"
7Ø NEXT:PRINT
8Ø RETURN
run
```

KOMUT: Alt proramın sonunu belirtir. BASIC alt programdan GOSUB komutundan sonraki emire döner.

Bağlantılı komut: GOSUB

RIGHT$

RIGHT$ (<dizi ifadesi>, <gerekli uzunluk>)

```
1Ø MODE 1: a$ = "CPC6128 bilgisayar"
2Ø FOR n = 1 TO 16: LOCATE 41-n,n
3Ø PRINT RIGHT$ (a$,n)
4Ø NEXT
run
```

FONKSİYON: Gerekli uzunluk parametresinde belirtilen konum dahil dizi ifadesinin sağından çıkartır. Eğer dizi ifadesi gerekli uzunluktan daha kısa ise bütün dizi ifadesi geri gelir.

Bağlantılı komutlar: LEFT$, MID$

RND

RND [(< sayısal ifade >)]

```
10 RANDOMİZE
20 FOR x = 1 TO -1 STEP -1
30 PRINT "rnd parametresi = "; x
40 FOR n = 1 TO 6
50 PRINT RND (x)
60 NEXT n, x
run
```

FONKSİYON: Eğer sayısal ifade pozitif değerse veya belirtilmemişse sıradaki rastgele sayıyı getirir.

Eğer sayısal ifadenin değeri sıfır ise üretilen son rastgele sayının aynısını verir.

Eğer sayısal ifadenin değeri negatif bir sayı ise yeni bir rastgele sayı dizisi başlar ve ilk numarasını verir.

Bağlantılı komut: RANDOMİZE

ROUND

ROUND (<sayısal ifade> [,<desimal>])

```
10 FOR n = 4 TO -4 STEP -1
20 PRINT ROUND (1234.5678,n),
30 PRINT "tamsayı ifadesi ile";n
40 NEXT
run
```

FONKSİYON: Sayısal ifadeyi desimal parametresinde belirtildiği gibi desimal sayısına veya 10 tabanlı üstel ifadeye yuvarlar. Eğer desimal sıfırdan küçükse o zaman sayısal ifade mutlak bir tam sayıya yuvarlanır, öyleki desimaldaki sayıda sıfır noktasından önceki yere yerleştirilir.

Bağlantılı komutlar: ABS, CINT, FIX, INT

RUN

RUN <dizi ifadesi >

```
RUN "disc"
```

KOMUT: BASIC veya binary programı disk'ten yükler ve çalışmaya başlatır. Daha önce yüklenmiş BASIC programı hafızadan silinir.
Korunmuş BASIC programları bu sebepten dolayı doğrudan RUN edilebilir.

Bağlantılı komut: LOAD

RUN

RUN [<satır numarası>]

```
RUN 200
```

KOMUT: Hali hazırdaki BASIC programını belirtilen satır numarasından itibaren çalıştırır. Satır numarası belirtilmemişse programı başından çalıştırır. RUN bütün şimdiki program değişkenlerinin değerlerini sıfırlar. Korunmalı programlar yüklendikten sonra bu şekilde RUN edilmemelidir.

Bağlantılı komutlar: CONT, END, STOP

SAVE

SAVE <dosya adı>[,<dosya tipi>][,<binary parametreler>]

```
SAVE "discfile.xyz"
```

....dosyayı normal, korunmamış BASIC mod'da saklar.

```
SAVE "discfile.xyz",P
```

....dosyayı korunmuş BASIC mod'da saklar

```
SAVE "discfile.xyz",A
```

....dosyayı ASCII mod'da saklar

```
SAVE "discfile.xyz',B,8000,3000,8001
```

....dosyayı Binary mod'da saklar. Örnekte bilgisayarın belleğini 8000 adresinden başlayarak saklar, dosya uzunluğunu 3000 olarak tesbit eder, seçmeli girme adresi 8001'dir.

KOMUT: Bellekte bulunan programı disk'e saklar. Bir binary dosya, diskte saklanan bir bellek alanıdır. Binary parametreleri şunlardır:

<başlangıç adresi>, <dosya uzunluğu>[,<giriş noktası>]

Ekran belleği Binary olarak saklanabilir. Bu "screen dump" (ekran dökümü) olarak bilinir ve aşağıdaki komut kullanılarak gerçekleştirilir:

```
SAVE "screen",B,&C000,&4000
```

Daha sonra ekrana geri yüklemek için;

```
LOAD "screen"
```

Bağlantılı komutlar: CHAIN, CHAIN MERGE, LOAD, MERGE, RUN.

SGN

SGN (<sayısal ifade>)

```
10 FOR N=200 TO -200 STEP -20
20 PRINT "SGN";
30 PRINT SGN (n);n;"değeri için dönüyor"
40 NEXT
run
```

FONKSİYON: Sayısal ifadenin işaretini saptar. Sayısal ifade sıfırdan küçükse -1 getirilir, sayısal ifade sıfır ise 0 getirilir, son olarak sayısal ifade sıfırdan büyükse 1 getirilir.

Bağlantılı komut: ABS

SIN

SIN (<sayısal ifade>)

```
10 CLS:DEG:ORIGIN 0,200
20 FOR n=0 TO 720
30 Y=SIN (n)
40 PLOT n*640/720,198*y:NEXT
50 GOTO 50
run
```

FONKSİYON: Sayısal ifadenin sinüsünü hesaplar.

Bağlantılı komutlar: ATN, COS, DEG, RAD, TAN

SOUND

SOUND <kanal durumu>,< ton periyodu> [,<süre>[,<volume>[,<volume zarfı>[,<ton zarfı>[,<gürültü periyodu>]]]]]

```
1Ø FOR z=0 TO 4095
2Ø SOUND 1,z,1,12
3Ø NEXT
run
```

KOMUT: Bir ses programlar. Komutun parametreleri:

PARAMETRE 1: <kanal durumu>

Kanal durumu parametresi 1'den 255'e kadar bir tam sayıdır. Parametre bir bit'e karşılıktır. Kanal durumunun binary değerinin bit'i aşağıdakileri belirler:

Bit Ø: (on tabanlı 1) sesi A kanalına gönder.
Bit 1: (on tabanlı 2) sesi B kanalına gönder.
Bit 2: (on tabanlı 4) sesi C kanalına gönder.
Bit 3: (on tabanlı 8) kanal A ile randevu
Bit 4: (on tabanlı 16) kanal B ile randevu
Bit 5: (on tabanlı 32) kanal C ile randevu
Bit 6: (on tabanlı 64) ses kanalını tut.
Bit 7: (on tabanlı 128) ses kanalını bırak.

Kanal durumu parametresi 68'in anlamı: "sesi tut 64 konumunda C kanalına (4) gönder" demektir.

PARAMETRE 2: < ton periyodu >

Bu parametre sesin frekansını belirler veya diğer bir deyişle hangi nota olduğunu belirler. (do, re, mi, vs.) Her notanın bir ayar numarası vardır ve bu numara ton peyiyodu 'dur. "Bilginiz için" adlı konuya bakın.

PARAMETRE 3: < süre >

Bu parametre sesin uzunluğunu ayarlar diğer bir deyişle "ne kadar sürecek"dir. Parametre saniyenin 0.01'lik birimleriyle çalışır ve eğer süre 'yi belirtmezseniz bilgisayar saniyenin beşte biri (0.20 sn.) ne daha önceden ayarlanmıştır.

Eğer süre parametresi sıfır ise ses belirlenen volume ses şiddeti zarfının sonuna kadar sürer.

Eğer süre parametresi negatif ise belirlenen ses şiddeti zarfı süre 'nin mutlak değeri kadar tekrar edilir.

PARAMETRE 4: <ses şiddeti>

Bu parametre bir notanın başlangıç ses şiddetini belirtir ve 0 ile 15 arasındaki değerleri alabilir. "Ø" kapalı, "15" en yüksek ses şiddetidir. Eğer herhangi bir sayı belirtilmemişse, bilgisayar daha önceden belirlenmiş 12'yi kullanır.

PARAMETRE 5: <ses şiddeti zarfı>

Notanın kalış süresince volume (ses şiddeti) değiştirmek için kullanılır ve ayrı bir komut olan ENV ile ses şiddeti zarfı belirleyebilir ayrıca 1'den 15'e kadar 15 değişik ses şiddeti zarfı meydana getirebilirsiniz. Ses şiddeti zarfı parametresi, SOUND komutunda kullanılmak için, uygun bir ses şiddeti zarfı referans numarası çağırır.

ENV komutunun tarifine bakınız.

PARAMETRE 6: <ton zarfı>

Notanın kalış süresince ton'un veya frekansının değiştirilmesi için kullanılır, ton zarfını belirlemek için ayrı bir komut olan ENT'yi kullanabilirsiniz ve 1'den 15'e kadar değişen 15 ayrı ton zarfı meydana getirebilirsiniz. Ton zarfı parametresi, SOUND komutunda kullanılmak için, uygun bir ton zarfı referans nemarası çağırır. Eğer ENT komutunda negatif bir zarfı belirlemişseniz SOUND komutundaki zon zarfı parametresindeki bu negatif sayının mutlak değerini kullanır. ENT komutunun tarifine bakınız.

PARAMETRE 7: <gürültü>

Gürültü parametresi Ø ile 31 arasında değiştirilerek sese ilave edilecek gürültüyü (beyaz gürültü) kapamak veya eklemek mümkündür. Bu konuda daha fazla bilgi için "Boş vaktinizde" adlı konuya bakın.

Bağlantılı komutlar: ENT, ENV, ON SQ GOSUB, RELEASE, SQ

SPACE$

SPACE$(<tam sayı ifadesi>)

```
1Ø MODE 1
```

```
20 PRINT "Benimle senin arana";
30 PRINT SPACE$ (9);
40 PRINT "9 boşluk koy!"
run
```

FONKSİYON: Tam sayı ifadesi 'nde belirtilen uzunlukta bir boşluk yaratır.

Bağlantılı komutlar: SPC, STRING$, TAB

SPC

(PRINT SPC'ye bakın)

SPEED INK

SPEED INK <periyod 1>, <periyod 2>

```
10 BORDER 7, 18
20 FOR i = 30 TO 1 STEP -1
30 SPEEK INK i,i
40 FOR t = 1 TO 700: NEXT t,i
run
```

KOMUT:INK ve BORDER komutunda belirtilen iki ink renginin aralarında değişme hızını ayarlar. Periyod 1 kullanılacak 1. rengin, 0.02 sn.lik birimlerde, zamanını belirtir; periyod 2 ise ikinci rengin zamanını ayarlar.

Renk ve tekrar hızlarını seçerken dikkatli olup manyetik etkilerden kaçınılmalısınız.

Bağlantılı komutlar: BORDER, INK

SPEED KEY

SPEED KEY <başlama gecikmesi>, <tekrar periyodu>

```
10 CLS:FOR k = 7 TO 1 STEP -2
20 PRINT "İsminizi yazın, sonra RETURN"
```

```
3Ø SPEED KEY k,k
4Ø LINE INPUT a$:NEXT
5Ø PRINT "Bu çok hoş bir ad!"
run
```

KOMUT: Tuş takımının otomatik tekrar hızını ayarlar. Başlama gecikmesi parametresi, otomatik terrar başlamadan zamanı 0.02 sn.lik birimlere ayarlar.

SPEED KEY sadece otomatik tekrarları daha önceden belirlenmiş tuşlarda veya KEY DEF komutu ile otomatik tekrara ayarlanmış tuşlarda çalışır.

Başlama gecikmesi 'ni küçük değerler ile çalışması istendiğinde tuş takımının önceden ayarlanmış durumuna çevirmek için sayısal tuşlardan birine önprogram yapmak gerekir.

KEY Ø, "SPEED KEY 3Ø,2" + CHR$(13)

SPEED KEY'i önceden belirlenmiş değerine, fØ tuşuna basıldığında, resetler.

Bağlantılı komut: KEY DEF

SPEED WRITE

SPEED WRITE <tam sayı ifadesi>

SPEED WRITE 1

KOMUT: Eğer bağlı ise kasete yazılmasının veya saklanmasının hızını ayarlar. Kasete tam sayı ifadesi 1 iles, 2000 baud (bit/sn.); tam sayı ifadesi Ø ise 1000 baud yazılabilir. Teypten bir dosya yüklerken bilgisayar otomatik olarak doğru yükleme hızını seçer.

Yüksek data verimliliği için SPEED WRITE Ø kullanın. SPEED WRITE komutunun disk operasyonuna hiç bir etkisi yoktur.

Bağlantılı komutlar: OPENOUT, SAVE

SQ

SQ (<kanal>)

```
1Ø SOUND 65, 1ØØ, 1ØØ
2Ø PRINT SQ (1)
```

```
run
67
```

FONKSİYON: Belirtilen kanal'daki ses kuyruğunun durumunu belirtemk için kullanılır. Kanal tam sayı olmalıdır ve şu değerlerden biri olabilir:

1: A kanalı için
2: B kanalı için
4: C kanalı için

SQ fonksiyonu bir bit karşılığı tam sayıdır:

- Bit 0,1 ve 2: kuyrukdaki serbest giriş adedi
- Bit 3,4 ve 5: kuyruğun başındaki buluşma durumu
- Bit 6 : kuyruğun başı tutulmuş
- Bit 7 : kanal halen aktif

Bit Ø en az, bit 7 en sık kullanılandır.

Yukarıdada görüldüğü gibi eğer Bit 6 ayarlanmışsa Bit 7 ayarlanmaz (bunun tersi de geçerlidir). Aynı şekilde 3, 4 ve 5. Bitler ayarlanmışsa 6 ve 7. Bitler ayarlanamaz.

Bu konuda daha fazla bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: ON SQ GOSUB, SOUND

SQR

SQR (< sayısal ifade >)

```
PRINT SQR (9)
3
```

FONKSİYON: Belirtilen sayısal ifade 'nin kare kökünü alır.

Bağlantılı komut: Yok

STEP

(FOR komutuna bakın)

STOP

STOP

```
10 n = 1 TO 30:PRINT n:NEXT
20 STOP
30 FOR N = 31 TO 60 :PRINT n:NEXT
run
cont
```

KOMUT: Programın yürütülmesini, BASIC'in CONT komutu ile yeniden başlatılacak şekilde durdurur. STOP, programı belli bir noktada düzeltmek için durdurmakta kullanılır.

Bağlantılı komutlar: CONT, END

STR$

STR$ (<sayısal ifade>)

```
10 a = &FF :REM 255 hex
20 b = & x 1111 :REM 15 binary
30 c$ = "***"
40 PRINT c$ + str$(a + b) + c$
run
*** 270***
```

FONKSİYON: Sayısal ifadeyi ondalık diziye çevirir.

Bağlantılı komutlar: BIN$, DEC$, HEX$, VAL

STRING$

STRING$ (<uzunluk>, <karakter belirticisi>)

```
PRINT STRING$ (40, "*")
****************************************
```

FONKSİYON: Belirtilen bir karakterden oluşan uzunluğu belirlenmiş sayıda tekrarlanan bir dizi ifadesi verir. Yukardaki örnek aşağıdaki gibi de yazılabilir:

```
PRINT STRING$ (40,42)
****************************************
```

Karakter belirticisi 42, "*"nın ASCII değeridir.

Diğer bir yazılış şekli de PRINT STRING$ (40,CHR$(42))dir.
Bağlantılı komut: SPACE$

SWAP

(WINDOW SWAP komutuna bakın)

SYMBOL

SYMBOL <karakter numarası>, <sıra listesi>

```
10 MODE 1: SYMBOL AFTER 105
20 row1 = 255:REM binary 11111111
30 row2 = 129:REM binary 10000001
40 row3 = 189:REM binary 10111101
50 row4 = 153:REM binary 10011001
60 row5 = 153:REM binary 10011001
70 row6 = 189:REM binary 10111101
80 row7 = 129:REM binary 10000001
80 row8 = 255:REM binary 11111111
100 PRINT "110. satır i harfini (105) olarak tarif ediyor. Bazı i'ler yazın ve görün!
Sonra programı listeleyin"
110 SYMBOL 105, row1, row2, row3, row4, row5, row6, row7, row8,
run
```

KOMUT: Ekrandaki karakterin şeklini tekrar tanımlar. Her iki parametrede 0 ile 255 arasında bir tam sayı olabilir.

Yeni tanımlanan bir karaktere CPC6128 hafızasında yer ayırmak için bilgisayar aşağıdaki komut ile hazırlanmalıdır:

```
SYMBOL AFTER x
```

x tekrar tanımlamayı istediğiniz karakter sayısına eşit veya küçük olabilir.

Tuş takımından direkt olarak yazılabilen karakteri tanımlayan x'in değerine bakmadan şu komut ile, yeniden tanımlanan karakter, ekrana yazılabilir:

```
PRINT CHR$(x)
```

SYMBOL x'den sonra, yukarıdan başlayan 8 ayrı paralel karakter satırını tanımlayan 8 parametre vardır. Her parametre 0 ile 255 arasındadır. 8 parametrerin her birinin binary gösterilişi, bitirilmiş karakterin sırasındaki numuneyi hesaplar.

Örneğin: 8 parametreden ilki 1 ise karakterin en üst sırası binary olarak gösterilmesi ØØØØØØØ1'dir. 1 gözüktüğünde karakter bölümü PEN renginde yazılmıştır, Ø gözüktüğünde karakter bölümü gözükmez çünkü PAPER renginde yazılmıştır. Bu sebeple yeni tanımlanan karakterin üst sırasının sağ üst köşesinde bir noktası vardır. Bu örneğe devam edersek diğer 7 parametreyi 3,17,15,31,63,Ø,Ø diye belirtebiliriz. Bu 8 parametrenin binary gösterilmesi şöyledir:

parametre (row) 1:ØØØØØØØ1 binary: (ondalık 1)
parametre (row) 2:ØØØØØØ11 binary: (ondalık 3)
parametre (row) 3:ØØØØØ111 binary: (ondalık 7)
parametre (row) 4:ØØØ1111 binary: (ondalık 15)
parametre (row) 5:ØØ11111 binary: (ondalık 31)
parametre (row) 6:Ø0111111 binary: (ondalık 63)
parametre (row) 7:ØØØØØØØØ binary: (ondalık Ø)
parametre (row) 8:ØØØØØØØØ binary: (ondalık Ø)

Yukardaki binary gösterilişlerine bakarak yeni karakterin şeklinin nasıl bir şey olacağı hesaplanabilir. Bu parametreleri 255 karakter numarasına ayarlayalım:

```
SYMBOL 255, 1, 3, 7, 15, 31, 63, Ø, Ø
```

Son iki parametrede görülen Ø değeri sizin sadece aşağıdakini yazmanız demektir.

```
SYMBOL 255, 1, 3, 7, 15, 31, 63
```

Ondalık formda yaratmış olduğunuz sembollerin çevrilen kopyalarını saklamak için parametreleri binary olarak girebilirsiniz (&× binary örneğini kullanmayı unutmayın). Örnek olarak:

```
SYMBOL 255, &×ØØØØØØØ1, &×ØØØØØØ11, &×ØØØØØ111, &×ØØØ1111,
&×ØØ11111, &×Ø1111111
```

Şimdi karakteri görelim:

```
PRINT CHR$ (255)
```

Parametreleri tuş takımından yazılabilen karaktere ayarlamak yeni bir karakterin oluşmasına neden olur. Bu uygun bir tuşa basmakla olur ve daha önce hangi karakter yazarsa yazsın değişmez. Bu konuda daha fazla bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakınız.

Bağlantılı komut: SYMBOL AFTER

SYMBOL AFTER

SYMBOL AFTER <tam sayı ifadesi >

```
10 CLS
20 SYMBOL AFTER 115
30 PRINT "40. satır s'yi tekrar edecek";
40 SYMBOL 115, 0, 56, 64, 48, 8, 8, 112
50 PRINT "s'ye"
60 PRINT "Bu s tarifini SYMBOL AFTER 240"
70 PRINT "yazarak iptal ed"
run
```

KOMUT: Kullanıcı tarafından tanımlanmış müsaade edilebilen karakter sayısını belirler(0 ile 256 arası).Önceden belirlenmiş ayarlama 240'dır ve 16 adet kullanıcı tarafından tanımlanan karakter (240-255 arası) verir. Eğer tam sayı ifadesi 32 ise 32'den 255'e kadar bütün karakterler tekrar tanımlanabilir. SYMBOL AFTER 256 hiç bir karakterin yeniden tanımlanmasına izin vermez. SYMBOL AFTER komutu kullanıldığında kullanıcı tarafından tanımlanan bütün karakterler ilk konuma gelir.

SYMBOL AFTER, MEMORY komutunu kullanarak, HIMEM değeri değiştirildikten sonra çağırılırsa veya OPENIN, OPENOUT ile bir dosya "buffer"ı açıldıktan sonra çalışmaz. bu gibi durumlarda "Improper argument" (5 nolu hata) hatası verir (SYMBOL AFTER 256 hariç)

Daha fazla bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: HIMEM, MEMORY, SYMBOL

TAB

(PRINT TAB'a bakınız)

TAG

TAG [# <kanal ifadesi>]

```
10 INPUT "adınızı girin"; a$:CLS
20 PRINT "etrafta dolaşıyormuşsun"; a$
30 TAG
40 x = LEN (a$)?17:y = 50 + RND * 300:MOVE -x,y
50 FOR f = -x TO 640 STEP RND* 7 + 3
60 MOVE f,y:PRINT " ";a$;:FRAME:NEXT
70 FOR b = 640 TO -x STEP -RND?7 + 3
```

```
80 MOVE b,y:PRINT a$;" ";:FRAME:NEXT
90 GOTO 40
run
```

KOMUT: kanal ifadesi 'ne gönderilen metin, grafik kursörün pozisyonuna yazılır. Bu, metin ve sembollerin grafik ile karışmasına imkan verir. Kanal ifadesi belirtilmemişse #0 önceden belirlenmişti.

Karakter hücresinin sol üstü grafik kürsörüne bağlanmıştır ve PRINT bildirimi noktalı virgül ile bitmemişse yazmayan kontrol tuşları (yeni satır görmek ve "carrige return") görüntülenir.

Daha önce belirlenmiş kanalda (#0), BASIC direkt mod'a geri dönerken TAG kapatılır (switch off).

Bağlantılı Komut: TAGOFF

TAGOFF

TAGOFF [# <kanal ifadesi>]

```
10 MODE 2:2:TAG:REM Grafikte metin
20 sene = 1985:FOR x = 1 TO 640 STEP 70
30 MOVE x,400:DRAWR 0, -350
40 sene = sene + 1:PRINT sene;:NEXT
50 TAGOFF:REM Grafik yokken metin
60 LOCATE 34,25:PRINT "Senelik şekiller"
70 GOTO 70
run
```

KOMUT: TAG'ı iptal eder ve metni TAG kullanılmadan önceki metin kürsörü pozisyonuna gönderir.

Bağlantılı komut: TAG

TAN

TAN (<sayısal ifade>)

```
PRINT TAN (45)
1.61977519
```

FONKSİYON: sayısal ifade nin tanjantını hesaplar (-200000 ile +200000 arasındaki sayılar).

DEG ve RAD kullanılarak hesaplardaki sonuçların derece ve radyan cinsinden sonuçları elde edilir.

Bağlantılı komutlar: ATN, COS, DEG, RAD, SIN

TEST

TEST (<x-koordinatı>,<y-koordinatı>)

```
10 CLS
20 PRINT "PEN numarasını kullanıyorsunuz";
30 PRINT TEST (10,386)
40 PRINT "PEN ve MODE değiştirerek deneyin";
50 PRINT ".....sonra RUN"
run
```

FONKSİYON: Grafik kürsörü x ve y koordinatlarıyla belirlenen mutlak pozisyona götürür ve bu yeni yerdeki ink değerini bildirir.

Bağlantılı komutlar: MOVE, MOVER, TESTR, XPOS, YPOS

TESTR

TESTR (<x-uzaklığı>,<y-uzaklığı>)

```
10 MODE 0:FOR x = 1 TO 15: LOCATE 1,x
20 PEN x:PRINT STRING$ (10,143);:NEXT
30 MOVE 200,400:PEN 1
40 FOR n = 1 TO 23:LOCATE 12,n
50 PRINT "pen"; TESTR (0,-16):NEXT
run
```

FONKSİYON: Grafik kürsörü, şimdiki yerinden, belirtilen x ve y uzaklığı değerleri kadar götürür ve bu yeni yerde ink değerini bildirir.

Bağlantılı komutlar: MOVE, MOVER, TEST, XPOS, YPOS

THEN

(IF komutuna bakınız)

TIME

TIME

```
10 CLS:REM saat
20 INPUT "saat";saat
30 INPUT "dakika";dakika
40 INPUT "saniye";saniye
50 CLS:datum = INT (TIME/300)
60 WHILE saat < 13
70 WHILE dakika < 60
80 WHILE tick < 60
90 tick = (INT(TIME/300)-datum) + saniye
100 LOCATE 1,1
110 PRINT USING "      ";saat, dakika, tick
120 WEND
130 tick = 0:saniye = 0:dakika = dakika + 1
140 GOTO 50
150 WEND
160 dakika = 0:saat = saat + 1
170 WEND
180 saat = 1
190 GOTO 60
run
```

FONKSİYON: Bilgisayarın açıldığı andan veya resetlendiği andan itibaren geçen zamanı bildirir.

Gerçek zamandaki her saniye TIME/300'e eşittir.

Bağlantılı komutlar: AFTER, EVERY, WEND, WHILE

TO

(FOR komutuna bakınız.)

TROFF
TRON

TROFF
TRON

```
10 TROF:PRINT:PRINT "TRace-OFF"
20 FOR n = 1 TO 8
30 PRINT "program çalışıyor":NEXT
```

```
40 IF f=1 THEN END
50 TRON:PRINT:PRINT "TRace-ON"
60 f=1:GOTO 20
run
```

KOMUT: Her satırı ele almadan önce satır numarasını yazarak programı yürütürken izleme olanağı verir. Satır numaraları köşeli parantez [ ] içinde görünür. TRON izlemeyi başlatır; TROFF durdurur.

Bu kolaylık, hata meydana gelmeden önce program satırlarının yürütülmesinde çalışmak için faydalıdır.

Bağlantılı komut: Yok

UNT

UNT (<adres ifadesi>)

```
PRINT UNT (&FF66)
-154
```

KOMUT: adres ifadesi 'nin değerinin ikiz eşleniği olan -32768 ile +32767 arasında bir tamsayı verir.

Bağlantılı komutlar: CINT, FIX, INT, ROUND

UPPER$

UPPER$ (<dizi ifadesi>)

```
10 CLS:a$="buraya yazılanlar büyük harf olacak"
20 PRINT UPPER$ (a$)
run
```

FONKSİYON: Dizi ifadesinin belirttiği karakter dizisini (A'dan Z'ye) büyük harfe çevirir.

Bağlantılı komut: LOWER$

USING

(PRINT USING'e bakınız)

VAL

VAL (<dizi ifadesi>)

```
10 CLS:PRINT "Çarpım tablosu yapalım"
20 PRINT:PRINT "1'den 9'a kadar bir tuşa basınız"
30 a$ = INKEY$:IF a$ = "" THEN 30
40 n = VAL (a$):IF n <1 OR n> 9 THEN 30
50 FOR x = 1 TO 12
60 PRINT n;"X";x;" = ";n * x
70 NEXT:GOTO 20
run
```

FONKSİYON: dizi ifadesi nde belirtilen ilk karakteri, negatif işareti ve ondalık noktası dahil olarak getirir.

Eğer ilk karakter sayı değilse Ø getirilir. Eğer ilk karakter eski işaretli veya ondalık noktadan sonra gelen sayısal olmayan bir karakter ise 9'Type mismatch'' hata mesajı (13) verilir.

Bağlantılı komut: STR$

VPOS

VPOS (# <kanal ifadesi>)

```
10 MODE 1: BORDER Ø:LOCATE 8,2
20 PRINT "kürsör yukarı/aşağı tuşlarını kullanın"
30 WINDOW 39, 39, 1, 25;CURSOR 1,1
40 LOCATE 1,13
50 IF INKEY (Ø)< > -1 THEN PRINT CHR$(11);
60 IF INKEY (2) < >-1 THEN PRINT CHR$(10);
70 LOCATE # 1, 3, 24
80 PRINT # 1, "metin kürsörü";
90 PRINT # 1, "dikey pozisyon = ";
100 PRINT 1, VPOS ( # Ø):GOTO 50
run
```

FONKSİYON: Metin penceresine göre metin kürsörünün dikey pozisyonunu verir. Kanal ifadesi mutlaka belirtilmeli ve daha önceden belirlenen # Ø olmalıdır.

Bağlantılı komutlar: POS, WINDOW

WAIT

WAIT <kapı numarası>, <kalıp> [,<çevirme>]

```
WAIT &FF34, 20, 25
```

KOMUT: Belirtilen I/O kapı numarasına 0 ile 255 arasında bir değer velinceye kadar bekler. BASIC, I/O kapısı okunurken çevirim içine girer. Okunan değer çevirme ile exclusive OR, kalıp ile AND'dir; taki sıfır olmayan sonuç görünene kadar.

BASIC, gereken şartlar ortaya çıkana kadar bekler.

Yeni başlayanların kullanabileceği bir komut değildir.

Bağlantılı komutlar: INP, OUT

WEND

WEND

```
WEND
```

KOMUT: WHILE çeviriminde yürütülecek programın sonunu belirtir. WEND otomatik olarak WHILE komutunu seçer.

Bağlantılı komutlar: TIME, WHILE

WHILE

WHILE <mantıksal ifade>

```
10 CLS: PRINT "On saniyelik zamanlayıcı";t = TIME
20 WHILE TIME t + 3000
30 SOUND 1, 0, 100, 15
40 wend: SOUND 129, 40, 30, 15
run
```

KOMUT: Verilen satırlar doğru olduğu sürece bir program kısmının yürümesini tekrar ettiririr. WHILE komutu çevrimin başını ve mantıksal ifadedeki şartı tanımlar.

Bağlantılı komutlar: TIME, WEND

WIDTH

WIDTH <tamsayı ifadesi>

```
WIDTH 40
```

KOMUT: Yazıcı ünite bağlandığında, BASIC'e yazılacak her satırın karakter sayısını bildirir. BASIC buna göre uygun zamanda "extra carriage return" (satırbaşı) yapar.

Bilgisayar, WIDTH komutu belirtilmezse, 132 karaktere ayarlanmıştır. WIDTH 255 komutu yazıcının ekstra satırbaşı yapmasını önler.

Bağlantılı komut: POS

WINDOW

WINDOW [# <kanal ifadesi>,]<sol>, <sağ>, <üst>, <alt>

```
10 MODE 0:BORDER 0:REM test kartı
20 INK 0,0:INK 1,25:INK 2,23:INK 3,21
30 INK 4,17:INK 5,6:INK 6,2:INK 7,26
40 PAPER 0:CLS
50 PAPER 1:WINDOW 2,4,1,18:CLS
60 PAPER 2:WINDOW 5,7,1,18:CLS
70 PAPER 3:WINDOW 8,10,1,18:CLS
80 PAPER 4:WINDOW 11,13,1,18:CLS
90 PAPER 5:WINDOW 14,16,1,18:CLS
100 PAPER 6:WINDOW 17,19,1,18:CLS
110 PAPER 7:WINDOW 2,19,19,25:CLS
120 GOTO 120
run
```

KOMUT: Metin penceresinin ekrandaki boyutunu tespit eder. Sol, sağ, üst ve alt parametre değerleri ekran MODE'unu kullanırken sabit karakter yerleri ile ilgilidir.

Eğer kanal ifadesi belirtilmemişse BASIC kanalı # 0 diye belirler.

Daha fazla bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakınız.

Bağlantılı komut: WINDOW SWAP

WINDOW SWAP

WINDOW SWAP <kanal ifadesi>, <kanal ifadesi>

```
10 MODE 1:INK 1,24:INK 2,9:INK 3,6
20 WINDOW 21,40,13,25:PAPER 3
```

```
30 WINDOW # 1,1,20,1,12:PAPER 1,2
40 CLS:PRINT "pencere numarası 0"
50 CLS 1:PRINT # 1, "pencere numarası 1"
60 LOCATE 1,6
70 PRINT "kırmızı pencere (0)";SPC(2)
80 LOCATE # 1,1,6
90 PRINT # 1, "yeşil pencere (1)"
100 FOR t=1 TO 1000:NEXT
110 WINDOW SWAP 0,1:GOTO 60
run
```

KOMUT: İlk kanal ifadesi'nde belirtilen metin penceresini ikinci kanal ifadesi'nde belirtilen ile değiş dokuş eder.
Her iki kanal ifadesi mutlaka belirtilmelidir ve bu durum kanal idarecisinden önce olmamalıdır. Komut BASIC tarafından üretilen mesajları tekrar yönlendirmek için kullanılmalıdır. Normal olarak mesajlar kanal # 0'a gönderilirler. Pencerelerle ilgili daha fazla bilgiyi "Boş vaktinizde" adlı konunun 2. bölümünde bulabilirsiniz.

Bağlantılı komut: WINDOW

WRITE

WRITE [# < kanal ifadesi > ,][< yazı listesi >]

```
10 REM diskin üzerine değişkenleri yaz
20 INPUT "değişken numarasını verin";a
30 INPUT "dizi değişkenini verin;a$
40 OPENOUT "veri dosyası"
50 WRITE # 9, a,a$
60 CLOSEOUT:PRINT "Veri diskete kaydedildi"
run
```

KOMUT: Yazı ifadesi'ndeki parçaları değerlerini kanal ifadesi'nde belirtilen kanala yazar. Parçalar virgüllerle ayrılır; dizi çift tırnak içine konur.

Bu örnekte değişken değerleri kanal # 9'a yazılır (disk kanalı).

(Bu değişkenleri disketten çağırmak için aşağıdaki programı kullanmak gerekir.)

```
10 REM değişkenleri disketten bulup getirmek
20 OPENIN "Veri dosyası":INPUT # 9, a.a$
30 CLOSEIN:PRINT "İki değer="
40 PRINT:PRINT a,a$
run
```

Bağlantılı komutlar: INPUT, LINE INPUT

XOR

< arguman > XOR < argüman >

```
IF "ahmet" < "mehmet"XOR "köpek" > "kedi" THEN PRINT "doğru" ELSE PRINT "yanlış" yanlış
IF "mehmet"< "ahmet" XOR "kedi" > "köpek" THEN PRINT "doğru" ELSE PRINT "yanlış" yanlış
IF "ahmet" < "mehmet" XOR "kedi" > "köpek" THEN PRINT "doğru" ELSE PRINT "yanlış" doğru


....
PRINT 1 XOR 1
0
PRINT 0 XOR 0
0
PRINT 1 XOR 0
1
```

Çalışması: Tam sayılarda bit türü "boolean" çalışmasını yapar. Her iki argüman da aynı olmadığı taktirde sonuç 1'dir (exclusive OR).

Daha fazla bilgi için "Boş vaktinizde" adlı konunun 2. bölümüne bakınız.

Bağlantılı komutlar: AND, OR, NOT

XPOS

```
XPOS
10 MODE 1: DRAW 320, 200
20 PRINT "Grafik kürsörün x pozisyonu = ";
30 PRINT XPOS
run
```

FONKSİYON: Grafik kürsörün o andaki yatay (x) konumunu bildirir.

Bağlantılı komutlar: MOVE, MOVER, ORIGIN, YPOS

YPOS

```
YPOS
10 MODE 1: DRAW 320, 200
20 PRINT "Grafik kürsörün y pozisyonu = ";
30 PRINT YPOS
run
```

Bağlantılı komutlar: MOVE, MOVER, ORIGIN, XPOS

ZONE

ZONE <tam sayı ifadesi>

```
1Ø CLS: FOR z=2 TO 2Ø
2Ø ZONE z
3Ø PRINT "X", "X ZONE =";z:NEXT
```

KOMUT: Yazılacak parçaların arasına virgül konularak PRINT de kullanılan baskı bölgesinin genişliğini değiştirir. Daha önceden belirlenmiş baskı bölgesi 13 kolonluktur, fakat tam sayı ifadesi ile 1'den 255'e kadar değiştirilebilir.

Bağlantılı komut: PRINT

KONU 4
Disk ve Kaseti Kullanmak

Bölüm 1: Disk
Disk ile Çalışma

Bu bölümde disklerin nasıl kullanılacağı açıklanacak ve CP/M ile kullanım kolaylıkları tanıtılacaktır.

Konular

* Esas disk'in (Master Disc) kopyasını çıkartmak.
* CP/M Plus ile başlarken.
* Yardım dosyalarını kullanmak.
* Tek ve Çoklu sürücü çalışması.
* PIP ile dosya kopyası.
* BASIC diski ile çalışma.
* AMSTRAD BASIC uygulamaları.
* CP/M Plus uygulaması tanzimi.
* GSX ile başlarken.
* CP/M 2.2 ile çalışma.

Temel kursun. 7. bölümünde boş sistem diskinin nasıl formatlanacağı tarif edilmişti.

Temel kursun 10. bölümünde DISCKIT3 programı ile disklerin tam kopyalarının nasıl yapılacağı gösterilmişti.

Bu bölümde üzerlerindeki programlarla disklerin nasıl kullanılacağı anlatılacaktır.

Esas Diskin Kopyasını Yapmak

Bilgisayarınızla birlikte olması gereken Esas Sistem/Faydalı disklerinizin kopyasını yapmak çok önemlidir; böylelikle tahrip olarak size pahalıya mal olmasını önleyebilirsiniz. Unutmayınız ki size verilen iki diskin toplam 4 yüzü vardır. Her diskin herhangi bir amaçla kullanabileceğiniz iki yüzü vardır.

1. yüz en önemlisidir ve CP/M Plus'un ana kopyasını ve bazı disklerle ilgili faydalı programlar ihtiva eder.
2. Yüzde assembler programcıları için dosyalar vardır.
3. Yüzde ise Dr. LOGO, Yardımcı dosyalar (Help Files) ve GSX vardır.
4. Yüzde CP/M 2.2 ve Dr. LOGO (AMSTRAD Model CPC 664 ve CPC 464 + DDI1 için) bulunur.
Bu programlar ihtiyacınız olabilir diye önceden hazırlanmıştır. Normal olarak kullanılmazlar.

Program kütüphanesi olarak ana disk kopyalarına dikkat etmelisiniz. Normal olarak size gerekli olan programı boş bir diskete kopye edip oradan kullanmak yerine kütüphane diskinizden istediğiniz programı kullanın. Bir kere daha belirtmek isteriz ki ana disklerinizin bir kopyesini mutlaka yapınız (DISCKIT3 ile).

CP/M Plus ile başlarken

CPC 6128 açıldığında AMSTRAD BASIC görüntüsüne alışacaksınız. AMSDOS tarafından BINary program işletilmediği zaman veya CP/M Plus yüklenmediği zaman BASIC engellenemez, çalışmasına devam eder.CP/M Plus bir kere yüklendiği zaman CPC 6128 bir daha 1. yüze danışmaya gerek görmez. Sadece başlama diski sistem diski olmalıdır, diğerleri veri diskleridir ve büyük kayıt kapasiteleri vardır.

Bir programı çalıştırmak için gerekli programı ihtiva eden diski yerleştirip ismini yazmak yeterlidir. Program için kullanılacak veriler program ile aynı diskte veya ayrı bir diskte olabilir. CP/M Plus, AMSDOS'un yaptığı gibi disklerin değiş tokuş yapılmasına izin verir. Birkaç programın uzunlukları sebebiyle bir diskte olması gerekebilir, o zaman 1. yüzdeki PIP adlı programı kullanın (biraz ilerde ve 5. Derste bahsedilecektir.)

Hİ PROFİLE

Sistem diskinizde PROFILE.SUB adlı bir dosya hazırlanmıştır. Bu, CP/M Plus başlatıldığında otomatik olarak işleme konan bir dizi komut listesi içerir. Siz Sistem Diskinin 1. yüzünü yükleyin ve A > çıktığında şunu yazın.

```
REN PROFİLE.SUB = PROFILE.ENG
```

PROFILE.ENG'den PROFILE.SUB dosyasını yaratır. Bu profil gelecek sefer CP/M Plus başlarken etkili olur ve şu komutları içerir:

```
SETKEYS KEYS. CCP
LANGUAGE 3
```

Kürsör tuşlarını, CP/M komutlarının yazılması için, uygun bir konuma ayarlar ve ekran çıkışını USA'DAN UK'ye çevirir; SHIFT 3 "£" sembolu olarak görüntülenir.

SETKEYS KEYS.CCP komutuyla düzenlenen tuştakımı ile BASIC satırları gibi CP/M komut satırları düzenlenebilir. SETKEYS ile ilgili bütün detaylar 5. dersin 2. bölümündedir.

Bir Yardım Eli

Sistem disk paketinizin 3. yüzü "Help" adlı özel bir program içerir. Bu CP/M Plus programları için bir elektronik el kitapçığı olarak tasarlanmıştır. Faaliyete geçirmek için 3. Yüzü sokunuz A > çıkınca şunu yazınız:

```
HELP
```

ve "Help" programı sizin bir çok sualinize cevap vererek, bir yol gösterici olacaktır.

Bir Yahut İki Sürücü?

CP/M Plus ilk defa yüklendiğinde bağlanan disk sürücü sayısını kontrol eder. Bu sayı bağlantı mesajının bir parçası olarak görüntülenir. Bu işlem eğer ikinci disk kısmen sokulmuşsa yanıltıcı olabilir.

Disk mekanizması ile ilgili bütün hata mesajları ekranda daha önceden belirlenmiş 25. satırda gözükür. Programlar ekranda sadece ilk 24 satırı kullanırlar. sizin bir sürücü sisteminiz bile olsa alt satırda "Drive is A:" veya "Drive is B:" mesajını göreceksiniz. Bu CP/M Plus'un size bir mekanizmayla iki taneymiş gibi çalışma imkanı tanımasıdır. Sizin değiştirerek kullanacağınız iki tane diskiniz olabilir, alt satır programın gerektirdiği doğru diski kullanmanızı hatırlatır. Bu tip operasyonlar ikinci bir disk sürücüsü almanızı bertaraf eder fakat oldukça sık disk değiş tokuşu zaman kaybına ve muhtemel hatalara sebep olur.

Diskten diske dosya kopyası

PIP (Peripheral Interchange Program) adlı program bir dosyayı bir diskten diğerine kopyasına imkan verir. Önce PIP programını 1. yüzden yükleyin, A> çıkınca:

```
PIP
```

yazın. Çıkan "*" PIP'in doğru yüklendiğini belirtir. Normalde kaynak diskinden yani A sürücüsünden B sürücüsündeki diske dosyaları kopya edersiniz. Biz tek sürücülü sistemde A ve B sürücüsünü aynı mekanizma olarak gördük.
Bir dosyayı kopya etmek için, örneğin SUBMIT.COM,* göründükten sonra şunu yazın:

```
B: =A:SUBMIT.COM
```

Kaynak diskteki bütün dosyaları kopya etmek için şu komutu kullanın:

```
B:=*.*
```

PIP'ten çıkmak için, * durumundayken RETURN'e basınız.

PIP çok karışık bir programdır ve çalışmasının detayları 5. derste verilecektir.

BASIC için sadece disk

Normal olarak Sistem diskinin CP/M Plus'a başlarken kullanılacak bir ön disk olduğunu gördük. BASIC için kullanılacak disk sadece veri diskleridir ve daha büyük bir kapasiteye sahiptir.

Disk, DISCKIT3 programı ile, mutlaka formatlanmalıdır. Bu tür disklerin üzerine program kopya etmek için PIP veya LOAD ve SAVE kullanılmalıdır.

Açınca Çalışan AMSTRAD BASIC Diskleri

Eğer, CPC 6128 için AMSTRAD BASIC ile yazılmış uygulama programı aldıysanız, açar açmaz çalışmaya hazır olmalıdır. Bu tür disklere "Turn key" (açar açmaz çalışan) denir. Ana sistem disk paketlerinizde olduğu gibi orjinali koruyunuz ve kopyasıyla çalışınız.

Açınca Çalışan CP/M Diskleri

CP/M çalışma sistemi, CP/M'e dayanan kişisel bilgisayarlar için yazılmış çok büyük bir yazılım kütüphanesini yükleyip çalıştırmanıza izin verir. Şimdilik bu programların temel mantığı kullanılmayacaktır. Bütün yapacağınız şey onları 6128'de kullanarak uygun bir diske aktarmaktır.

Özel bir uygulamayı yapmak için bir diskte toplanan bir dizi programa "paket"

denir. Bu paketler değişik bilgisayarlarda kullanılmak için tasarlanmışlardır ve her birinin değişik boyutlarda bir ekranı ve kürsörün dolaşacağı kendine has bir tarzı vardır.

6128, CP/M Plus ile çalışırken, Basic'e dayanan kontrol kodlarından değişik karakterde "terminal emulator" ile daha önceden hazırlanmıştır.

Bazen aldığınız paket programlar AMSTRAD için hazırlanmıştır veya 6128 için uygundur. Eğer mümkünse Zenith Z19/Z29 protokol'lu önceden hazırlanmış bilgileri izleyin. Eğer paket programda bu yoksa veya AMSTRAD esaslı fakat şekli ayrı ise CP/M program şekli bazı komutlar belirtir ki bu komutlar 6128 ekranında paketin gerektirdiği bazı etkiler meydana getirir. Normal olarak paket program için önceden hazırlanmış emirlere uyunuz.

Almış olduğunuz yazılım, sisteme ve diske uygun olmalıdır. Her değişik bilgisayar değişik formda disk kullanır. Bazen aynı büyüklükte olmasına rağmen biri diğeriyle uyum sağlıyor demek değildir. Satıcınıza AMSTRAD 3inch tipini sorunuz.

Açınca çalışan CP/M disk'i yaratmak

Uygulama programının kendisi kadar, "Turnkey" diskte SETKEYS.COM ve belkide SUBMIT.COM (bağlantılı dosyalarla birlikte) olması genellikle faydalıdır. PIP,.COM dosyalarının transferi için kullanılabilir. Bu son durumda PIP basit bir "one-line-at-a-time" (her seferinde bir satır) yazıcıdır. Örneğin 3. Yüzdeki LOGO3.SUB dosyası şu komut ile oluşturulabilir. (Sistem diskinin 1. Yüzünde) Şunu yazın:

```
PIP
```

(Sistem diskini çıkartıp, amaç diskini takın) Şunu yazın:

```
LOGO3.SUB = CON:
SETKEYS KEYS.DRL
[CONTROL] J LOGO 3
[CONTROL] Z
```

CP/M Program Şekillendirilmesi

CPC 6128'in CP/M ile çalışması donatılmış yazılım paketlerine uygun, geniş bir kontrol kodlarına dayanır. Bir çok veri prosesinde ve diğer program paketlerinde, ekranın herhangi bir yerine mesaj yazabilmeye, ekranın herhangi bir yerinden gi-

riş kabul etmeye ve kürsör kontrollerini anlamaya gerek vardır.

Eğer program paketiniz AMSTRAD için hazırlanmışsa bütün bunlarla ilgilenmek zorunda değilsiniz.

Paket Programda Çıkışın Şekillendirilmesi

Bir paketin tertip edilmesi normalde özel bir program ile mümkün olur (INSTAL adı verilir). Eğer Z19/Z29 tip terminal veya CPC6128 değilse 6128'in ekranı ile ilgili parametreler hakkında birçok sual soracaktır. Bu cevaplar Ders 7 Bölüm 15'de bulunabilir.

Kontrol Kodları	Onaltılık	Ondalık	Operasyon
[BEL]	&07	7	Ses dalgası.
[BS]	&08	8	Kürsörü bir geri götürür.
[LF]	&0A	10	Kürsörü bir aşağı indirir.
[CR]	&0D	13	Kürsörü penceredeki sol kenara şimdiki satırın üzerine götürür.
[ESC] A	&1B &41	27 65	Kürsörü bir yukarı çıkartır.
[ESC] C	&1B &43	27 67	Kürsörü bir pozisyon ilerletir.
[ESC] E	&1B &45	27 69	Ekranı temizler.
[ESC] H	&1B &48	27 72	Home kürsör.
[ESC] J	&1B &4A	27 74	Kürsörden ekran sonuna kadar temizler.
[ESC] K	&1B &4B	27 275	Kürsörden ekranın sağ kenarına kadar temizler.
[ESC] L	&1B &4C	27 76	Satır sok.
[ESC] M	&1B &4D	27 77	Satır iptal et.
[ESC] N	&1B &4E	27 78	Kürsör pozisyonundaki karakterleri iptal et.
[ESC] Y	&1B &59	27 89	Kürsörü verilen pozisyonun üzerine götür. (c)
	<c><r>	< c ><r>	kolon +32, (r) sıra +32.
[ESC] d	&1B &64	27 100	Ekranı baştan kürsöre kadar temizle.
[ESC] o	&1B &6F	27 111	Ekranı sol kenardan kürsöre kadar temizle.
[ESC] p	&1B &70	27 112	Inverse video girişi.
[ESC] q	&1B &71	27 113	Inverse video çıkışı.

Paket Programa girişin şekillendirilmesi

Paket program tuş takımı ile yakından ilgilidir. 6128'in tuş takımındaki birçok tuş standart değerler taşır, kürsör tuşu hariç. SETKEYS kullanılarak tuş takımından

üretilen kodlar tekrar tanımlanabilir. Her değişik paketin standart değerler kabul etmesi tercih edilir.

Ne yazıkki farklı yazılım parçaları arasında genel bir anlaşma yoktur (hangi tuş darbesinin hangi kontrol fonksiyonunda kullanılacağı hakkında). Örneğin kürsörü satır başına götürmek:

CP/M komutları [CONTROL] B
Dr. LOGO [CONTROL] A

Tipik bir Wordprocessor için [CONTROL] Q S'e basmak gerekir.
Üç adet tuştakımı kodu standart olarak önceden hazırlanmıştır. Her düzenleme sistem disk paketinin 1. Yüzündeki dosyadan doğrulanabilir. Şöyleki:

SETKEYS KEYS. CCP

PROFILE.SUB tarafından otomatik olarak gönderilen komutlardan bir tanesi CP/M komutları için tuş takımını uygun bir hale sokar.

Makina Açıldığında Çalışmaya Başlayan CP/M Paketi

Normal olarak bütün gereken şey, A > hazırken paket ana programının adını yazmaktır. Örneğin PAYROLL.COM adlı ücret programını çalıştırmak için şunu yazın:

PAYROLL

İşlemesi için herhangi bir düzenleme gerekiyorsa bazen bir SUBmit dosyası önceden hazırlanabilir. Buna örnek 3. Yüzdeki LOGO3.SUB dosyasıdır. Şu komutla yüklenir:

SUBMIT LOGO3

Dosyanın içeriği ise aşağıakini yazarak görülebilir:

TYPE LOGO3.SUB

ve şunu verir:

SETKEYS KEYS.DRL....Tuş takımını tekrar düzenlemek.
LOGO3.....Dr. LOGO programını işletebilmek
SETKEYS KEYS.CCP.....Tuş takımını eski haline koymak.

Makina Açıldığında Çalışmaya Başlayan CP/M Paketinin Otomatik Başlatılması

Başlama sistem diskinden, kısmi bir programın, otomatik olarak işletilmesi, CP/M Plus çalışma sistemi için bir düzenleme yapılarak mümkün olur. Bu, disketi PROFILE.SUB dosyasının sonuna programın adını ilave etmekle gerçekleştirilir.

GSX ile Başlarken

GSX, Grafik Sistem Uzantısı (Graphics System eXtension) demektir. Bu CP/M'in metinde olduğu gibi çıkış grafiklerinin de programlanmasına izin verir. Bu, çubuk ve dairesel grafikler çizmede ayrıca değişik boy ve stillerde başlık yazmada kullanılır. Bir sonraki sayfada bu kolaylığı gösteren gerçek bir çıkışın reprodüksiyonu gösterilmiştir. GSX, ekrana, yazıcıya veya kalemli çiziciye çıkış izni verir.
GSX'in, CP/M'in kelime işlem yapabilmesinden daha fazla resim yapabilme yeteneği yoktur; bir uygulama programı gerektirir. Örnek "DR Graph" digital araştırma programı kullanılarak çizilmiştir. GSX'in yaptığı ise ekran, yazıcı ve çizici için standart kolaylıklara izin vermesi, böyleliklede uygulama programlarını en az şekilde tekrar tanzim edilerek bir makinadan diğerine aktarılabilmesidir.
GSX programı ile işleyebilecek uygun bir disk yapmak için 3. Yüzden GSX.SYS, ASSIGN.SYS dosyalarını ve gerekli cihaz sürücülerini (uygulama programları ile birlikte) boş, formatlanmış diske PIP kullanarak kopye ediniz. ASSIGN.SYS dosyası üç çıkış sürücü cihazı tertibi içerir, büyüklüğe göre ters olarak:

```
21a:ddf×lr7 ; Epson 7 bit yazıcı cı
11a:ddhp 7470 ; kalemli çizici (pen plotter)
 Ø1a:ddmode2 ; 2. ekran modu
 Ø2a:ddmode1 ; 1. ekran modu
 Ø3a:ddmode0 ; Ø. ekran modu
```

Sayılar GSX'e hangi tip sürücü olduğunu söyler (Yazıcı/Çizici/Ekran). Hafızanın aynı alanına normalde sadece bir sürücü yüklenebilir, böylece GSX yeterli boşluk tahsis etmek için önce ne kadar büyük olduğunu bilebilir.
6128'in değişik ekran modları ve standart yazıcılar için sürücü seçimi daha önceden hazırlanmıştır. DRIVERS.GSX dosyası 6128 ile bulunabilenlerin bir özetini içerir. Bu dosyayı 3. Yüzde şunu yazarak okuyabilirsiniz.

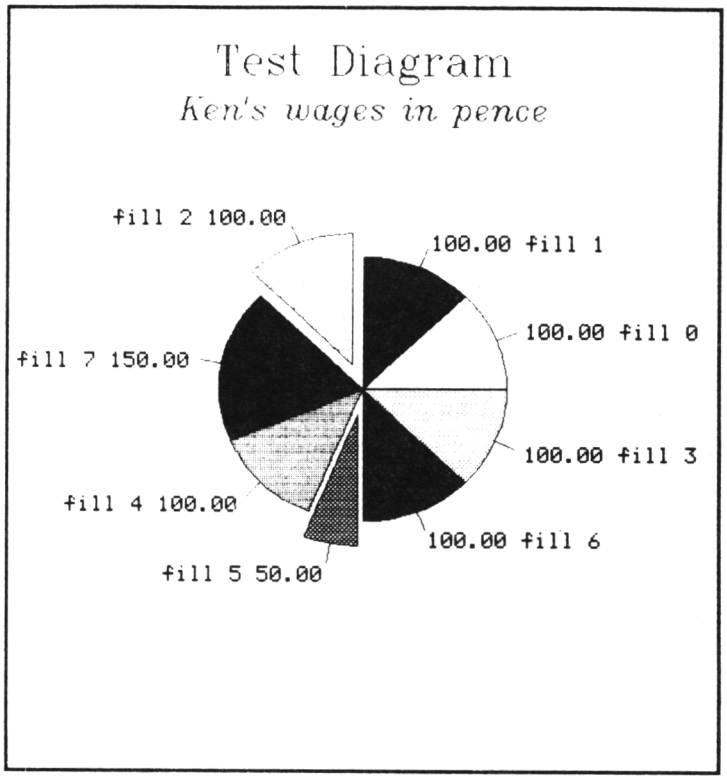
```
TYPE DRIVERS.GSX
```

Böylelikle uygun cihaz sürücüsünü seçebilirsiniz. Birçok uygulama programı GSX yükleyicisini bulundurur. Böylece yapılacak şey programın adını A>'ya yazmak-

tır. Eğer uygulama programınızda GSX yükleyicisi yoksa GENGRAF.COM dosyasını (3. Yüz), GSX çalışma diskinizin üzerine kopya edin ve şunu yazın:

```
GENGRAF YOURFILE
```

GENGRAF.COM, YOURFILE.COM GSP'i ilave ederken silinir.



Yazıcıdan çıkmış bir GSX örneği.

CP/M 2.2 ile Çalışma

CP/M Plus'a ters olarak CP/M 2.2 disklerin değiştirilmesinde çok titizdir. Program çalışmaya başladıktan sonra A>'dan geri dönüldüğünde A sürücüsündeki sistem diskinden çalışma sisteminin bir kısmını tekrar yüklemeniz gerekir. Eğer ikinci bir disk sürücünüz yoksa PIP yerine FILECOPY kullanmalısınız.

Eğer AMSTRAD CPC664 veya CPC464 + DDI1 kullanıyorsanız, yazılımları bir açıdan CP/M Plus ile uygun değildir, CP/M 2.2'yi kullanmanız önerilir.

> Uyarı: Bazı CPC664/CPC464 + DD44 CP/M 2.2 uygulama yazılımları, bazı özel tertipli talimatlar içerir ve CP/M Plus ile çalışmaz. Bu durumlarda sistem disk paketinizin 4. Yüzündeki CP/M 2.2'yi kullanmanız mecburidir.

Bölüm 2: Kasetler

Eğer sisteme bağlanmış kaset ünitesi kullanmak istiyorsanız, bilgisayar ITAPE komutuyla kaset operasyonuna ayarlanınca bir kısım BASIC komutları farklı çalışırlar. Birçok değişik yazılım mesajları ve "prompt"ları (diskle çalışırken görülmeyen) görünür. Eğer kaset üniteniz hızlı ileri ve geri uzaktan kumanda ile çalışıyorsa bu operasyonlar sırasında ya uzaktan kumanda fişini çıkartınız ya da kaset moturunu çalıştırmak için ITAPE:CAT yazınız. ESC tuşuna basarak bu işlemi iptal edebilirsiniz.
Disklere benzemeyen kasetlerdeki dosya isimleri, şekil olarak sıkı kaidelere bağlı değildirler. Uzunlukları 16 karaktere kadar olabilir ve içinde boşluk ile noktalama işaretleri içerebilirler. Bazı durumlarda hepsi ihmal edilmiş olabilir.
Aşağıdaki listede BASIC komutlarının çalışırken olan farklılıkları anlatılmıştır. Komutların tariflerini "AMSTRAD CPC6128 temel kelimelerin komple listesi" adlı derste bulabilirsiniz.

CAT

Aşağıdaki gibi uyarılırsınız:

Press PLAY then any key:

Kaset ünitenizdeki PLAY tuşuna basmalısınız, sonra bilgisayarınızdaki herhangi bir tuşa basınız. Kasetin içindeki bant dönmeye başlar ve bilgisayar kasette bulduğu her dosyanın ismini sırayla ekranda görüntülemeye başlar.

Dosyadaki her kısım görüntülenir ki burada tek bir karakterle ne tip bir dosya olduğu belirtilir. Şöyle ki;

$ korunmaya alınmamış BASIC dosya
% korunmaya alınmış BASIC dosya
* ASCII dosya
& Binary dosya

Eğer dosyayı başarıyla okumuşsa belleğe yüklendiğini belirten:

```
Ok
```

sözcüğünü görüntüler. CAT fonksiyonu bilgisayarın belleğinde bulunan programı etkilenmez.
Eğer kasetteki dosyanın belirli bir adı yoksa CAT şu şekilde görüntü verir:

```
Unnamed file
```

CAT'e ESC tuşuna basarak son verilir.

Okuma Hataları

Eğer yukardaki dosyada okuma mesajları görüntülenmemişse veya aşağıdaki mesajı almışsanız

```
Read error a....veya... Read error b
```

Bu mesaj aşağıdakilerden birini belirtir:

1. Kaset üniteniz bilgisayarın TAPE soketine doğru olarak bağlanmamıştır.
2. Kaset ünitenizdeki VOLUME veya LEVEL kontrolleri doğru olarak ayarlanmamıştır.
3. Bandın kalitesi düşük veya kullanılmaz hale gelmiştir.
4. Bant, hoparlör, televizyon gibi şeylerin yanına konularak manyetik alan etkisinde kalmıştır.
5. Siz AMSTRAD bilgisayar sisteminde kullanılmak için yapılmamış bir kaset ünitesinde çalışıyorsunuz.

CHAIN
CHAIN MERGE
LOAD
MERGE
RUN

İlk uygun dosyanın yüklenmesini istiyorsanız bir dosya adı belirlemenize gerek yoktur. Örnek komutlar:

```
CHAIN ""
CHAIN "",100
CHAIN MERGE ""
CHAIN MERGE "",100
CHAIN MERGE "",100,DELETE 30-70
LOAD ""
LOAD "",&IF4Ø
MERGE ""
RUN ""
```

(Bu komut, CONTROL tuşuna basarken aynı zamanda ENTER tuşuna basarsanız kendi kendine işler.)

Şu şekilde uyarılırsınız:

```
Press PLAY then any key:
```

PLAY tuşuna bastıktan sonra bilgisayardaki tuşlardan birine basarsanız kaset dönmeye ve bilgisayar dosyayı yüklemeye başlar.

Ekranda aşağıdaki yükleme mesajını alırsınız:

```
Loading FILENAME block 1
```

Dosya tamamen yükleninceye kadar kaç tane blok varsa hepsi görüntülenir.

Eğer dosya isminin ilk karakteri ! ise yukardaki mesaj saklanır (bastırılır) ve dosya yüklemesi için herhangi bir tuşa basmanız gerekmez (kaset ünitesindeki PLAY tuşu hariç). Eğer programınız ! işareti kullanıyorsa ve diskte çalıştırılacaksa ! işareti disk operasyonu sırasında gözönüne alınmaz (disk dosya adı okunurken). Kaset veya disk dosya adında ! işareti herhangi bir karakter pozisyonu yerine doldurmaz.

Komutu durdurmak için ESC tuşunu kullanmak ekranda hata mesajı oluşturur. Şöyle ki:

```
Broken in
```

Eğer dosya başarıyla yüklenmemişse bu bölümün "Okuma hataları" adlı kısmına bakınız.

UYARI: CPC464 modeli için ticari yazarlar tarafından kullanılan kasetler CPC6128 + kaset ünitesiyle tam olarak çalışmazlar.

EOF
POS (# 9)

Kasetteki bu fonksiyonlar disk gibi çalışır.

INPUT # 9
LINE INPUT # 9
OPENIN ve CLOSEIN

Eğer uygun bir dosyanın kasetten yüklenmesinin, isim belirtmeden olmasını istiyorsanız aşağıdaki komutu kullanabilirsiniz:

OPENIN ""

Ekranda şu yazı görüntülenir:

Press PLAY then any key:

Teybin PLAY tuşuna basıp bilgisayardaki tuşlardan birine basınca bant dönmeye başlar ve bilgisayar dosyanın ilk 2Kbyte'lık bölümünü belleğin "file buffer" adlı verilen bir bölümüne yükler. Giriş, "file buffer" boş oluncaya kadar alınır ve bilgisayar tekrar hazır duruma geçerek;

Press PLAY then any key

yazar ve tekrar dosyadan 2Kbyte'lık bir bölüm yüklenir. Ekranda aşağıdaki yükleme mesajı görüntülenir:

Loading FILENAME block 1

ve diğer blok numaraları da yüklendikçe görüntülenir. Eğer OPENIN komutundaki dosya isminin ilk karakteri ! işareti ise yukardaki mesajlar bastırılır ve dosyayı yüklemek için herhangi bir tuşa basmanız gerekmez (kaset ünitesinin PLAY tuşu hariç). Eğer programınız ! işareti kullanıyorsa ve diskten işletilecekse ! işareti disk operasyonunda ihmal edilir.

Dosya girişini durdurmak için ESC tuşuna basmak ekranda hata mesajı verir. Şöyleki:

Broken in

Eğer dosya başarıyla yüklenmemişse "Okuma hataları" adlı bölüme bakın.

LIST # 9
OPENOUT ve CLOSEOUT
PRINT # 9
WRITE # 9

Eğer isterseniz, dosya adını belirtmeden, dosyayı saklayabilirsiniz. Örnek komut:

OPENOUT ""

Kasette saklanacak dosyanın ilk 2Kbyte'lik kısmı "file buffer" adı verilen hafızanın bir bölümüne yazılır. "File buffer" dolduğu zaman aşağıdaki şekilde uyarılırsınız.

Press REC and PLAY then any key:

Kaset ünitenizin RECORD ve PLAY tuşlarına ve bunu takiben bilgisayarın herhangibir tuşuna basın. Kaset ünitenizdeki bant dönmeye başlar ve bilgisayarınız "file buffer"daki içeriği saklar. Daha sonra bilgisayar "file buffer"a bir sonraki 2Kbyte'lık kısmını (dosyanın) doldurur ve tekrar hazır ibaresini verir:

Press REC and PLAY then any key:

ve 2Kbyte daha kasete saklar.
Eğer "file buffer" kısmen doluysa ve CLOSEOUT komutu ile karşılaştırılırsa bilgisayar "file buffer'daki kalan içeriği kasete saklarken şu mesajı verir:

Press REC and PLAY then any key:

Ekranda saklandı (saving) mesajı görüntülenir:

Saving FILENAME block < x >

Eğer OPENOUT komutundaki dosya adının ilk karakteri ! işareti ise yukardaki mesajlar bastırılır ve dosyayı saklamak için herhangi bir tuşa basmanız gerekmez (RECORD ve PLAY tuşları hariç). Eğer programlarınız ! işareti kullanılırsa ve diskten işletilecekse ! işareti disk operasyonu sırasında ihmal edilir (disk dosya adı okunurken).

ESC tuşu ile dosya çıkışları durdurulmak istenirse ekranda aşağıdaki hata mesajı görünür:

Broken in

Başarılı Saklama

Dosyanın başarılı saklandığından emin olmak için:

1. Kaset ünitenizin bilgisayardaki TAPE soketine doğru olarak bağlandığını kontrol ediniz.
2. Kaset ünitenizin RECORD veya LEVEL kontrollerinin doğru olarak ayarlandığından emin olunuz.
3. Bozuk kaset veya C120'lik kaset kullanmadığınızı kontrol ediniz.
4. Kasetlerinizin hoparlör, televizyon gibi manyetik alan üreten cihazların yanına konulmadığını kontrol ediniz.
5. Sakladıktan sonra programı bellekten silmeden önce bandın kataloğunu (CAT) isteyip, programın başarılı bir şekilde saklandığını kontrol ediniz.
6. Kaset ünitenizin periodik bakımlarını yaptırıp teyp başlarını (kristal) muntazam temizlediğinizden emin olunuz.

SAVE

Eğer programınızı isim koymadan isimsiz dosya olarak saklamak isterseniz aşağıdaki komutu kullanabilirsiniz:

```
SAVE ""
```

Ekranda aşağıdaki yazıyı göreceksiniz:

```
Press REC and PLAY then any key:
```

gerekli tuşlara basıp bant dönmeye başlayınca bilgisayar programı saklar ve ekranda saklandı mesajını görürsünüz:

```
Saving FILENAME block 1
```

her blok saklandıkça blok numarası değişir.

Eğer dosya adının ilk karakteri ! işareti ise yukardaki mesajlar bastırılır ve saklanacak dosya için herhangi bir tuşa basmanız gerekmez. (RECORD ve PLAY tuşları hariç). Eğer programınız ! işareti kullanıyorsa ve diskte işletilecekse disk operasyonu süresince ! işareti dikkate alınmaz.
ESC tuşu ile komutu etkisiz hale getirmek, ekranda hata mesajı meydana getirir:

```
Broken in
```

"Başarılı saklama" adlı bölümü okuyunuz.

SPEED WRITE

Bu komut sadece kasette çalışır ve bilgisayar disk operasyonuna ayarlandığında çıkartılır.

Hata Mesajları

7,21,24,25,27,32 nolu hata mesajları eğer uygun ise kaset operasyonu sırasında üretilir. (''Bilginiz için'' adlı dersin 6. bölümüne bakınız)

AMSDOS harici komutlar

Kaset'e veya disk'e giriş ve çıkış idaresi şu komutlar tarafından yerine getirilirler:

ITAPE (ITAPE.IN ve ITAPE.OUT diye ikiye ayrılmıştır)
IDISC (IDISC.IN ve IDISC.OUT diye ikiye ayrılmıştır)

Bütün harici komutlar:
IA
IB
ICPM
IDIR
IDRIVE
IERA
IREN
IUSER

Kaset oparesyonu seçilmiş olsada diske uygulanır.

KONU 5
AMSDOS ve CP/M

Bölüm 1: AMSDOS

İlgili konular:

* AMSDOS'a başlangıç
* Disk idaresi
* Disk değiştirmek
* Dosya adları ve tipleri
* AMSDOS başlıkları
* 2 Sürücüde dosya adları
* Rastgele kartlar
* AMSDOS komutlarını kullanan örnek program
* AMSDOS komutlarının özeti
* Dosyaların kopyası ve işletilmesi
* Hata mesajlarının referans rehberi

Başlangıç

AMSDOS, bilgisayarınız ile birlikte verilen AMSTRAD BASIC'ın başına | sembolü gelen harici komutlarla genişletilmiştir.

AMSDOS, herhangi bir dosya kullanımda değilken, kullanıcıya disklerin serbestçe değiştirilebilmesi için izin verir. Eğer açık dosya yazılıyorsa hata mesajı görüntülenir ve verilen data'larda kaybolabilir.

Disk İdaresi

Her diskin bölümü vardır, idare ve veri (data) alanı. İdare, dosya isim listesi ve her bulunan dosyanın nerede olduğunu gösteren harita (map) içerir. AMSDOS veya CP/M bir dosyanın boyutunu rehber kaydını kontrol ederek hesaplar. Diskteki artan yerin miktarını rehberdeki bütün dosyaların birbirine ekleyip kullanılmayan miktarı görerek hesaplar.
Bir dosya okunduğunda disk yeri verilmek için rehber girişi kontrol edilir. Yeni bir dosya yaratıldığında boş alan buna tahsis edilir ve bir dosya silindiğinde alan terkedilir. Büyük dosyalarda her 16K da bir giriş yeri vardır, normal olarak bu gerçek, kullanıcıdan saklanmıştır.

Disk değiştirmek

AMSDOS (ve CP/M Plus) da sürücü okumazken, giriş ve çıkış dosyaları açık değilken disk değiştirilebilir. CP/M 2.2'nin tersine diske "log in" yapmaya gerek yoktur.
Disk, yazılırken değiştirilirse diskin üzerindeki veriler bozulabilir. Eğer disk, üzerindeki dosyalar açıkken değiştirilirse, AMSDOS bunu çok geçmeden sezer sürücüdeki bütün açık dosyalar terkedilir ve bir hata mesajı verilir. Henüz yazdırılan bir veri kaybolur ve en son rehber girişi diske yazılmaz. Bununla beraber AMSDOS, bu değişikliği sadece rehberi okuduğu zaman sezer ki dosyadaki her 16K için bu işi yapar (ve dosya açık veya kapalı olduğu zaman). Böylece muhtemelen 16K'lık veri, diskteki dosyalar açıkken disk değiştirildiğinde, bozulabilir.

AMSDOS dosya isimleri ve dosya tipleri

Disk dosyalarını isimlerken hangi tip olduklarını belirten isim kullanmak standart bir uygulamadır. Bu isimlendirme, bilgisayarı, dosyaların herhangi bir şekilde kullanımı yolunda zorlamaz, bununla beraber bazı programlar ismi doğru tip olan dosyaları kabul eder. AMSDOS herhangi bir isim tipini kabul eder fakat belirlenmediği zaman bir tercih hakkı içinde arar (AMSDOS başlıkları bölümüne bakın).

Dosya Adlarının Yapılması

Dosya adları, nokta ile ayrılan iki kısından meydana gelmiştir. İlk kısım en fazla 8 karakter uzunluğunda ve ikinci kısım 3 karakter uzunluğunda olabilir. Örnek olarak: "ROINTIME.DEM", "DISCKIT3.COM", "DISC.BAS" geçerli dosya isimleridir.
Dosya adlarının ikinci kısmı dosya tipi diye adlandırılır. Dosya adları ve dosya tipleri harflerden ve numaralardan oluşabilir fakat arada boşluk veya noktalama işaretleri olamaz. Bazı konvensiyonel dosya tipleri şunlardır:

- (boşluk) Belirtilmemiş tip. Belki bir OPENOUT "dosya adı" tarafından yaratılan veri veya SAVE "dosya adı", A kullanan AMSDOS tarafından saklanan BASIC programı.
- BAS SAVE "dosya adı", SAVE "dosya adı", P veya SAVE "dosya adı", A kullanan AMSDOS tarafından saklanan BASIC programı.
- BIN SAVE "dosya adı",B, (binary parametre) kullanan AMSDOS tarafından saklanan bellek alanı veya program tipi
- BAK Dosyanın eski halidir ki AMSDOS veya program yeni halini eski adıyla saklamıştır.Bu kullanıcıya eski hali geri izleme olanağı (BACK-up) sağlar.

- COM Komut dosyası. CP/M programları bu tiptir.
- SUB CP/M SUBMIT programı için talimat dosyalarıdır.

AMSDOS Başlıkları

AMSDOS dosyaları uygun bir tip tanıtıcısı ile birlikte otomatik olarak saklar (SAVE), böylelikle daha önceden belirtilmişin üzerine yazmak istenmediği sürece tanımlamak gerekmez. BASIC program dosyaları, korunmuş BASIC program dosyaları ve Binary dosyalar diske başlık kaydıyla saklanırlar. AMSDOS komutu:
LOAD "dosya adı"

bunları tanır ve uygun hareket için alır. Eğer AMSDOS komutu LOAD başlığı bulunamazsa, dosyayı ASCII programı olarak kabul eder.

Bununla beraber AMSDOS'a dosya yüklendiğinde başlığın içeriğinde dosya tipi belirtilmemişse önce dosya tipini araştırır:

- (boşluk)

Eğer mevcut değilse dosya tipini araştırır:

- BAS

Sonra sonuç olarak bir tip:

- BIN

Bu kullanıcıya dosya isminin kısaltılması iznini verir yani dosya tipini belirtmeye gerek yoktur.
Bir disk veri dosyası OPENOUT komutuyla başlatır ve sonra yazılır bunun başlığı yoktur ve içeriği ASCII dir. Örneğin BASIC WRITE, PRINT veya LIST'den basit yazı. OPENIN disk komutu eğer dosya tipi belirlenmemişse aynı LOAD'daki sırayla dosyayı arar.

İki Sürücüde Dosya Adları

İki sürücü sisteminde, yani bilgisayara ilave bir sürücü bağlanmışsa dosya her iki sürücüde de olabilir. Bilgisayar her iki sürücüde de dosyayı otomatik olarak ara-

maz, bu nedenle kullanıcı hangi sürücünün kullanıldığını belirtmek zorundadır. | DRIVE, |B, |A komutlarıyla sürücülerden birini seçebilirsiniz ve normal bir dosya adı kullanılabilir veya daha önceden tayin edilen sürücünün hakkını alarak sürücüyü A: veya B: şeklindeki dosya isminin ön işareti olarak tanımlayabilirsiniz. örneğin:

```
|B
SAVE "PROG.BAS"
|A
```

ve

```
|A
SAVE "B:PROG.BAS"
```

Her ikisi de programı ikinci sürücü, B'de saklar.
Benzer şekilde daha önceden belirlenmiş USER numarası tayinini (USER numarası rehberin içinden ayırım yapmayı mümkün kılar) önemsiz hale getirerek USER numarasını (Ø ile 15 arası) dosya adının ön işareti gibi tanımlayabilirsiniz. Örneğin:

```
LOAD "15:PROG.BAS"
```

ve

```
SAVE "15:PROG.BAS"
```

Daha önce belirlenmiş USER numarası ne olursa olsun, diskin USER numarası 15 olan bölümüne programı yükler ve saklar. (|USER komutuna bakın)
En sonunda daha önceden tespit edilmiş her iki USER ve sürücü ayarlamalarının geçersiz kılarak dosya isminin ön işareti şeklinde beraberce belirleyebilir. Örneğin:

```
RUN "15B:PROG.BAS"
```

Rastgele Kartlar

Çoğunlukla bazı disk operasyonlarının (kopya, silme, vs.) birden fazla disk dosyasında gerçekleştirilmesi gerekebilir. Bir dosya ismi kısmi bir operasyon için tanımlandığında AMSDOS, disk rehberini tam uyacak bir isim için inceden inceye araştırır. Bununla beraber operasyonu dosya gurubunda gerçekleştirmek mümkündür. Burada dosya ismindeki karakterlerden bazısı "önemsiz" olabilir. Bu "önemsiz" pozisyon ? işareti kullanılarak gösterilir. Eğer dosya isminin herhangi bir kısmındaki bütün blok (veya bütün bloğun kalanı) "önemsiz" ise ? bloğu * sembolüne kısaltılır. Örneğin: FRED.*, FRED.???'in kısaltılmışıdır ve F*.BAS, F???????.BAS'ın kısaltılmışıdır.

Sonuç olarak *.* bütün dosyalar demektir.

Örnekler:

REHBER	*.BAS	FRED?.BAS	F*.BA?
BERT.BAS	BERT.BAS		
FRED1.BAS	FRED1.BAS	FRED1.BAS	FRED1.BAS
FRED2.BAS	FRED2.BAS	FRED2.BAS	FRED2.BAS
FRED3.BAK			FRED3.BAK
FRED3.BAS	FRED3.BAS	FRED3.BAS	FRED3.BAS
FINISH.BAS	FINISH.BAS		FINISH.BAS

AMSDOS Komutlarının Programda Kullanılma Özellikleri:

AMSDOS komutlarını daha iyi anlamak için bu dersin ilgili bölümlerindeki örneklerle çalışınız. Bu programları Orijinal Ana CP/M sistem diskiniz takılı iken YAZMAYIN veya İŞLETMEYİN.

Değişken Saklanması ve Ekran Dökümünün Gerçekleştirilmesi:

Aşağıdaki örnek program diske yazar, bu sebeple formatlanmış boş bir disk veya programı işletebilecek bir çalışma diski disket ünitenizde hazır olmalı. Program bir Union Jack flag çizer ve bütün ekranı diske aktarır.

```
10 dumpfile$ = "flag dökümü.srn"
20 MODE 1:BORDER 0
30DIM colour (2)
40 FOR i = 0 TO 2
50 READ colour (i):REM renkleri veri cümlesinden alacak
60 INK i, colour (i)
70 NEXT
80 ON ERROR GOTO 430
90 OPENIN "param.dat"
100 CLOSEIN:ON ERROR GOTO 0
110 IF errnum = 32 AND DERR = 146 THEN CLS: GOTO 160 dosya mevcut değil
120 CURSOR 1: PRINT "Eski dosyanın üstüne yazmak istiyormusunuz? Y/N";
130 a$ = INKEY$:ON INSTR("YN", UPPER$(a$)) GOTO 130, 150, 140:GOTO 130
140 PRINT a$:PRINT "Program terkedildi":END
150 PRINT a$:CURSOR 0
160 OPENOUT "param.dat"
170 WRITE # 9, dumpfile$, 1:REM dosya adını ve mod'u sakla
180 FOR i = 0 TO 2
190 WRITE # 9, colour (i):REM renkleri sakla
200 NEXT i
```

```
210 CLOSEOUT
220 CLS
230 gp = 1:GRAPHICS PEN gp:w = 125
240 x = -65:a = 240:y = 400:b = -150:GOSUB 400
250 y = 0:b = 150:GOSUB 400
260 x = 575:a = -240:y = 400:b = -150:GOSUB 400
270 y = 0:b = 150:GOSUB 400
280 gp = 2:GRAPHICS PEN gp:w = 40
290 a = 240 x = -40:y = 400:b = -150:GOSUB 400
300 x = 0:y = 0:b = 150:GOSUB 400
310 a = -240 x = 640:y = 0:b = 150:GOSUB 400
320 x = 600:y = 400:b = -150:GOSUB 400
330 ORIGIN 0,0,256,380,0,400:CLG 1
340 ORIGIN 0,0,0,640,150,250:CLG 1
350 ORIGIN 0,0,280,352,0,400:CLG 2
360 ORIGIN 0,0,0,640,168,230:CLG 2
370 SAVE dumpfile$,b,&C000,&4000
380 DATA 2,26,6
390 END
400 MOVE x,y:DRAWR a,b:DRAWR w,0:DRAWR -a,-b
410 MOVE x + a/2 + w/2,y + b/2:FILL gp
420 RETURN
430 errnum = ERR:RESUME NEXT
run
```

Dosya tipleri .DAT ve .SRN'nin kullanılmasına dikkat edin. Bu dosya tipleri bize tabiatlarında bulunan önemden değil, dosyada ne olması açısından daima aklımızda olacaktır. PARAM.DAT dosyası ASCII veri dosyasıdır ve başlığı yoktur. FLAG-DUMP.SRN başlıklı bir AMSDOS binary dosyasıdır.

Programın, dosyası yazılmadan önce, hali hazırda mevcut mu diye düşünerek PARAM.DAT dosyasından okumaya çalışmasına dikkat edin. Eğer dosya yoksa BASIC tarafından hata rapor edilir; işlem kesintiye uğramadan program kesintiye uğramadan, program tarafından hata yakalanır. Eğer dosya mevcutsa hiç bir hata mesajı rapor edilmez ve program otomatik olarak "mevcut olan dosyanın üzerine yazmak istiyor musunuz?" diye sorar.

Kısmi ekran dökümü ki, ekran modu, renkler ve hakiki bilgileri içeren dosya isimleri, parametre kütüğünde saklanır. Bu veri dosyasının program değişkenleri (dump file$) ve sabitleri (1) yazmak için ayrıca başka programlarda kullanmak için saklamak üzere nasıl kullanıldığını gösterir.

Ekranı geri yüklemek

Aşağıdaki örnek, parametre dosyası kullanarak, çalışmasını kontrol eden genel amaçlı ekran dökümü görüntüleme programıdır. Dosyanın boyutunun otomatik olarak değişmesine müsaade eder. EOF fonksiyonu ile birlikte veri dosyasından değişkenlerin nasıl girdiklerine dikkat edin. Önemli olan şey bu program tarafından ekran dökümü görüntülenmesi, ekran ile hafızanın bilinen bir pozisyonuna saklanmıştır, aksi halde sonuç çarpık olur. Bu, MODE komutunu icra eden saklama programı tarafından emniyet altına alınır ve sonra ekranın yuvarlanmasına sebep olmamak için dikkatli olunur.

```
10 DIM colour (15): REM 16 renk için hazırlanmak
20 OPENIN "param.dat"
30 INPUT # 9, filename$, screenmode
40 i = 0
50 WHILE NOT EOF
60 INPUT # 9, colour (i)
70 INK i, colour (i)
80 i = i + 1
90 WEND
100 CLOSEIN
110 MODE screenmode:BORDER 0
120 LOAD filename$
run
```

AMSDOS Komutlarının Özeti

| A

| A

KOMUT: Daha önceden tespit edilmiş sürücüyü Sürücü A'ya ayarlar. Eşdeğeri |DRIVE ile A parametresidir.

| B

| B

KOMUT: Daha önceden tespit edilmiş sürücüyü Sürücü B'ye ayarlar. Eşdeğeri |DRIVE ile B parametresidir.

| CPM

| CPM

KOMUT: Sistem diskinden işletim sistemini yükleyerek diğer disk çevresine geçer. Bilgisayar ile birlikte verilen işletim sistemleri CP/M Plus ve CP/M 2.2'dir. Bu komut, CP/M olan disk sürücüsünde yoksa başarısız olur. 1. yüz, CP/M Plus'ı: 4. yüz eski CP/M 2.2 işletim sistemini yükler.

| DIR

| DIR [,<dizi ifadesi>]
| DIR, "*.BAS"

KOMUT: Disk rehberini ve boş alanı görüntüler (CP/M tipinde). Eğer dizi ifadesi konulmamışsa, rastgele kart *.* kabul edilir.

| DISC

| DISC

KOMUT: DISC.IN ve DISC.OUT komutlarına eşdeğerdir.

| DISC.IN

| DISC.IN

KOMUT: Diski dosya giriş vasıtası gibi kullanır.

| DISC.OUT

| DISC.OUT

KOMUT: Diski dosya çıkış vasıtası gibi kullanır.

| DRIVE

| DRIVE, <dizi ifadesi>
| DRIVE, "A"

KOMUT: Belirtilen sürücüyü ayarlar. Bu komut, eğer AMSDOS, istenilen sürücüde diski okuyamazsa başarısız olur.

| ERA

| ERA, <dizi ifadesi>

| ERA, "*.BAK"

KOMUT: Dosya adına uyan ve "sadece oku" (Read/Only) olmayan bütün dosyaları siler. Rastgele kartlara müsaade edilir.

I REN

I REN, <dizi ifadesi>, <dizi ifadesi>
I REN, "NEWNAME.BAS", "OLDNAME.BAS"

KOMUT: Dosyaya yeni bir isim verir. Seçilen yeni isimli başka bir dosya mevcut değildir. Rastgele kartlara müsaade edilmez. USER (I USER'e bakınız) parametresi < dizi ifadesi >'nde herhangi bir önceden belirlenmiş ayarlamanın üzerine yazmak için belirtilebilir. Örneğin, I REN, "0:NEW.BAS", "15:OLD.BAS" komutu USER 15'deki "OLD.BAS" adlı dosyaya USER 0'daki "NEW.BAS" yeni adını verir ki I USER'ın daha önceki belirlenmiş veya yazılmış ayarlamaları göz önüne alınmaz

I TAPE

I TAPE

KOMUT: I TAPE.IN ve I TAPE.OUT ile eşdeğerdir. Eğer harici kaset ünitesi bağlı ise kullanılır.

I TAPE.IN

I TAPE.IN

KOMUT: Bandı dosya giriş vasıtası gibi kullanır. Eğer harici bir kaset ünitesi bağlıysa kullanılır.

I TAPE.OUT

I TAPE.OUT

KOMUT: Bandı dosya çıkış vasıtası gibi kullanır. Eğer harici bir kaset ünitesi bağlıysa kullanılır.

I USER

I USER, <tam sayı ifadesi>
I USER, 3

KOMUT: Rehberin 16 bölümünde hangisine disk fonksiyonlarının (CAT, LOAD, DIR, vs.) uygulanacağını hesaplar.

REN komutuyla bir dosyadaki USER numarası diğerine transfer olabilir. Örneğin: | REN, "15: ÖRNEK.BAS", "Ø:ÖRNEK.BAS"; dosyayı Ø USER numarasından USER 15'e dosyanın kendi adını değiştirmeden (ÖRNEK.BAS) transfer eder.

Disk'den Diske Dosya Kopyası

Başlıklı AMSDOS dosyaları

CP/M çevresindeki bu tip dosyaları PIP kullanarak (bu dersin 2. bölümüne bakınız) kopye etmek mümkündür. Herhangi bir dosya (AMSDOS tarafından oluşturulmuş ve başlık kaydı bulunan) diskten diske tam olarak kopya edilebilir, fakat genel olarak dosya içeriği CP/M programları tarafından anlaşılmaz.

ASCII dosyaları

AMSDOS tarafından oluşturulmuş başlıksız dosyalar ASCII'dir ve CP/M programları tarafından hem anlaşılabilir hem de kopya edilebilir. Kısmi olarak ASCII program dosyaları, ASCII data dosyaları ve ASCII yazı dosyaları AMSDOS ve CP/M programları arasında değiştirilebilirler.

Read/Only (Sadece/Oku) dosyalar

CP/M, kullanılarak herhangi bir kütüğü Read/Only'ye ayarlar ve/veya rehberdeki özel bir sistem statüsüne ayarlar. Bu vasıflar sadece CP/M çerçevesinde set ve reset olabilirler. Daha fazla bilgi için dersin 2. Bölümüne bakınız (SET).

Disk ve kaset arasında dosyaların kopyası

CP/M Plus'da bant dosyalarına giriş ve diske kopya etmek için veya diskten bant dosyalarının yapılması ile ilgili bir destekleme yoktur. Eğer dosyaların bu şekilde kopya edilmesi isteniyorsa CP/M 2.2 kullanılmalıdır. CP/M 2.2 bu amaçla CLOAD ve CSAVE yüklemelerini içerir ve tabloda gösterildiği gibi kullanılır.

Dosya Kopya Yöntemi

Tabloda disk ile bant arasında kopyalanacak her çeşit dosya kopya etme gösterilmiştir. İlave disk sürücüsü olmadığı kabul edilmiştir. Korunmuş BASIC programları ve binary dosyalar (makine kodlu video oyunları) yükleme adresleri bilinmediği takdirde teybe veya teypten kopyaları imkansızdır. PIP, CLOAD, ve CSAVE programlarıyla ilgili detayları bu dersin 2. bölümünde bulabilirsiniz.

Kopya tabloları için bundan sonraki iki sayfaya bakınız.

COPY TO:	COPY FROM: *AMSTRAD BASIC on tape**	*ASCII data on tape**	*Binary on tape**
AMSTRAD BASIC on tape*	|TAPE LOAD"FILE" ‹change tapes› SAVE"FILE" |DISC		
Binary on tape*			H=HIMEM |TAPE MEMORY ‹s›-1 LOAD"FILE" ‹change tapes› SAVE"FILE",B ‹s›,‹l›[,‹r›] |DISC MEMORY H ‹note 2›
ASCII on tape*	|TAPE LOAD"FILE" ‹change tapes› SAVE"FILE",A |DISC	‹Insert CP/M 2.2 disc› |CPM CLOAD"FILE",TEMP ‹change tapes› CSAVE TEMP,"FILE" ERA TEMP AMSDOS ‹note 1›	
AMSTRAD BASIC on disc*	|TAPE LOAD"FILE" |DISC SAVE"FILE"		
ASCII on disc	|TAPE LOAD"FILE" |DISC SAVE"FILE",A	‹Insert CP/M 2.2 disc› |CPM CLOAD"FILE" AMSDOS	
Binary on disc*			H=HIMEM |TAPE MEMORY ‹s›-1 LOAD"FILE" |DISC SAVE"FILE",B ‹s›,‹l›[,‹r›] MEMORY H ‹note 2›

* lı dosyaların başlığı vardır.

< note 1 > Geçici dosya "TEMP" için boş disk alanı gerekir.
< note 2 > < s > Dosyanın başlama adresidir, (l) uzunluğu, (r) seçmeli işletme adresidir.

COPY TO:	COPY FROM: *AMSTRAD BASIC on disc **	*ASCII data on disc*	*AMSDOS Binary on disc **	*All other on disc*
AMSTRAD BASIC on tape *	LOAD "FILE" |TAPE SAVE "FILE" |DISC			
Binary on tape *			H=HIMEM MEMORY ‹s›-1 LOAD "FILE" |TAPE SAVE "FILE",B, ‹s›,‹l›[,‹r›] |DISC MEMORY H ‹note 2›	
ASCII on tape *	LOAD "FILE" |TAPE SAVE "FILE",A |DISC -or- ‹Insert CP/M 2.2 disc› |CPM CSAVE FILE AMSDOS	‹Insert CP/M 2.2 disc› |CPM CSAVE FILE AMSDOS		‹Insert CP/M 2.2 disc› |CPM CSAVE FILE AMSDOS ‹note 3›
AMSTRAD BASIC on disc *	LOAD "FILE" ‹change discs› SAVE "FILE" - or - ‹Insert CP/M Plus disc› |CPM PIP B:=FILE AMSDOS			
ASCII on disc	LOAD "FILE" ‹change discs› SAVE "FILE",A	‹Insert CP/M Plus disc› |CPM PIP B:=FILE AMSDOS		
AMSDOS Binary on disc *			‹Insert CP/M Plus disc› |CPM PIP B:=FILE AMSDOS	
All other on disc				‹Insert CP/M Plus disc› |CPM PIP B:=FILE AMSDOS

* lı dosyanın başlığı vardır.

< note 2> < s > Dosya başlama adresi, (l) uzunluk, (r) seçmeli işletme adresidir.

< note3> Hedef dosyası BASIC tarafından direkt olarak kullanılamaz. Bununla beraber bu tercih ucuz taşıma veya geri sürme vasıtaları için faydalıdır. Dosya diske geri kopya edilmek için CLOAD "FILE" kulanılır (CP/M 2.2 altında)

AMSDOS Hata Mesajları Rehberi

AMSDOS taşıyamadığı komutlarla karşılaştığı zaman bir hata mesajı görüntüler. Eğer donanım ile ilgili bir problem varsa hata mesajı aşağıdaki şekildedir:

Retry, Ignore Or Cancel?

R, kullanıcının bazı önleyici çalışma yaptıktan sonra işlemin tekrarlanmasına neden olur.

I, beklenmedik ve uygunsuz neticelerden kaynaklanan problem hiç olmamış gibi bilgisayarı devam ettirir.

C, daha ilerde olan hata mesajından dolayı çalışmayı iptal eder.

Hata Mesajlarının Anlamları

Unknown command

.... Doğru ifade edilmemiş komut

Bad command

....Komut herhangi bir sebepten yapılamamıştır. Yazılım hatası veya uygun olmayan donanım şekli.

< dosya adı > already exists

....Kullanıcı kullanımda olan bir dosyaya tekrar isim vermeye çalışıyor.
<dosya adı > not found

....dosya mevcut değil.

Drive (sürücü): directory full

....Yeni bir giriş için disk rehberinde yer yok.

Drive < sürücü >: disc full

....Diskte yeni veri için yer yok.
Drive<sürücü> : disc changed, closing <dosya adı>

....Üzerindeki dosyalar açık iken disk değiştirilmiş.
<dosya adı> is read only

....Dosya çalışmaz çünkü Salt/Oku (Read/Only). Dosyalar CP/M çevresinde sadece Read/Only veya Read/Write (Oku/Yaz) olarak ayarlanabilir.
Drive <sürücü>: disc is write protected

....Yazım koruma deliği açık olan bir diske yazılmaya kalkışılmış. Diski kullanmak için çıkarınız, yazım koruma deliğini kapatın, tekrar sokun ve R yazın.
Drive <sürücü>: read fail

....Diske yazılırken donanım hatası. Tavsiye edilen hareket diski çıkartıp tekrar sokun ve R yazın.
Failed to load CP/M

....CPM komutu sırasında CP/M yüklenmesinde okuma hatası veya CP/M içeren geçerli sistem diskini kullanmadınız. Data format diskinden CP/M'i yüklemeye çalışırsanız okuma hatası (read fail) alırsınız.

Bölüm 2: CP/M

CP/M Plus

* CP/M'in tanıtılması
* CP/M Plus'ın yararlılığı
* Direkt Konsol mod
* Geçici hal programları (Transient)
* Kullanma çevresi
* CP/M 2.2 ile çalışma

CP/M Plus bir disk çalışma sistemidir. Bu özel program CPC6128'in tam gücüne ulaşmanızı sağlar. RAM'ın 128K'sının tamamında kullanılır ki 61K kullanıcı programlar için geçerlidir. CP/M data dosyalarına rastgele ulaşır ve CPC6128 tamamlayıcısı karmaşık VDU benzeticisini de içerir.

CP/M' çok değişik bilgisayarlar için mevcut olduğundan dolayı seçebileceğiniz binlerce uygulama paketleri vardır ve bütün bunlar size sıhhatli bir bilgi ve tecrübe kazandıracaktır.

CP/M'in kendi programlarınızı nasıl yazacağınızı içeren tüm detayları ve CP/M tamamlayıcısı AMSTRAD hakkındaki bilgileri SOFT971-CP/M Plus Guide'de bulabilirsiniz.

Tanıtım

CP/M çalışma sistemi, bilgisayarlarla iletişim kurmanızda ve dosyalar ile çevre elemanlarıyla çalışmanızda yol gösterir. Özel komutlar, (diskteki faydalı programlar) uygulama programlarınızı verilerle çalıştırırken esas amacınıza yardım eder.

BASIC'in direkt modu ve "Ready" hazır işareti olduğu gibi CP/M'in de direkt konsol modu ve A> veya B > hazır işareti vardır. Belirli built-in komutları mevcuttur fakat "housekeeping" işlerinin çoğunluğu "geçici programların" yüklenmesi ve işletilmesi için yapılır. "Geçici" diye adlandırılmalarının sebebi bunların sadece bilgisayarın içinde olmalarındandır ve kullanılırken built-in olmaya karşı koymalarındandır.

Standart CP/M hata mesajlarının yanı sıra ekranın en alt satırında hemen ayırdedilebilen birçok özel donanım hata mesajları vardır.

Diskteki CP/M Plus

CP/M'in en önemli kısmı ".EMS" dosya tipindeki özel bir dosyada bulunur ve Sistem Disk Paketinin 1. yüzündedir. Bilgisayar CP/M'i bu dosyadan yükler ve bu yüklemeyi iki aşama işlemi kullanarak yapar.

Başlangıçta I CPM, AMSDOS komutu Ø kanalının ilk bölümünü yükler. Bu bölüm sistem diski üzerinde bir program olarak ayarlandıktan sonra .EMS dosyasını belleğe yükler. Sistem kanalının geri kalanı kullanılmaz.

Profile Dosyası ile Başlama

Yükleme işlemi sırasında CP/M ilk harekete geçirildiğinde eğer diskte PROFILE.SUB varsa, bu dosyadaki talimatlar teslim edilir. Buda serbestlik, tuş takımının tekrar tanziminde, ekran çıkışının donatımında, yazıcının koşullanmasında ve hatta uygulama programının otomatik başlamasında kullanılır. 4. Derste profile dosyalarının tekrar isimlendirilmesini görmüştük.

Profile dosyası çalışırken diskte küçük bir geçici dosya write-enabled olsun diye açılır. Bu nedenledir ki ana disk tanımlanabilir profile dosyasını kabul etmez.

Profil dosyalar, kelime işlemciler, yazı düzenleyiciler (ED.COM gibi), hatta BASIC'le bile kurulabilir. Aşağıdaki BASIC program PROFILE.SUB yapmakta kullanılır.

```
1Ø OPENOUT "PROFILE.SUB"
2Ø PRINT # 9, "SETKEYS KEYS.CCP"
3Ø PRINT # 9, "LANGUAGE 3"
4Ø CLOSEOUT
```

Konsol Kontrol Kodları

CP/M çevresinde özel tuş operasyonlarının çeşitleri kullanılır. Bu tuş vuruşları AMSTRAD BASIC'deki kursör ve ESC tuşlarının işlemlerinin yerini alır. Aşağıdaki kontrol kodları şu komutu işlettikten sonra düzenlenir:

```
SETKEYS KEYS.CCP
```

Geçiş programı SETKEYS.COM ve komut dosyası KEYS.CCP sistem disk paketinizin 1. yüzünde bulunurlar.

[CONTROL] A ←	Kursörü bir geri götürür.
[CONTROL] B [CONTROL] ←	Kürsörü satır başına, başındaysa sonuna götürür.
veya	
CONTROL →	
[CONTROL] C [CONTROL] [ESC]	Terk etme
[CONTROL] E [CONTROL] [RETURN]	Fiziksel carriage return.
[CONTROL] F →	Kürsörü bir ileri götürür.
[CONTROL] G [CLR]	Kürsörün altındakini siler.
[CONTROL] H [DEL]	Arka tarafı siler.
[CONTROL] I [TAB]	Kürsörü bir sonraki tab duruşuna gönderir.
[CONTROL] J	Komut satırını gönderir.
[CONTROL] K [CONTROL] [CLR]	Satırın sonunu siler.
[CONTROL] M [RETURN] veya [ENTER]	Komut satırını gönderir.
[CONTROL] P	Ekranın yazıcıya çıkışının kontrolu (on/off)
[CONTROL] Q	Ekran çıkışını tekrar başlatır.
[CONTROL] R [CONTROL] [ENTER]	Komut satırını tekrar yazar.
[CONTROL] S [ESC]	CP/M'den gelen ekran çıkışını durdurur. Başlatmak için CONTROL Q
[CONTROL] U	Satır çıkartır (atar).
[CONTROL] W [COPY]	En son yazılan komut satırlarını tekrar çağırır.
[CONTROL] X [CONTROL] [DEL]	Satırın başından kürsöre kadar olan bölümünü siler.
[CONTROL] Z	Yazının (metnin) sonu.

Dosya Adları

Birçok komut, dosya adlarını parametre olarak alır ve belirtildikleri zaman dosya adları rastgele kartları (wild-cards) içerir. Bütün dosya adları büyük harf olur Direkt konsol komutları ve birçok işletme programları için dosya adları gerekmez. Dosya adları A: veya B: ön işaretlerine sahiptir ki CP/M'i uygun sürücüyü kullanması için zorlasın.

Bu sebepten dolayı tipik bir CP/M komutu şöyledir:
TYPE KEYS.CCP

Burada TYPE gereken fonksiyondur, anlamı "ekranda görütüle" ve KEYS.CCP görmek istediğimiz dosya adıdır.

Daha önceden belirlenmiş sürücüyü anahtarlamak

Eğer sizin ilave bir disk sürüeünüz bağlıysa, önceden ayarlanmış sürücü seçimini A: veya B: yazarak A ve B sürücülerine ayarlayabilirsiniz. > işaretinin önündeki harf size halihazırda belirlenmiş sürücüyü gösterir. A: ve B: ön işaretlerini dosya adlarına vermek önemini yitirir fakat önceden belirlenmiş sürücüyü yeniden ayarlamaz.

Direkt Konsol Komutları

A > veya B > ile yazılabilen bir çok Direkt Konsol Komutları vardır. Her komut kısaltılabilir ve her ne kadar aşağıda bahsedilen basit fonksiyonlar yapılmadıysa da, aynı isimle çok sunileşmiş geçiş komutları vardır.

DIR komutu

DIR, diskin rehberini listeler. Dosya isimleri belirli bir sırasıyla sıralanmazlar fakat DIR görüntülenmesinde dosya isimlerinin pozisyonları bu dosyanın disk rehberindeki giriş yerini belirler. Rastgele kartlara müsaade edilir. "SYS" vasfıyla ayarlanmış dosyalar listelenmez.

DIR	Önceden belirtilmiş diskteki dosyaları listeler.
DIR B:	B sürücüsündeki dosyaları listeler.
DIR *.BAS	.BAS tipi dosyaları listeler.
DIR B:*.BAS	B sürücüsündeki .BAS tipi dosyaları listeler.
DIR PIP.COM	PIP.COM dosyasını listeler (eğer varsa)

DIRSYS veya DIRS komutu

DIRSYS veya DIRS, "SYS" vasfıyla ayarlanmış rehber dizisini listeler. Aksi takdirde DIR gibi çalışır. SYS vasfı sonra anlatılacaktır.

ERASE veya ERA komutu

ERA rehberden dosyaları silmekte kullanılır. Sadece rehber girişi silinir böylece veri, yeni başka bir dosya tarafından kullanılıncaya kadar diskin veri bölümünde kalır; bununla beraber bilgi tekrar kazanılamaz. Rastgele kart dosya isimlerine müsaade edilir ve eğer kullanılırsa ERA tasdik için sorar. ERA silinen dosya isimlerini listelemez. Eğer silinecek herhangi bir dosya Read/Only olarak bulunursa komut durur. Read/Only vasfı daha sonra anlatılacaktır.

ERA PIP.COM	PIP.COM dosyasını siler.
ERA B:PIP.COM	Sürücü B'deki PIP.COM dosyasını siler.
ERA *.BAS	Bütün .BAS dosyalarını siler.

RENAME veya REN komutu

REN, halihazırda bulunan dosyanın yeniden adlandırılmasını sağlar. Önce yeni dosya adı konur, = işaretinden sonra mevcut bulunan dosya adı yazılır. Eğer yeni dosya adı mevcut ise hata mesajı görüntülenir. Rastgele kartlar, yapma komutu olarak REN'in sahasının dışında kalır ve geçiş programı RENAME.COM gerektirir.

REN NEWNAME.BAS = OLDNAME.BAS, OLDNAME.BAS adını NEWNAME olarak değiştirir.

REN B.NEWNAME.BAS = OLDNAME.BAS Sürücü B'deki OLDNAME.BAS adını NEWNAME.BAS olarak değiştirir.

TYPE veya TYP komutu

TYPE belirtilmiş bir dosyanın ekrana yazılacağını sorar. Eğer dosya ASCII yazı dosyası değilse önceden tayin edilemeyen ve muhtemelen istenmeyen yan etkiler olur.

```
TYPE KEYS.CCP
```

KEYS.CCP dosyasını görüntüler.

USER veya USE komutu

USER, halihazırda kullanılan kullanıcı numarasını değiştirir. CP/M Plus kullanılan kullanıcı numarasını Ø'a ayarlayarak başlar. Normal olarak dosyaları kullanıcı numarasıyla tanımlayarak girilebilir, böylece rehberin haricinde taksimatı önceden hazırlar.

Kullanıcı Ø'daki dosya (SYS sıfatıyla) bütün kullanıcı numaralarından girilebilir. Bu, her kullanıcı alanında gerçek kopyalarını yapmadan bütün kullanıcı numaralarına işlem ve uygulama programları yapmak için çok güçlü bir kolaylıktır.

```
USER 3
```

komutu kullanıcı numarasını 3'e ayarlar.

Geçiş Komutları

Direkt Konsol modda izin verilenden daha karmaşık bir dosya idaresi yapmak için önceden hazırlanmış değişik faydalı programları uygulamanız gerekir. Bunlar program adını yazmakla (muhtemelem peşinden dosya adı ve/veya bazı parametreler gelir) çağırılırlar. Siz belkide DISCKIT3 kullanacaksınız.

Komutlar aşağıda belirtilen kategorilerin sayısındadır. Bu programların tüm dökümentasyonu kapsamlıdır ve daha ileri bilgiler HELP dosyalarında (Sistem disk paketinin 3. yüzünde) ve SOFT971-CP/M Guide de vardır.

DISCKIT3, SETKEYS, SETLST, SETSIO, PALETTE, LANGUAGE, ve AMSDOS komutları, GSX Ekran cihazları sürücüleri ayrıca LOGO3 tertibi AMSTRAD tarafından tasarlanmışlardır ve yalnız AMSTRAD sisteminde çalışır. Diğer herhangi bir CP/M sisteminde çalışmazlar.

Bir satırda çoklu komutlar girilebilir ki komutlar ünlem işaretiyle biribirlerinden ayrılırlar. Örneğin:

```
LANGUAGE 3!SETKEYS KEYS.WP
```

Çevre İdaresi

DISCKIT3 komple disk formatlayıcısı, kopyacı ve kontrolcudur. Kopya ederken formatlaması, formatlamadan ve daha sonra kopya etmekten daha çabuktur. Geniş kapsamlı seçenek tablosu hangi tuşlara basılacağını (genellikle fonksiyon tuşları) belirtir. Vendor format diskleri sistem diskinin, yazılım dağılımı maksadıyla hazırlanmış özel bir formdur. Data format diskleri CP/M Plus çevresinde bir aktivite için daha uygundur.

Lisan Karakterleri

CPC6128 uluslararası karakterlerin hepsine sahiptir. LANGUAGE komutu bu karakterlerden bazılarını değiştirir böylelikle basit yazılım, değiştirilen ve aksanlı karakterleri ekranda görüntüleyebilir. Bu konuda daha fazla bilgiyi "Bilginiz için" adlı konunun 16. bölümünde bulabilirsiniz.

Komut:
LANGUAGE 3

6128'i İngiliz karakterlerine (# ve £ işaretleri) ayarlar.

Renkler

6128'de CP/M Plus ile önceden tespit edilmiş renkler mavi zemin üzerinde parlak beyaz karakterlerdir. Bu renkler PALETTE komutuyla değiştirilebilir. Her ink için parametre numarası vardır, şöyleki: ink Ø arka zemin ve çerçeve alanı, ink 1 ise yazı içindir. Her renk Ø'dan 63'e kadar bir sayı ile temsil edilir, yeşil monitörde renk yoğunluğu (parlaklık) ölçüsü için kullanılabilir. Ink numarasını 1'den 16'ya kadar belirtmek mümkündür, bununla beraber 80 kolonluk modda sadece ilk ikisi görünebilir.

Komut:
PALETTE 63,1

Normal ayarlamaları tersine çevirir yani ink 0 ve ink 1 değiştirilerek arka zemini parlak beyaz (63) yazıyı mavi (1) yapar.

Aşağıaki tabloyu kullanarak sizin için gerekli olan renkleri (veya yoğunluğu) seçebilirsiniz. Ondalık veya hex (16'lık) gösterimlerin birini kullanabilirsiniz.

Renk	Hex	Ondalık	Renk	Hex	Ondalık
Siyah	&ØØ	Ø	Pastel mavi	&2B	43
Mavi	&Ø2	2	Portakal	&2C	44
Parlak mavi	&Ø3	3	Pembe	&2E	46
Kırmızı	&Ø8	8	Pastel magenta	&2F	47
Magenta	&ØA	10	Parlak yeşil	&30	48
Leylak	&ØB	11	Deniz yeşili	&32	5Ø
Parlak kırmızı	&ØC	12	Parlak açık mavi	&33	51
Mor	&ØE	14	Limon yeşili	&38	56
Parlak magenta	&ØF	15	Pastel yeşil	&3A	58
Yeşil	&2Ø	32	Pastel açık mavi	&3B	59
Açık mavi	&22	34	Parlak sarı	&3C	60
Gök mavisi	&23	35	Pastel sarı	&3E	62
Sarı	&28	40	Parlak beyaz	&3F	63
Beyaz	&2A	42			

Tuş Takımı

Tuş takımı tarafından üretilen kodlar SETKEYS komutuyla değiştirilebilir. Bu tuşlara uygun kodların tayinine ve genişlemesine imkan verir. Hakiki kodlar bir dosyaya yazılmalıdır. Komut dosyası, yazı editörü, PIP veya BASIC'den yapılabilir. Örneğin:

```
SETKEYS KEYS.TST.
```

KEYS.TIST dosyası şunları içerir:

```
E &8C "DR ↑ M" expension token 12
8 NSC " ↑ H" backspace = [CONTROL] H, ASCII Ø8
```

önce [CONTROL] [ENTER] expension token (&SC ile gösterilen) DIR [RETURN] dizisi olarak tekrar tarif edilir ve sonra kürsör sola tuşu ← (tuş no:8) backspace'e döner.

Standart dosyalar 6128 ile önceden hazırlanmıştır. CP/M düzeltme komutu için KEYS.CCP; Dr.LOGO ile kullanmak için KEYS.DRL ve birçok kelime işleticileri için (word processor) KEYS.WP uygundur.

Yazıcılar

Yazıcıların başlangıcındaki durumu aşağıdaki komut ile oluşturabilirsiniz.

SETLST <dosya adı>

<Dosya adı>, yazıcıya gönderilecek dizi veya dizileri içerir. SETKEYS komut dosyasında olduğu gibi, kontrol kodları şu şekilde gösterilirler.

↑<karakter>

veya

↑ '<karakter değeri>'

veya

↑ '<kontrol kod adı>'

Kontrol kod adları ESC, FF vs.dir "Bilginiz için" adlı konunun ASCII tablosunda gösterilmiştir.

Birçok yazıcı için kullanışlı bir başlangıç durum kod değeri 15'dir ve bu da yoğun baskıdır (condenset printing).

Komut:
```
PRINT # 8,CHR$(15)
```

BASIC de ayarlanır. CP/M için ise komut:
```
SETLST CONDENSE
```

CONDENSE dosyası tek bir satır halinde aşağıdakilerden birini içerir:

```
↑ 'SI'
↑ 0
↑ '&F
↑ '15'
```

hepside ondalık 15 değerini gösterir.

Bazı uygulama programları ekranı 24×8Ø olarak ister. Komut SET24×8Ø ekran boyutunu ayarlamak için kullanılır.

Komut:
```
SET24×8Ø
```

veya

```
SET24×8Ø ON
```

24 × 8Ø modunu açar, ve
SET24 × 80 OFF

24 × 80 kodunu kapar.

6128'in normal ekran boyutu 24 × 8Ø'dir ve alt satır durum mesajları için ayrılmıştır. 24 × 8Ø modunu kapatmak sadece durum satırı ihmal edilirse dikkate değer. Ders 7 bölüm 15'de durum satırlarının nasıl açılıp, kapatılacağı detaylı olarak anlatılmıştır.

Seri Interface

CP/M Plus tek kanal seri giriş/çıkış interface için (RS232) yapılmıştır. SETSIO komutu (parametresiz) yazılarak önemli istatistikler incelenebilir.
SETSIO

veya

SETSIO, RX 1200, TX 75, PARITY NONE, STOP 1, BITS 8, HANDSHAKE ON, XOFF OFF

yeni bir konfigürasyon ayarlar.

Baud hızı ve XON/XOFF durumları DEVICE komutu ile mümkün olan ayarlamalardan etkilenir. DEVICE mantıksal ve fiziksel araçlarla ilgilenir. Mantıksal araçlar: iki nokta üstüste ile belirtilirler. Bütün araç sıfatlarını incelemek için şunu yazın: ve sıfatlar komutlarla değiştirilebilirler şöyle ki:

DEVICE SIO [1200]	SIO'yu 1200 baud'a ayarlar.
DEVICE SIO [XON]	SIO'yu XON/XOFF protokoluna açar.
DEVICE SIO [NOXON]	SIO'yu XON/XOFF protokoluna kapar.

Mantıksal ve fiziksel araçların arasındaki bağlantılar değiştirilebilir. Normal olarak CON: CRT'ye (tuştakımı/ekran); AUX: SIO'ya (seçmeli seri interface) ve LST:, LPT'ye (Centronics yazıcı interface) ayarlar. Komut:

DEVICE LST: = SIO

Eğer uygunsa, Yazıcı çıkışını seri interface'e gönderir.

Bu kanal tekrar yönlendirilmesidir ve PIP ile hazırlanan dosya kopyalama ile karıştırmamak gerekir. GET < dosya adı > ve PUT <dosya adı > komutları, araç kanalından ziyade dosya kullanmalarını bildirerek konsol giriş veya çıkışlarını ve yazıcı çıkışını yeniden yönlendirir.

PIP

PIP bilgileri bilgisayar ile çevresi arasında transferi sağlar. Komutun genel yapısı:

PIP <varış yeri>=< kaynak >

< kaynak > ve < varış yeri > kaynakta müsaade edilen rastgele kartlar ile bir dosya adı veya bir mantıksal araç olabilir. Aşağıdaki şu mantıksal araçlar kullanılabilir.

Kaynak olarak	Varış yeri olarak
CON: konsol girişi	CON: konsol çıkışı
AUX: auxiliary girişi	AUX: auxiliary çıkışı
EOF: dosya sonu işareti	LST: yazıcı
	PRN: tab genişlemesi, satır numarası ve sayfa kırılması ilave edilmiş yazıcı.

PIP için örnekler:

PIP B:=:*.COM

Sürücü A'daki bütün *.COM dosyalarını sürücü B'ye kopya eder.

PIP KEYBOARD.CPM = KEYS.CCP

KEYBOARD.CPM'den çağırılan KEYS.CCP'nin kopyasını yapar.

PIP CON:=KEYS.CCP

KEYS.CCP dosyalarını ekrana gönderir. (=TYPE KEYS.CCP)

PIP LST:=KEYS.CCP

KEYS.CCP dosyalarını yazıcıya gönderir.

PIP TYPEIN.TXT=CON:

TYPEIN. TXT adlı dosyaya tuştakımı girişlerini kabul eder.

En son operasyon [CONTROL] Z kontrol kodu ile biter ve yeni bir satır için her [RETURN]'den sonra [CONTROL] J yazmanız gerekir. [CONTROL] J satır beslemenin ASCII kodudur.

PIP'i parametresiz yazarsanız * işaretini alırsınız ve gerekli komutu girersiniz. Bu tür operasyonlar kaynak ve varış yeri disklerinde PIP.COM dosyaları olmadığı zaman dosyaların kopyasında kullanışlıdır. Sistem diskinin 1. Yüzünden PIP'i yükleyin, sistem diskini çıkartıp kopyalamada kullanacağınız diski sokun.

PIP'ten çıkmak için * hazırda RETURN'e basın. PIP bir diskden diğerine kopyalamada tek sürücü sisteminde kullanılabilir, hazır durumu otomatik olarak diskin değiştirilmesini görüntülenir. Kaynak ve varış yeri sürücü tanımlayıcısı değişik olmalıdır.

Sistem İdaresi

DIR, ERASE, RENAME ve TYPE, yapılmış kopyalarından daha fazla işlevleri olan geçiş programlarıdır. Digital araştırma tarafından sunulan diğer geçiş programları gibi ikinci parametreler köşeli parantezle belirtilmiştir. Bütün detaylar HELP dosyasındadır. Bir kaç örnek verelim:

DIR [FULL] Dosyanın ebadını ve sıfatını görüntüler.
ERASE*.COM [CONFIRM] Her dosyayı buldukça doğrulamak için hazır duruma geçirir.

RENAME	Yazılan yeni ve eski dosya adlarını sorar.
RENAME *.SAV = *.BAK	Bütün .BAK dosyaların adını .SAV olarak yeniler.
TYPE KEYS.WP [NOPAGE]	Ekran sayfa numaralandırmasını meneder.

SYS (Sistem) ve RO (Read/Only) dosya sıfatlandırılmasından daha önce bahsedilmişti. Bu ve diğer birçok sıfatlandırılmalar, rastgele kartları kabul eden, SET komutunu kullanarak tahsis edilirler. Aşağıdaki komutlar:

```
SET *.COM [RO]
SET KEYS.CCP [RO]
SET A: [RO]
```

kazara silinmeyi önlemek için dosyaları hazırlar veya Read/Only durumunu sürer.

Aşağıdaki komutlar:

```
SET *.COM [RW]
SET KEYS.CCP [RW]
SET A: [RW]
```

dosyaları resetler veya Read/Write durumunu sürer. Aşağıdaki komutlar:

```
SET *.COM [SYS]
SET KEYS.CCP [SYS]
```

sistem sıfatını dosyalara verir. Bu sıfatdaki dosyalar DIR komutu tarafından görülmez (DIRS veya DIRSYS gereklidir). Bununla beraber dosyalar kullanmaya ve ilaveye hazır olarak bulunurlar, eğer dosyalar Ø kullanıcı alanı numarasında bulunuyorsa diğer bütün kullanıcı alan numaraları içni mevcutturlar.

Aşağıdaki komutlar:

```
SET *.COM [DIR]
SET KEYS.CCP [DIR]
```

sistem sıfatlarını çıkartır.

Her diske bir başlık, isim ve hatta parola tayin edilebilir. Parola rehberin içindeki dosya adları yerine,rehberin kendini korur. Her dosyaya kendi başına bir parola tayin edilebilir.

```
SET [NAME = ROLAND]
SET [PASSWORD = SALLY]
SET [PROTECT = ON]
```

disk üzerinde etkili olur.

```
SET *.* [PASSWORD = SALLY]
SET *.* [PROTECT = READ]
```

Diskin üzerindeki dosyalar üzerinde etkili olur (rastgele kartlar, *.* burada bütün kayıtları belirtmek için kullanılıyor). Tarih ve zaman vurgulaması INITDIR komutu tarafından harekete geçirilir (Sistem disk paketinizin 2. Yüzünde). Komutlar:

INITDIR

SET [CREATE = ON] veya SET [CCESS = ON] ve

SET [UPDATE = ON]
DIR [FULL]

belirtilen sürücü üzerinde tarihi görüntüler ve başlatır, şöyle yazın:

DATE SET

Gerekirse her seferinde CP/M Plus saati ayarlamaya başlar. Saat bir kere ayarlandığında 6128 tarafından otomatik olarak işletilir ve

DATE ve
DATE CONTINUOUS

tarafından kontrol edilir.

> **UYARI**
>
> Eğer parola, disk isimlendirilmesi veya tarih ve zaman vurgulaması çalıştırılmışsa disk sonradan AMSDOS veya CP/M 2.2 tarafından yazılmamalı veya bu faaliyetlere dayandırılmamalıdır.

Normal olarak dosyalar daha önceden tespit edilmiş sürücüden girerler, sadece sürücü belirtilmiş ise bu sürücüden girerler. Komut şöyledir:

SETDEF *,A:

Burada * işareti daha önceden belirtilmiş sürücüyü temsil eder. Komut CP/M'i (dosyaları aradığı zaman) önce, önceden belirtilen sürücüden sonra sürücü A:dan aramasını bildirir. Diğer bir deyişle, daha önceden belirlenmiş sürücü B: ise, dosyalar sürücü A: da dahi olsa otomatik olarak bulunacaktır.

Komutlar:
SETDEF [PAGE] ve
SETDEF [NOPAGE]

Konsol görüntücüsünün otomatik sayfa numaralayıcısını on ve off durumlarına getirir.

DEVICES, SET ve SETDEF'in bir çok faaliyetinde CP/M Plus başlarken herseferinde tarih ile birlikte sürücünün ayarlanması gerekir. Bu uygun bir PROFILE.SUB dosyası için birinci uygulamadır.

SUBMIT bağımsız komut dosyalarının otomatik olarak işleme konması için gereklidir. Komut dosyalarının içeriği metindir ve eğer .SUB dosyasındaki satırların ilk karakteri < ise programlara giriş ilave etmek mümkündür.

Sürme boyutu, boşluk miktarı, dosyaları içeren kullanıcı alanları ve varsa disk etiketi ile birlikte diskte kalan rehber girdi sayısı, SHOW komutunun değişik kombinasyonları ile görüntülenirler. Şöyleki:
SHOW B:
SHOW B: [LABEL]
SHOW B: [USERS]
SHOW B: [DIR]
SHOW B: [DRIVE]

Hepside sürücü B'deki istatistikleri belirler.

CP/M Plus'dan Çıkmak

AMSDOS

Bu program CP/M tarafından kontrolu terkedip AMSTRAD BASIC'e döner ki AMSDOS disk komutlarını içerir.

İleri Programlama

Sistem disk paketinizin 2. Yüzünde ileri ve sistem programcılarının kullanması için hazırlanmış programlar içerir. Kullanıcıya SOFT971-A Guide to CP/M veya diğer referans çalışmalarına baş vurması önerilir.

CP/M 2.2 İle Çalışma

CP/M 2.2, Sistem diskinin ilk iki izi'nden (track) yüklenir. CP/M Plus'ı yüklerken kullanılandan, bootstrap bölgesi değişiktir ve bunları karıştırmamak için dikkatli olmalıyız. Vendor, IBM ve Data format disklerini her iki disk sürücüsünde de kullanmak mümkündür ama çalıştırma bağıntıları genellikle ikinci disk sürücüsünde kullanılmalarını dahdit eder. CP/M programı tarafından özel bir işlem yapılmazsa (FILECOPY'nin yaptığı gibi) CP/M 2.2, disk logged-in yapılmamışsa üzerine yazmanıza izin vermez. Bundan başka disk formatının tipi (System, Data veya IBM) sadece disk logged-in yapılmışsa yeniden hesaplanır. Bilgisayarla birlikte olan ana disk sürücüsü (Sürücü A) CP/M 2.2 Direkt konsol mod'a döndüğünde veya [CONTROL] C;A> veya B > iken yazıldığında yerini alır. İlave sürcü (Sürücü B) için Sürücü A logget-in olduktan sonra Sürücü B'deki disk'e giriş yapılınca yerini alır. Eğer disk'e logget-in olmadan yazmaya kalkarsanız aşağıdaki hata mesajını alırsınız:

Bdos Err on (sürücü):R/O

Devam için herhangi bir tuşa basın. Eğer değiştirilen disk değişik formatlıysa okuma ve yazma hatası meydana gelir, devam için C yazın. Eğer Vendor formatlı bir yazılım disketiniz varsa, bunu kullanmak için ya FILECOPY veya PIP kullanarak CP/M 2.2 sistem disketine kopya edin yada CP/M'i ilave ederek diski sistem diskine çevirin. Bu BOOTGEN ve SYSGEN komutlarıyla beraber arşivlenmiştir. Bu kitabın arkasındaki "İlave 1"deki Kullanıcı Lisans Antlaşmasını dikkatli okuyun.

SYSGEN (parametresiz) özel bir kopya programıdır, kaynak ve varış yeri disklerini hazır duruma getirerek CP/M 2.2 sistem kanallarını bir diskten diğerine kopya eder. BOOTGEN aynı şekilde, bölge 1, kanal 0 (yükleyici) ve konfigürasyon bölgesini bir diskten diğerine kopya eder.

DIR komutu parametre almaz (dosya belirleyici hariç). Dosya isimleri herhangi bir sıraya göre ayrılmaz fakat DIR'de görülen dosya isim pozisyonları, disk rehberindeki bu dosyaların girişlerini belirtir.

STAT, SET ve SHOW'un bazı temel fonksiyonları önceden hazırlanmıştır. Aşağıdaki komutlar:

```
STAT
STAT A:
STAT B:
```

diskin vaziyeti ve boş alanını gösterir. Aşağıdakiler

```
STAT *.COM
STAT DISC.BAS
```

belirli bir dosya için genişletilmiş rehber bilgisini görüntüler. Aşağıdaki komutlar:

```
STAT *.COM $R/O
STAT DISC.BAS $R/O
```

Read/Only'yi ters çevirerek dosyayı Read/Write haline sokar. Aşağıdaki komutlar:

```
STAT *.COM $R/W
STAT DISC.BAS $R/W
```

Dosyayı kazara silinip üzerine yazılmaması için Read/Only haline sokar. Aşağıdaki komutlar:

```
STAT *.COM $SYS
STAT SECRET.BAS $SYS
```

Dosyayı sistem haline çevirir, şöyleki, rehber listesi ve dosya kopyalama programları görülmez. Dosya diğer maksatlar için mevcuttur. Aşağıdaki komutlar:

```
STAT *.COM $DIR
STAT SECRET.BAS $DIR
```

Dosyayı sistem halinde rehber haline çevirir.

FILECOPY tek sürücü kullanarak dosyaların diskten diske kopyasına iman verir. Program, ekranda bütün talimatları verir. FILECOPY sizden her dosyanın tek tek kopya edilip edilmeyeceğini teyid etmenizi ister. Program her dosya kopya edildikçe dosya adlarını vererek sizi haberdar eder.

FILECOPY *.COM	Bütün .COM dosyalarını kopya eder.
FILECOPY PIP.COM	PIP.COM dosyasını kopya eder.

DISCKIT2, DISCKIT3 ile de aynı fonksiyonlar yerine getirilir ancak kopya biraz yavaştır. Disk ve kaset arasında dosyaları transfer etmek gibi iki faydası daha mevcuttur. Özel kullanımlar haricinde muhtemel olmayan ASCII'den farklı (adi yazılar) dosyalar kullanışlı bir biçimle transfer edilir.

CLOAD (Kaset yükleme) iki parametre içerir, birincisi kaynak (kaset) dosya adı tırnak işareti içindedir, ikincisi varış yeri (disk) dosya adıdır. Eğer varış yeri dosya adı konmamışsa disk dosyası, kaset dosyasıyla aynı adı alır. Eğer kaynak dosya adı konmamışsa CLOAD teybin üzerindeki ilk dosyayı okur. Eğer kaset dosya adının ilk karakteri ! işareti ise normal kaset mesajı bastırılır.
Örnek komut:

```
CLOAD "MY LETTER" MYLETTER.TXT
```

CSAVE (kasete sakla) üç parametre içerir. İlk parametre kaynak (disk) dosya adıdır, ikincisi varış yeri (kaset) dosya adıdır ve tırnak işareti içindedir. Eğer varış yeri dosya adı konmamışsa kaset dosyası, disk dosyası ile aynı adı alır. Eğer kaset dosya adının ilk karakteri ! işareti ise normal kaset mesajı bastırılır. Eğer her iki dosya adı belirtilmemişse üçüncü parametre teybin yazma hızını belirlemek için kullanılır; Ø nominal 1ØØØ baud için, 1 nominal 2ØØØ baud için. Örnek komut:

```
SCAVE OUTPUT.TXT "OUT PUT TEXT" 1
CSAVE DATAFILE
```

SETUP

Bu kolaylık disk sürücü bölümünün seri interface ve 6128 tuş takımının karakteristiklerinin yeniden tanımlamaya imkan verir. CP/M Plus ile önceden hazırlanmış ayrı faydalara benzemeyen, SETUP diskin konfigürasyon bölgelerini değiştirir (sadece disk cold-booted durumunda iken). Bu yolla PROFILE.SUB kolaylığı ile hareketi aynıdır.
Program seçenek tablosu sürücüdür (menü-driven) ve ekran kısmen doğru ise veya herhangi düzeltme gerekmiyorsa aşağıdaki soruya Y cevabıyla bir sonrakine geçer.

```
Is this correct (Y/N):..
```

Program CONTROL C tuşu ile durdurulur. Bütün değişiklikler yapıldığı zaman şu mesaj görüntülenir:

Do you want to up date your system disc (Y/N): _

N, mevcut konfigürasyon bölgesini elinizde bulundurma şansı verir. Daha sonra:

Do you want to restart CP/M (Y/N): -

Y, yeni konfigürasyonu yükleyip çalışmanıza izin verir. Konfigürasyon bölgesinin bir diskten diğerine kopya etmek için ya BOOTGEN'i kullanın ya da kaynak disten her soruya cevabı vererek ve "Dou you want to update your system disc (Y/N:" sorusuna Y cevabı vermeden varış yeri diskinizi girerek SETUP'a yükleyin.

ASCII değerleri ondalık 32'den az olan karakterler, ! işaretini takip eden @, A-Z, [,\,],>,- işaretlerinden uygun olanının yazılmasıyla, dizi içine yazılabilirler.

Aşağıdakiler çok dikkat gerektiren bir seçeneklerdir:

** Initial command buffer:

Buna girilen herhangi bir karakter CP/M ilk yüklenirken sanki direkt konsol moda yazılmış gibi gözükecektir. Bu o anda belirli bir programın otomatik işletme etkisine sahip olması demektir. RETURN tuşunu ilave etmeyi unutmayın. (↑ M). Örneğin DIR'i otomatik olarak çalıştarmak için başlangıç komut buffer'ı aşağıdakini içermelidir:
DIR ↑ M

veya Dr.LOGO programı için:
SUBMIT LOGO2 ↑ M

Sign-on string;

Bu CP/M ilk yüklenirken ekranın en üstünde görüntülenen bir mesajdır. Carriage return/Satır besle etkisini yapan ↑ J ↑ M'in kullanılmasına dikkat edin. Standart mesajın ilk kısmı 80 kolonluk modda çalışmak için ekran ve çerçeve renklerini ayarlar, eğer saklanmışlarsa tam olarak kopya edilmelerini sağlar.

Tuş takımı değişimi
BASIC'in içindeki KEY DEF komutuyla belli tuşlara belli ASCII değerleri programlanabilir. Gereken parametreler tuş kodlarıdır ve ASCII değerleri bunları ayarlar. Tuş numaraları bilgisayarınızın sağ üst köşesindeki tabloda veya "Bilginiz için" adlı konunun 4. bölümünderir.

Tuş takımı genişlemesi

BASIC'deki KEY komutu ile ayarlanır.

SON OLARAK

DISCKIT2, SYSGEN, BOOTGEN, FILECOPY, SETUP, CSAVE ve CLOAD komutları AMSTRAD tarafından yaratılmıştır. AMSTRAD'dan başka diğer CP/M sistemleri ile çalışmaz ancak diğer firmaların kendi aletleri için aynı işleri gören (genellikle de aynı adda) programları vardır.

Aşağıdaki CP/M 2.2 programları, daha ileri düzeyde çalışanlar için Sistem Disk Paketinin 4. Yüzünde bulunmaktadır ve kullanımları SOFT159-A Guide to CP/M veya diğer referans kitaplarında ayrıntılı biçimde anlatılmıştır.

ASM	8080 Assembler
DDT	8080 Disassembler
DUMP	Hexadecimal formatta dump kolaylığı
ED	Basit bir editör
LOAD	ASM tarafından yapılan .HEX dosyasını .COM'a çevirir.
MOVCPM	CP/M 2.2 ile yeni TPA boyutu yaratır.
SUBMIT	Konsol Komut Programının "batch process"i
XSUB	Geçiş programının "batch process"i

Yukardaki programlar CP/M 2.2 ile kullanılmak içindir (4.Yüzde). Diğer yüzlerdeki CP/M Plus programlarıyla karıştırmayın.

KONU 6
LOGO'NUN TANITIMI

Bu kısımda LOGO konusunun örneklerle ve eldeki komutlarla rehber hazırlayarak tanıtımı hedeflenmiştir. Bu bütün teferruatıyla izah eden eğitici veya referans rehberi değildir; ve bu konuda AMSOFT'un yayınlarına ve diğer bağımsız yayımcılara baş vurmanız tavsiye olunur.
Bu dersin içeriği;

* **LOGO kavramı**
* **Dr. LOGO'nun yüklenmesi ve çalıştırılması**
* **Turtle grafikleri**
* **Kendi usulünüzü yazmak**
* **Kendi usulünüzün montajı**

LOGO NEDİR?

LOGO size daha önce program yapıp yapmamanıza bakmadan bir programcı gibi yetişmenize yardım edebilir.

LOGO kullanması ve öğrenmesi çok kolay olduğu için süratle popülerite kazanan kuvvetli bir programlama lisanıdır.

Siz usulleri tuğlalar gibi kullanarak LOGO programları meydana getirebilirsiniz. Dr.LOGO bir usuller topluluğudur ve siz bunları kullanarak kendi programlarınızı yapabilirsiniz.

1970'lerde bir grup bilgisayarcı ve öğretmen Seymour Papert'ın idaresi altında çok küçük çocukların program yapabilmesine ve bilgisayar kullanabilmesine izin veren turtle grafikleri ile LOGO'yu geliştirdiler.

Papert'ın söylediği gibi genç öğrencilerin düşünecekleri bir nesneyi yeni yollardan öğrenirken yardımcı bir aracı olan Turtle (kaplumbağa) geliştirdiler.

Dr.LOGO

Dr.LOGO programları daha kolay hale getirmek için AMSTRAD bilgisayarları için özel olarak hazırlanmış LOGO'nun düşünceli bir tamamlayıcısıdır. Genişlemelerin ilavesiyle 6128'in güçlü ses kolaylıklarının elde edilmesi mümkün kılınmış ve kursör tuşlarının ilavesiyle program düzeltmede kolaylık sağlanmıştır.

Başlarken

Dr. LOGO ile çalışmak için ana disk paketinizin 1. yüzünü bilgisayarın disk sürücüsüne sokun ve şunu yazın:

```
ICPM
```

A> promptu belirdiğinde diski çıkartın ve sistem disklerinden 3. yüzü (DR.LOGO HELP) girin. Dr. LOGO'yu çalıştırmak için şunu yazın:

```
SUBMIT LOGO3
```

Birkaç saniye sonra "Welcome" açılış mesajını ve ardından ? işaretini göreceksiniz.

CP/M 2.2 İçin Dr. LOGO

Ana disk paketinin 4. yüzü, Dr.LOGO'nun CP/M 2.2 ile çalışması için (AMSTRAD CPC664 ve CPC464 + DDI1 ile) kullanılır. Bu sebeple Dr.LOGO'nun 4. yüz versiyonunun genel olarak 6128 ile kullanılması tavsiye edilmez.

Eğer buna rağmen Dr.LOGO'nun CP/M 2.2 versiyonunu kullanmak isterseniz 4. yüzün kopyasını sokunuz ve şunu yazınız:

```
ICPM
```

A> belirdiği zaman ise şunu yazınız:

```
SUBMIT LOGO2
```

Birkaç saniye sonra "Welcome" açılış mesajını ve ardından ? işaretini göreceksiniz.

İlk Adımlar

? işareti ikazı Dr.LOGO'nun tuş takımından herhangi bir şey yazmanızı beklediğini söyler. Şunu küçük harflerle yazınız:

fd 60

Başladığı yerden bittiği yere kadar arkasında bir çizgi bırakan ve 60 birim ileriye giden bir kaplumbağa (turtle-büyük bir ok başı) göreceksiniz. Ekran, ? işareti ikazının ekranın dip kısmına yaklaşmasıyla büyük bir grafik alanı ve daha küçük metin arası vermek için temizlenir.

Dr.LOGO büyük metin ve grafik alanı verebilmek için ekranı ayarlar. Aşağıdakini yazınız:

rt 90

Kaplumbağa 90 derece sağa gitmiştir.

Şimdi şunu yazınız:

fd 60

Başka bir çizgi aynı uzunlukta ilkine dik açı yaparak çizilir.

Ekranda neler olduğunu görmek için fd, bk, rt, lt (sırası ile; ileri, geri, sağa ve sola) talimatlarını deneyin.

Dr. LOGO Yöntemleri

Şimdi siz, Dr.LOGO'da yapılmış ve "ilkeller" diye adlandırılan komutları ekleyerek ilk yönteminizi yazabilirsiniz.

fd, bk, rt ve lt yapılmış ilkellerdir ve kendi yönteminizi yazmak için yapı taşları gibi istediğiniz zaman kullanırsınız.

Diğer faydalı ilkel cs'dir ve ekranı temizleyip başa döner.

Basit Bir Yöntem Yazmak

eğer hareket aşağıdaki gibi olursa görüntülemek mümkündür.

fd 60 rt 90

eğer yukarıdaki 4 defa tekrarlanırsa kenarları 60 birim olan bir kare çizilir.

repeat 4 [fd 60 rt 90]

ekranı temizleyin ve yukarıdakini yazarak ne olacağını izleyin.

Bu formülü yeni bir yöntem yapmak için (kare adlı) şunu yazın.

```
to kare
repeat 4 [fd 60 rt 90]
end
```

Dr.LOGO "kare"yi anlar ve her "kare" sözcüğü geçtiğinde bir kare çizer. Biz bu yönteme herhangi bir ad verebiliriz fakat ne olduğunu hemen hatırlayalım diye "kare"yi seçtik.

Dr.LOGO bütün komut setlerini beraber yazmamıza imkan verir o zaman talimat:
kare rt 45 kare

2 kare çizer ki 2'ncisi 1'ncisiyle 45 derecelik açı yapar.

Parametreli Yöntemler

Kenar boyutları değişik kareler çizen bir yöntem yapmak için "kare" tarifini değiştirmek gerekir. Şöyle ki:

```
to squareany size:side
repeat 4 [fd:side rt 90]
end
```

:side değişkeninin kolondan başladığını önceden belirtmeniz gerekir. Bu Dr. LOGO'ya :side'ın bir komuttan çok bir değişken olduğunu belirtir. Squareanysize yöntemini kullanırken :side değişkeninin bir değeri olması gerekir. Squareany size 150 emri, her köşesi 150 ünitelik bir kare meydana getirir. İki emri beraber toplayarak deneyin ve neler olacağını görün. Örnek olarak:

```
cs squareanysize 100 rt squareanysize 150
```

kaplumbağa, değişik boyutlarda ve biri öbürüne 45 derece açı yapan iki tane kare çizecektir.

Komut satırı, bir ekran satırını aşınca, Dr. LOGO'nun sizi ! işaretiyle uyarmasına dikkat ediniz.

Değerleri Hatırlamak İçin Kullanılan Değişkenler

Dr. LOGO değerleri hatırlamak için değişken kullanmaya da izin verir, bu bir emirde geçen değerlerde olabilir.

Önce adı üçgen olan yeni bir komut yazalım:

```
to üçgen
repeat 3 [fd :edge rt 120]
end
```

Test için şunu yazabiliriz:

```
make "edge 100
üçgen
```

Eğer :edge'nin değerini bilmek istiyorsak ? prompt'undan sonra :edge yazmak yeterlidir, Dr.LOGO değeri yazacaktır.

Son olarak :edge değişkenini bir model çizdirmek için kullanabiliriz. :Edge değerinin her seferinde daha öncekine göre nasıl arttığına ve her bir modeli çizdiğinde nasıl büyüdüğüne dikkat ediniz.

```
to model
üçgen lt 60 üçgen rt 60
make "edge :edge + 4
model
end
make "edge 10
cs model
```

Yeteri kadar gördükten sonra durdurmak isterseniz [ESC] tuşuna basınız.

Programlar ve Yöntemler Yazmak

Dr. LOGO, yazım hatalarımızı ve belirtiğimiz yöntemdeki yanlışlarımızı düzeltmemize razı olur. Düzeltme tuşları, kürsörün tuşları, yukarı, aşağı, sağa, sola gitmenize yardım eder. Eğer bu tuşlara CONTROL tuşuyla beraber basarsanız, metnin başına, sonuna veya satır başına, sonuna gider. CLR kürsör'ün altındaki karakteri siler, DEL ise kürsörün solundaki karakteri siler. RETURN tuşu Dr.LOGO'ya bir satırın bittiğini ve yeni satır istendiğini belirtir. ESC, programın terk edildiğini; COPY ise programın yazılmasının bittiğini belirtir. Yeni komutlar ve yöntemler yazarken sadece elinizdeki metni olduğu gibi ekrana geçirin. Metnin içindeki karakterler kürsördeki pozisyonlarına göre yazılmıştır.

Daha önceden var olan bir yöntemi görmek için ed komutunu kullanın; Dr.LOGO bunun eski durumunu size gösterecektir. İşte o zaman yukarda bahsedilen tuşları kullanarak, istediğiniz değişikliği yapabilirsiniz.

'model' adlı daha önceden yaptığınız programı isteyin:

ed "model

Düzeltme tuşlarıyla değişiklikleri yapın. Bitirdiğiniz zaman ESC tuşuna basın, Dr.LOGO hemen ekrana düzeltilmiş programı yazacaktır. ed "model'i tekrar yazın ve 4 sayısını 8 ile değiştirdikten sonra çıkmak için COPY'ye basın. Programı tekrar işletin ve ekran çıkışının nasıl değiştini görün. Tabii başlangıç :edge değerini belirtmeyi unutmayın.

İşlem Kolaylıkları

Dr.LOGO'nun kullandığı çalışma alanı "node"lara ayrılmıştır. Ne kadar kaldığını şöyle anlayabilirsiniz:

```
nodes
```

Sıklıkla, "node"ların bitimine yakın, Dr.LOGO çalışma alanını düzenleyecektir ve siz bunu kaplumbağa'nın biraz ara vermesi ile anlayabilirsiniz. Sizde Dr.LOGO'dan, çalışma alanını düzenlemesini isteyebilirsiniz, bunu şöyle belirtebilirsiniz:

```
recycle
```

Dr.LOGO yer kalmadığını belirtinceye kadar, sizin çalışmanıza izin verecektir.

Eğer CP/M 2.2'nin Dr.LOGO'sunu kullanıyorsanız (4.Yüzdeki) öncelikle diskinizde boş yer olup olmadığına bakın, çünkü yaptığınız programları yüklemek isteyebilirsiniz. Bunun için AMSDOS'daki CAT komutunu kullanabilirsiniz.

Şimdide, belirtilen komutları kullanarak, biraz çalışın Dr.LOGO hakkında öğrendiklerinizle, gittikçe daha iyiye giden çalışmalar yapacağınıza eminiz.

Dr.LOGO ile işiniz bittiğinde şunu yazın:

```
bye
```

Dr.LOGO Komutlarının Listesi

Gelecek bölümdeki Dr.LOGO komutlarının alfabetikal liste gurupları, sıklıkla kullanım örnekleriyle, size kullanmasını gösterecektir.

> **NOT:**
> Asterix (*) işareti ile belirtilmiş komutlar CP/M 2.2'nin Dr.LOGO'suyla ÇALMAZ. Ve bu komutları kullanan programlar, CP/M 2.2 ile çalışan CPC664 veya CPC464 + DDI1 ile uygun değildir.

Kelime ve Listeleme Komutları:

(Aşağıdaki örneklerde ? ve > işaretlerine dikkat edin)

ascii

Giriş kelimesinin ilk karakterinin ASCII değerini verir.

```
?ascii "G
71
?ascii "g
103
```

bf

(but first) Girilen karakterlerin ilki hariç diğerlerini görüntüler.

```
?bf "amstrad
mstrad
?bf [1 2 3]
[2 3]
```

bl

(but last) Girilen karakterlerin sonuncusu hariç diğerlerini görüntüler.

```
?bl "amstrad
amstra
?bl [ 1 2 3 4]
[1 2 3]
```

char

Verilen numaranın ASCII değerindeki karakter karşılığını görüntüler.

```
?char 83
S
```

count

Girilen karakterlerin sayısını verir.

```
?count "altı
4
?count [0 1 2]
3
```

emptyp

Giriş karakteri boş bir kelime veya liste ise DOĞRU (TRUE), aksi ise YANLIŞ (FALSE) görüntülemesi verir.

```
?emptyp"
TRUE
?emptyp [ ]
TRUE
?emptyp [x]
FALSE
?make "x [ ]
?emptyp :x
TRUE
```

first

Girilen karakterlerin ilk elemanını görüntüler.

```
?first "zebra
z
?first [1 2 3]
1
```

fput

(first put) Birbiri arkasına girilen iki elemanı birleştirerek yeni bir dizi yapar.

```
?fput "a "mstrad
amstrad
?fput 1 [2 3]
[1 2 3]
```

item

Dizide belirlenen karakteri görüntüler.

```
?item 4 "amstrad
t
```

*

last

Girilen karakterlerin son elemanını görüntüler (first ile karşılaştırın).

```
?last "amstrad
d
```

*

lc

Girilen alfabetik karakterleri küçük harfle yazar (uc'ye bakın).

```
llc "AMSTRAD
amstrad
```

list

Girilen karakterlerden liste çıkartır, bunları köşeli parantez ile sınırlandırır (se ile karşılaştırın).

```
?(list 1 2 3 4)
[1 2 3 4]
?list "dr logo [amstrad]
[dr logo [amstrad]]
?(list)
[ ]
```

*

listp

Girilen karakterler bir liste ise DOĞRU (TRUE), değil ise YANLIŞ (FALSE) yazar.

```
?listp "anne
FALSE
?list [baba kardeş yeğen]
TRUE
```

*

lput

(last put) Birbiri arkasına girilen iki elemanı, ilkini ikincinin sonuna getirerek, yeni dizi yapar.

```
?lput "d "amstra
amstrad
?lput "d [amstra]
[amstra d]
```

*

memberp

İlk girilen, ikinci girilen gurubun elemanı ise TRUE (doğru) çıkışını verir.

```
?memberp "y" yalnız
TRUE
?memberp "çikolata    [[vanilya][çikolata][çilek]]
FALSE
?memberp [çikolata]    [[vanilya][çikolata][çilek]]
TRUE
```

*

numberp

Girilen sayı ise TRUE çıkışını verir.

```
?numberp 374.926
TRUE
?numberp "altı
FALSE
?numberp first [2 4 6 8]
TRUE
```

*

piece

Girilen elemanda belirtilen sayılar arasındaki grubu yazar.

```
?piece 4 7 "İstanbul
anbu
?piece 1 4 [Ahmet Mehmet Ali Veli Osman]
[Ahmet Mehmet Ali Veli]
```

se

(sentence) Girilen elemanlardan bir liste meydana getirir ve bunu parantezler arasında belirtir. (list ile karşılaştırın).

```
?make "instr _ list rl
repeat 4 [fd 50 rt 90]
?run (se "cs :instr list "ht)
```

instr ile list arasındaki karakter [SHIFT] ile Ø tuşuna basarak elde edilir.

*

shuffle

Verilen listedeki elemanları gelişigüzel sıralar.

```
?shuffle [a b c d]
[c b d a]
```

*

uc

Yazılan alfabetik karakterleri büyük harfle yazar (lc ile karşılaştırın).

```
?uc "ahmet
AHMET
```

*

where

memberp ifadesinde sorulan harfin kaçıncı sırada olduğunu belirtir.

```
?memberp "h" nehir
TRUE
?show where
3
```

word

Girilen kelimelerden yeni bir kelime yapar.

```
?word "ay "ışığı
ayışığı
```

wordp

Girilen sayı veya kelime ise TRUE çıkışını verir.

```
?wordp "merhaba
TRUE
?wordp [ ]
FALSE
```

Aritmetik işlemleri:

*

arctan

Girilen sayının (derece olarak) ark tanjantını hesaplar.

```
?arctan 0
0
?arctan 1
45
```

cos

Girilen derecenin kosinüsünü hesaplar.
?cos 6Ø
Ø.5

int

Girilen bölümlü sayının sadece tam sayısını verir.
?int 4/3
1

*
quotient

Girilen iki sayının bölümü sonrası tam sayıyı verir.
?quotient 14 4
3
?14/4
3.5

random

Giriş sayısından küçük, rastgele, negatif olmayan bir sayı üretir.
?random 2Ø
7

*
remainder

Birinci sayının ikinciye bölünmesinden kalan sayıyı görüntüler.
?remainder 7 3
1
?remainder 8 4
Ø

*

rerandom

Aynı random sırasında yeni bir random alt gurubu üretir.

```
?repeat 1Ø [(type random 1Ø char 9)]
3       7       5       3       2       0       4       2       6
?repeat 1Ø [(type random 1Ø char 9)]
9       9       1       0       6       1       3       5       1
?rerandom
?repeat 1Ø [(type random 1Ø char 9(]
2       9       0       3       1       6       2       3       7
?rerandom
?repeat 1Ø [(type random 1Ø char 9)]
2       9       0       3       1       6       2       3       7
```

*

round

Girilen sayıyı en yakın tam sayıya tamamlar.

```
?round 3.3333333
3
?round 3.5
4
```

sin

Girilen derece cinsinden sayının sinüsünü verir.

```
?sin 3Ø
Ø.5
```

+

Girilen sayıların toplamını verir.

```
? + 2 2
4
?2 + 2
4
```

-

Girilen sayıların farkını verir.

```
?- 1Ø 5
5
?1Ø-5
5
```

*

Girilen sayıların çarpımlarını verir.

```
?* 4 6
24
?4*6
24
```

/

Girilen sayıların bölümünü verir.

```
?/ 25 5
5
?25/5
5
```

Mantık İşlemleri:

and

Girilen ifadeler doğru ise TRUE görüntüler.

```
?and (3< 4) (7> 4)
TRUE
```

not

Giriş ifadesi doğru ise FALSE yanlış ise TRUE görütüler.

```
?not (3 = 4)
TRUE
?not (3 = 3)
FALSE
```

or

Girilen bütün ifadeler yanlış ise FALSE görüntüler.

```
?or "TRUE" FALSE
TRUE
?or (3 = 4) (1 = 2)
FALSE
```

=

Girilen iki ifade birbirine eşitse TRUE, yanlış ise FALSE görüntüler.

```
?= "LOGO" LOGO
TRUE
?1 = 2
FALSE
```

>

İlk girilen ifade ikinciden büyükse TRUE, aksinde FALSE görüntüler.

```
? 19 20
FALSE
?20 19
TRUE
```

<

İlk girişen ifade ikinciden küçükse TRUE, aksinde FALSE görüntüler.

```
?   27 13
FALSE
?13   27
 TRUE
```

Değişkenler:

local

Akış prosedüründe kabul edilen ve prosedürün çağırdığı değişken isim girişleri yaratır.

```
> (local "x "y "z)
```

make

Girilene eşit değişken isim girişleri yaratır.

```
?make "side 50
?:side
50
```

*

namep

Giriş kelimesi daha önceden belirlenmiş ise TRUE görüntüler.

```
?make "un "çikolata
?:un
çikolata
?namep "un
TRUE
?namep "çikolata
FALSE
```

*

thing

Değişken isim girişinin değerini görüntüler.

```
?make "bilgisayar "amstrad
?thing "bilgisayar
amstrad
```

Prosedürler:

*

define

Özgün prosedür isminin belirtilmesinin, belirtim listesini yapar.

```
?define "say.hello [[ ] [pr "hello]]
?po "say.hello
to say. hello
pr "hello
.text "say.hello
[[ ] [pr "hello]]
end
```

end

Prosedür belirtiminin sonunu belirtir; tek başına en son satırın başında yer almalıdır.

```
?to kare
>repeat 4  [fd 5Ø rt 9Ø]
>end
kare defined
?kare
```

po

(print out) özgün prosedür(ler) veya değişken(ler)in ifade(leri)sini görüntüler.

```
?po "kare
to kare
repeat 4 [fd 5Ø rt 9Ø]
end
?make "x 3
?po "x
x is 3
```

pots

(print out titles) Çalışmada kullanılan bütün prosedürlerin adını ve başlıklarını verir.

```
?pots
```

*

text

Özel bir prosedürün listesinin açıklamasını görüntüler.

```
?to yıldız ;beş köşeli yıldız
> repeat 5  [fd 3Ø rt 144 fd 3Ø lt 72]
>end
yıldız defined
?text "yıldız
[[ ] [repeat 5 [fd 30 rt 144 fd 30 lt 72]]]
```

to

prosedür açıklamasının başladığını belirtir.

```
?to kare
>repeat 4  [fd 5Ø rt 9Ø]
>end
kare defined
```

Düzeltme:

ed

(edit) Belirtilen prosedür ve/veya değişkenleri düzeltmek için ekran "buffer"ına yükler.
?ed "kare

*

edall

Bütün prosedür ve değişkenleri düzeltmek için ekran "buffer"ına yükler ve ekran editörüne giriş yapar.
?edalı

*

edf

Belirtilen disk dosyasını, direkt olarak diskten ekran "buffer"ına yükler veya yeni bir dosya yaratarak ekran editörüne boş bir "buffer" girer.
?edf "yıldız

Printer Fonksiyonları:

*

copyon

Metnin printer'e yazılmasını başlatır.
?copyon
*

copyoff

Metnin printer'e yazılmasını sonlandırır.
?copyoff

Metin Ekranı:

ct

(clear text) Ekrandaki bütün metni ve aynı zamanda kürsörü de siler ve kürsörü ekranın sol üst köşesine yerleştirir.
?ct

*

cursor

Metin ekranında kürsörün satır ve kolon numarasını içeren koordinat listesi görüntüler.

```
?ct
?cursor
[01]
?(type [Kürsörün şu andaki pozisyonu  ] show cursor
Kursörün şu andaki pozisyonu [32 23]
```

pr

(prınt) Girilen nesneleri metin ekranında görüntüler, son girişi bir "carriage return" ile izler (show ve tpye ile karşılaştırın).

```
?pr [a b c]
a b c
```

*

setcursor

Kürsörü, girilen ekran metninin listesinde belirtilen yere götürür.

```
?ct
?to resim
>make "x random 20
>make "y random 12
>setcursor list :x :y pr "*
>end
?resim
```

setsplit

Split ekranında satır sayısını belirler

```
? setsplit 10
```

show

Girilen nesneleri metin ekranında görüntüler, son girişi bir "carriage return" ile izler (pr ve type ile karşılaştırın).

```
?show [a b c]
[a b c]
```

ts

(test screen) Bütün metin ekranını seçer.
?ts

type

Girilen nesneleri ekranda görüntüler ancak son girişi bir "carriage return" takip etmez.
?type [a b c]
a b c

Grafik ekran:

Ekran, dört renk veren, AMSTRAD BASIC de kullanılan aynı koordinatlara sahip, Mode 1'dedir. Diğer bir deyişle bütün ekran pozisyonları önceden numaralanmış ekran noktasının yakınlarına yerleştirilecektir. Kırmızı, yeşil ve mavi sırasıyla şu renkleri alırlar: Ø, 1, 2.

clean

Kaplumbağa hariç tüm grafik ekranı siler.
?fd 5Ø
?clean

cs

(clear screen) Grafik ekranı siler ve kaplumbağa'yı (Ø,Ø) pozisyonuna yerleştirir.
?rt 9Ø fd 5Ø
?cs

dot

Ø andaki kalem renginde, belirtilen koordinat listesindeki yere bir nokta koyar.
? dot [5Ø 1Ø]

*

dotc

Belirtilen koordinat listesindeki noktanın renk numarasını verir, eğer yerleşimi ekran dışında ise -1 görüntülenir.

```
?cs
?setpc 1
?do -5Ø 5Ø
?setpc 2
?dot 5Ø 5Ø
?setpc 3
gdot 5Ø -5Ø
?dotc 5Ø 5Ø
2
?doıtc -5Ø -5Ø
Ø
?dotc 1ØØØ 3ØØØ
-1
```

fence

Görünen grafik ekranda kaplumbağa'yı sınırlayan bir hudut kurar. Window hududu hareket ettirir.

```
?fence
?fd 3ØØ
Turtle out of bounds
```

*

fill

Ø andaki kalemin durumunu kullanarak, kaplumbağa'nın altındaki noktanın rengini değiştirerek (ve aynı renkte bütün yatay ve dikey sürekli noktaları da), o anki kalem rengi ile bir alan çizer.

```
?make "x 5
?cs
?st
?pd
?reeat 3Ø [fd:x rt 9Ø make "x:x +5]
?fd 2Ø rt 9Ø
?fd 1Ø
?pu
?home
?bk 2
?pd
?setpc 2
?fill
```

fs

(full creen) Bütün grafik ekranı seçer.
?fs

pal

(palette) Kaleme eşlik eden renklerin sayısal karşılıkları görüntüler.
?pal 2
[Ø 2 2]

*

setbg

Grafik ekranın arkasını, girilen renk sayısına bağlı renge getirir.
?sf
[Ø SS 5 FENCE 1]
(Bu, ekran arkasının Ø'a ayarlandığını gösterir.)
?pal Ø
[Ø Ø 1]
?setbg 2
?sf
[2 SS 5 FENCE 1]

setpal

(set palette) Palet kalem rengini ayarlar. Kaleme renkleri bağlar.
?setpal 3 [1 1 2]
?pal 3 [1 1 2]

*

setscrunch

Grafik ekran görünümünü giriş sayısına göre ayarlar.
?sf
[Ø SS 5 FENCE 1]
? to çember
>repeat 36Ø (fd 1 rt 1)
>end
çember defined
?setscrunch 2
?sf
[Ø SS FENCE 2]
?çember
?setscrunch 2.5
?çember

sf

(screen facts) Grafik ekran hakkında bilgi verir. Formu şöyledir [<Geriplan rengi><ekran durumu><bölme boyutu><pencere durumu><scrunc>].<geriplan rengi> background pen numarasıdır (CP/M 2.2'de genellikle 0), <ekran durumu>nda SS (bölmeli ekran), FS (bütün ekran), TS (metin ekranı) kullanılır. <bölme boyutu>, bölmeli ekranın metin penceresinde görüntülenen metin satır sayısıdır. <pencere durumu>, WINDOW, WRAP, PENCE modlarını belirtir. <scrunch> ekranın görünümüdür ve 1'e ayarlıdır. Stescrunch ile sıfırlanabilir (CP/M 2.2 ile geçezsizdir.)

```
?sf
[Ø SS 5 FENCE 2.5]
```

ss

(split screen) Grafik ekranda metin için pencere açar.

```
?ss
```

window

wrap veya fence ifadelerinden sonra kaplumbağa'nın görünen grafik ekranın dışına çizmesine müsaade eder.

```
?fence fd 3ØØ
Turtle out of bounds
?window
?fd 3ØØ
```

wrap

Sınırları aştığında, kaplumbalğa'nın grafik ekranda, ters tarafta, tekrar görünmesini sağlar.

```
?cs wrap
?rt 5 fd 1ØØØ
?cs window
?rt 5 fd 1ØØØ
```

Kaplumbağa Grafikleri:

bk

(back) Kaplumbağa'nın durduğu yerden belirtilen adım kadar geri gitmesini sağlar.

```
?cs fd 15Ø
?bk 5Ø
```

fd

(forward) Kaplumbağa'nın durduğu yerden belirtilen adım kadar ileri gitmesini sağlar.

```
?fd 8Ø
```

*

home

Kaplumbağa'yı evine ([Ø Ø] pozisyonu = grafik ekranın ortası) döndürür.

```
?fd 1ØØ
?rt 45
?fd 1ØØ
?home
```

ht

(hide turtle) Kaplumbağayı görünmez hale getirir.

```
?ht
?cs fd 5Ø
?st
```

lt

(left) Kamlumbağa'yı belirtilen derece kadar sola döndürür.

```
?lt 9Ø
```

pd

(pen down) Kamlumbağa'nın kalemini aşağı çevirir; kaplumbağa çizmeye devam eder.

```
?fd 20 pu fd 20
?pd
?fd 20
```

pe

(pen erase) Kaplumbağa'nın rengini geri plan rengine çevirir; böylece çizilmiş satırları siler.

```
?fd 50
?pe
?bk 25
?fd 50
?pd fd 25
```

pu

(pen up) Kaplumbağa'nın kalemini yukarı çevirir; kaplumbağa çizmeyi durdurur.

```
?fd 30
?pu
?fd 30
?pd fd 30
```

px

(pen exchange) Kaplumbağa'yı renk değiştirir; daha önce boyanmış yerde ise tersini, değilse uygun rengi verir.

```
? fd 20 pu fd 20
?pd setpc 3 fd 20
?px
?bk 80
?fd 80
?pd bk 100
```

rt

Kaplumbağa'yı belirtilen derece kadar sağa döndürür.

```
?rt 90
```

seth

(set heading) Kaplumbağa'yı belirtilen sayı kadar döndürür; sayı pozitif ise saat yönüne, negatif ise saat yönünün tersine döndürür.

```
?seth 90
```

setpc

(set pen colour) Kaplumbağa'nın kaleminin rengini girilen sayıya göre ayarlar.

```
?setpc 1
```

setpos

(set position) Kaplumbağa'yı girilen koordinat listesinde belirtilen pozisyona götürür.

```
?setpos [30 20]
```

*

setx

Kaplumbağa'yı yatay pozisyonda, belirtilen x koordinatına götürür (sety'yede bakın).

```
?setx 80
?fd 100
?setx -50
?fd 50
```

*

sety

Kaplumbağa'yı düşey pozisyonda, belirtilen y koordinatına götürür (sety'yede bakın).

```
?sety 90
?fd 20
?sty -50
?fd 50
```

st

(show turtle) Kaplumbağa kaybolmuşsa görünür hale getirir.

```
?ht
?fd 50
?st
```

tf

(turle facts) Kaplumbağa hakkında bilgi verir. Formu şöyledir: [(xkoord) (ykoord) (başlık) (kalem) (k.rengi) (görünürlük)]. (xkoord) kaplumbağa'nın x koordinatıdır, (ykoord) y koordinatıdır. (başlık) kaplumbağa'nın kapladığı hacmi belirtir. (görünürlük), kaplumbağa'nın kapladığı hacmi belirtir. (görünürlük), eğer kaplumbağa görünüyorsa TRUE yazar. (kalem) kalemin durumunu belirtir: PD (kalem aşağı), PE (silici kalem), PU (kalem yukarı). (k.rengi) kalemin renk numarasını belirtir.

```
?setpos (15 30)
?rt 60
?setpc 3
?pe
?ht
?tf
(15 30 60 PE 3 FALSE)
```

*

towards

Koordinat listesinde durumu belirtilmiş kaplumbağa kadranının yapabildiği başlık görüntüler.

```
?set towards list :x :y
```

Çalışma Alanı Düzenlemesi:

er

(erase) Çalışma alanındaki belirli bir prosedürü siler.

```
?er "kare
```

*

erall

Çalışma alanındaki tüm verileri siler.

```
?erall
```

ern

(erase name) Çalışma alanındaki belirlenen değişkenleri siler.

```
?make "kenar [100]
?make "açı [45]
?:kenar:açı
[100]
[45]
?ern [kenar açı]
?:kenar
kenar has no value
```

nodes

Çalışma alanında ne kadar boş yer kaldığını görüntüler.

```
?nodes
```

*

noformat

Prosedürleri yorumları ile beraber çalışma alanında boş bir yere formatlanmaya götürür.

```
?noformat
```

*

poall

Çalışma alanındaki bütün değişken ve prosedürlerin açıklamalarını görüntüler.

```
?poall
```

*pons

Çalışma alanındaki bütün büyük değişkenlerin adını ve değerlerini görüntüler.

```
?pons
medium is 40
small is 20
large is 80
```

*

pops

Çalışma alanındaki bütün prosedür ve açıklamaları görüntüler.

```
?pops
```

recycle

Mümkün olduğunca az yer kullanarak çalışma alanını reorganize eder.

```
?recycle
?nodes
```

Destekleme Listeleri:

glist

(get list) Kendi destekleme listesinde, destek isim girişi yapılmış, çalışma alanındaki bütün nesnelerin listesini görüntüler.

```
?glist ".DEF
```

gprop

(get property) Adı konmuş nesnenin, destekleme adı girilmiş destek değerini görüntüler.

```
?make "yükseklik "72
?gprop "yükseklik ".APV
72
```

plist

(property list) Adı konmuş nesnenin destekleme listesini görüntüler.

```
?plist "yükseklik
(.APV 72)
```

pprop

(put property) Girilen destekleme çiftini, adı konmuş nesnelerin destekleme listesine yerleştirir.

```
?pprop "master ".APV "Scott
?:master
Scott
```

*pps

Çalışma alanındaki bütün nesnelerin, standart olmayan, destekleme çiftini görüntüler.

```
'pprop "Sally "extension 213
?pps
Sally's extension is 213
?plist "Sally
[extension 213]
```

remprop

(remove property) Adı konmuş nesnelerin destekleme listesinden belirtilen destelemeleri değiştirir.

```
?remprop "master ".APV
```

Disk Dosyaları:

*

changef

Diskteki bir dosyanın adını değiştirir.

```
?dir
[KARE ÇEMBER YILDIZ]
?changef "kutular "kare
?dir
(KUTULAR ÇEMBER YILDIZ)
```

*

defaultd

Hatalı sürücünün adını görüntüler.

```
?defaultd
a:
```

dir

(directory) Hatalı veya özgün diskteki dosya adlarını Dr.LOGO listesi halinde görüntüler.

```
?dir "a:????????
```

(Konunun 1. bölümündeki "AMSDOS ve CP/M de ???????? (wild cards karakterleri) kullanımı. Dr. LOGO'nun *işaretini kabul etmediğini gözleyiniz.)

*dirpic

Hatalı veya belirlenmiş diskten görüntü dosya adları listesini görüntüler. Çok büyük dosya adını da kabul eder.

```
?dirpic "b:
[MY-PICK KARE YILDIZ NOTLAR]
```

load

Çalışma alanındaki isimli dosyayı diskten okur.

```
?load "myfile
?load "b:yıldız
```

*

loadpic

Grafik ekrana belirli resim dosyasını yükleyerek, çizimi tekrar yaratır.

```
?loadpic "my pic
?loadpic "b:my pic
```

save

Çalışma alanının içeriğini belirlenen disk dosyasına kaydeder.

```
?save "shapes
```

NOT- Yüklemeden önce format edilmiş diskte yeterli boş yer olup olmadığını kontrol edin. Hiçbir zaman master disklerine yükleme yapmayınız. Eğer CP/M 2.2.'nin Dr.LOGO'sunu kullanıyorsanız, çalışmanın arasında disk değiştiremezsiniz, ki CP/M 2.2 Dr.LOGO'nun bulunduğu bölümde programlarınız için yeterli boş yer vardır.

*

savepic

Grafik ekran içeriğini belirlenen görüntü dosyasına kaydeder.

```
?savepic "my pic
?savepic "b:my pic
```

NOT- Yüklemeden önce format edilmiş diskte yeterli boş yer olup olmadığını kontrol edin. Hiçbir zaman master disklerine yükleme yapmayınız. Eğer CP/M 2.2'nin Dr.LOGO'sunu kullanıyorsanız çalışmanın arasında disk değiştiremezsiniz, ki CP/M 2.2 Dr.LOGO'sunun bulunduğu diskte programlarınız için yeteri kadar boş yer vardır.

*

setd

(set drive) Belirlemiş sürücüyü çalıştırır.

```
?defaultd
A:
?dir
[KUTULAR ÇEMBER YILDIZ]
?setd "b:
?defaultd
B:
?dir
[ÜÇGEN EV]
```

Tuştakımı ve Oyunçubuğu:

buttonp

(button pressed) Belirlenmiş oyunçubuğunda düğme aşağıdaysa TRUE görüntüler; Ø ve 1 sayıları oyunçubuğunun iki olasılığını belirtir.

```
?to ateş
  label "loop
  if (buttonp Ø) [pr (fire Ø!]]
  if (buttonp 1) [pr (fire 1!]]
  go "loop
  end
```

Oyunçubuğunun pozisyonu paddle ile test edilir.

keyp

Eğer bir karakter tuştakımına yazılmış ve okunmayı bekliyorsa TRUE görüntüler.

```
?to inkey
  if keyp [op rc] [op "]
  end
```

paddle

Oyunçubuğunu 0 veya 1 pozisyonuna döndürür. Oyunçubuğunun pozisyonları aşağıda belirtilmiştir:

Geri dönen değer	Açıklama
255	Hiç bir tuşa basılmamış
0	Yukarı
1	Yukarı ve sağa
2	Sağa
3	Aşağı ve sağa
4	Aşağı
5	Aşağı ve sola
6	sola
7	Yukarı ve sola

```
?paddle 0
255
```

Ateş tuşu buttonp ile test edilir.

rc

(read character) Tuştakımına ilk yazdırılan karakteri görüntüler.

```
?make "key rc
```

....arkasından X tuşuna bakın...

```
?:key
x
```

rl

(read list) Tuştakımına yazılmış bir satır içeren listeyi görüntüler; girişi bir "carriage return"nün izlemesi gerekir.

```
?make "instr-list rl
repeat 4 [fd 50 rt 90]
?:nstr list
[repeat 4 [fd 50 rt 90]
```

rq

(read quote) Tuştakımına yazılmış bir satır içeren kelimeyi görüntüler; girişi bir "carriage return"nün izlemesi gerekir.

```
?make "komut rq
repeat 3 [fd 60 rt 120]
?:komut
repeat 3 [fd 60 rt 120]
```

Ses:

Ses komutları AMSTRAD'ın Dr.LOGO'sunda tektir ve AMSTRAD BASIC ile benzeşir.

Daha fazla açıklama için Temel kursun 9. bölümüne bakınız.

sound

Ses kuyruğuna bir ses koyar. Formu şöyledir: [(kanal durumu) (ton periyodu) (süre) (volüm) (volüm envelopu) (ton envelopu) (gürültü)]. (Süre)den sonraki parametreler isteğe bağlıdır.

```
?sound [1 20 50]
```

env

Volüm envelopu oluşturur. Formu şöyledir: [(envelop numarası) (envelop bölümü)]

```
?env [1 100 2 20]
?sound [1 200 300 5 1]
```

ent

Ton envelopu oluşturur. Formu şöyledir: [(envelop numarası) (envelop bölümü)]

```
?ent [1 100 2 20]
?sound [1 200 300 5 1 1]
```

release

sound komutuyla üst duruma ayarlanmış ses kanallarını bıraktırır. Kanal bırakma durumları şöyledir:

Giriş değeri	Bırakılan kanal
0	Hiçbiri
1	A
2	B
3	A ve B
4	C
5	A ve C
6	B ve C
7	A, B ve C

```
?release 1
```

Kontrol Akışı:

bye

Dr.LOGO bölümünden çıkış

```
?bye
```

co

CONTROL Z, pause veya ERRACT tarafından konmuş arayı sonlandırır.

```
?co
```

go

Aynı giriş kelimesi ile birlikte, label ifadesini takip eden o anki prosedürü çalıştırır.

```
> go "loop
```

if

Giriş ifadesindeki değere bağlı iki kullanım listesini bir ifade olarak çalıştırır; giriş kullanım listeleri çift taraflı köşeli parantez içine alınmalıdır.

```
> if (a > b) [pr [a büyüktür]]
> [pr ]b büyüktür]]
```

label

Giriş kelimesi ile birlikte go ifadesinden sonra çalıştırılacak satırı belirtir.

```
> label "loop
```

op

(output) Giriş nesnesinin, o anda mevcut prosedür'e göre çıkışını yapar.

```
?op [result]
```

repeat

Giriş ifade listesinde belirtilen tekrar sayısını başlatır.

```
?repeat 4 [fd 50 rt 90]
```

run

Girilen ifade listesini başlatır.

```
?make "instr-list [fd 40 rt 90]
?run:instr-list
```

stop

Kullanılan prosedürü durdurur ve TOPLEVEL'e veya prosedür çağırmaya gider (? çıkar).

```
?stop
```

wait

Çalışan prosedürü giriş sayısında belirtilen zaman kadar bekletir. Bekleme zamanı = girilen sayı * 1/60 saniye.

```
?wait 200
```

Özel Kullanımlar:

catch

Giriş ifade listesinin çalışması sırasında oluşan hata ve özel şartları yakalar.

```
> catch "hata [+ [][]
> pr [buradayım]
buradayım
```

error

En sık hata yaratan elemanların listesini görüntüler.

```
catch "hata [hatadan.itibaren.yap]
show error
```

*

notrace

Prosedür çalışmasında tarama görüntülemesini kapatır. (trace'e bakınız)

```
?notrace
```

*

nowatch

Bütün veya belirlenen prosedürlerde izleme görüntülemesini kapatır (watch'a bakınız).

```
?nowatch
```

pause

Araya edit veya tercüme işlemini konması için programı işlevini durdurur.

```
> if:size > 5 [pause]
```

throw

önceden catch ifadesiyle belirtilmiş giriş satırını çalıştırır.

```
?throw "TOPLEVEL
```

*

trace

Prosedür çalışmasında tarama görüntülemesini açar.

```
?trace
```

*

watch

Bütün veya belirlenen prosedürlerin izleme görüntülemesini açar.

```
?watch
```

Sistem İlkeleri:

.contents

Dr.LOGO sembol boşluğunun içerdiklerini gösterir.

.deposit

İkinci girilen sayıyı, birinci girilen sayı tarafından belirtilen yerdeki mutlak hafızaya yerleştirir.

.examine

Belirtilen yerdeki mutlak hafızayı gösterir.

*

.in

Girişi belirlenmiş kapıdaki aktif değerleri tekrar dener.

*

.out

Giriş değerini, giriş belirlenmiş kapıya gönderir.

Sistem Değişkenleri:

ERRACT

TRUE ise bir ara yaratır, bariz hatada TOPLEVEL'e döner.

FALSE

Sistem değişkeni

REDEFP

TRUE ise ilkellerin tekrar belirlenmesine döner.

TOPLEVEL

throw "TOPLEVEL bütün belirlenmiş prosedürlerden çıkar.

TRUE

Sistem değişkeni

Sistem Desteklemeleri:

.APV

Destekleme değerine bağlıdır; global değişkenlerin değeridir.

.DEF

Prosedürün tanımlamasıdır.

*

.ENL

Bir "carriage return" veya boşluk ile kırılmış prosedür satırının sonudur.

*

.EMT

Bir "carriage return" veya boşluk ile kırılmış prosedür satırının başlangıcıdır.

.PRM

İlkel belirtir.

*

.REM (or;)

Uyarı veya yorumlar.

KONU 7
Bilginiz İçin

Bu derste bu bilgisayarı kullanmayı öğrenirken ihtiyacınız olan birçok başvuru bilgileri içerir.
Konular
* Kursör yerleri ve kontrol kod uzantıları
* Kesintiler (interrupt)
* ASCII ve grafik karakterler
* Tuş referansları
* Ses
* Hata mesajları
* BASIC anahtar kelimeler
* Plancılar
* Bağlantılar
* Yazıcı
* Oyun çubuğu
* Disk organizasyonu
* Yerleşmiş sistem uzantıları (RSX)
* Bellek
* CP/M Plus terminal benzeticisi (emülatör)
* CP/M Plus karakter seti

Bölüm 1: BASIC Kursör Yerleri ve Kontrol Kod Uzantıları

Değişik uygulama programlarında metin kursörüne hali hazırdaki pencerenin dışında bir yer verilebilinir. Değişik operasyonlar gerçekleşmeden önce kursörü geçerli bir pozisyona zorlar; bunlar şöyledir:

1. Karakter yazarken

2. Kursör kabarığı çizerken

3. Listenin başındaki yıldız ile işaretli kontrol kodlarına uyarken.

Kursörün geçerli bir pozisyona zorlamak için yöntem şöyledir:

1. Eğer kursör sağ kenarın sağında ise bir sonraki alt satırın en sol kolonuna hareket ettirilir.

2. Eğer kursör sol kenarın solunda ise bir üst satırın en sağ kolonuna götürülür.

3. Eğer kursör üst kenarın üstünde ise pencere bir satır aşağıya yuvarlanır ve kursör pencerenin üst satırına ayarlanır.

4. Eğer kursör alt kenarın altında ise pencere bir satır üste yuvarlanır ve kursör pencerenin alt satırına ayarlanır.

Testler ve operasyonlar verilen kaide içinde yapılır. Uygunsuz kursör pozisyonları pencerenin alt veya solunun dışında sıfır veya negatif olabilir.

Yazı ekranına gönderilen Ø ile 31 arasındaki karakter değerleri ekranda karakter üretmez fakat kontol kodları olarak tercüme edilir. (tedbirsizce uygulanmamalıdır) Bazı kodlar, kod parametreleri olan bazı karakterlerin anlamını değiştirir.

Grafik ekrana gönderilen kontrol kodu ancak fonksiyonu ile ilgili sembolleri eğer tuştakımından girilmişse yazar. (örneğin: &0R "BEL"-(CTRL)2) Eğer komut formundan adreslerse kontrol fonksiyonunu işletir.

PRINT CHR$ (&07), veya PRINT " " (sembolü PRINT cümlesinde (CTRL) G'ye basarak elde edilir.)

Yıldız * işareti ile işaretlenmiş kodlar uygulanmadan önce kursörü şimdiki pencerenin nizami bir pozisyonuna zorlar fakat kursörü gayri nizami bir pozisyonda bırakabilir.

BASIC Kontrol Karakterleri

Değeri Adı	Parametre	Anlamı
& 0 0 0 NUL		Etkisi yok ihmal edildi
& 01 1 SOH	0-255	Parametre değeri tarafından verilen sembolü yaz. Bu 0 ile 31 arası sembollerin görüntülenmesine izin verir.
& 02 2 STX		Yazı kursörünü kapat kullanıcı anahtar parametre değeri 0 olan CURSOR komutuna eşdeğerdir.
& 03 3 ETX		Yazı kursörünü aç. Kullanıcı anahtar parametre değeri 1 olan CURSOR komutuna eşdeğerdir.
& 04 4 EOT	0-2	Ekran modunu ayarlar. MOD4'den parametre alınır. MODE komutuna eşdeğerdir.
& 05 5 ENQ	0-255	Parametre karakterini grafik kursöre gönderir.
& 06 6 ACK		Yazı ekranına imkan verir (&15 NAK'a bakınız)
& 07 7 BEL		Bip sesi. Bu, ses dizilerini çaktırır.
& 08 8 *BS		Kursörü bir karakter geri götürür.
& 09 9 *TAB		Kursörü bir karakter ileri götürür.
& OA 10 *LF		Kursörü bir satır aşağıya indirir.
& OB 11 *VT		Kursörü bir satır yukarı çıkarır.
& OC 12 FF		Metin penceresini temizler ve kursörün sol üst köşeye götürür. CLS komutunun eşdeğeridir.

Değeri Adı	Parametre	Anlamı
& OD 13 *CR		Kursörü pencerenin sol kenarında hali hazırda kullanılan satırın üzerine götürür.
& OE 14 SO	0-15	Sayfa ink'ini ayarlar. MOD 16'dan parametre alınır. PAPER komutunun eşdeğeridir.
& OF 15 SI	0-15	Pen ink'ini ayarlar. MOD 16'dan parametre alınır. PEN komutunun eşdeğeridir.
& 10 16 *DLE		Hali hazırdaki karakteri siler. Karakter hücresini sayfa ink'i ile doldurur.
& 11 17 *DC1		Pencerenin sol kenarından hali hazırdaki karakter dahil temizler. Bütün işlem yapılan karakter hücrelerini sayfa ink'i ile doldurur.
& 12 18 *DC2		Hali hazırdaki karakter pozisyonu dahil pencerenin sağ kenarına kadar temizler. Etkilenen karakter hücrelerini sayfa ink'i ile doldurur.
& 13 FG *DC3		Pencerinin başlangıcından karakter yeri dahil siler
& 14 20 *DC4		Hali hazırdaki karakter pozisyonu dahil pencerenin sonuna kadar temizler. Etkilenen hücreleri sayfa ink'i ile doldurur.
& 15 21 NAK		Metin ekranını kapatır. Ekran ACK (& 06 6) gönderilinceye kadar, herhangi bir şeye tepki göstermez.
& 16 22 SYN	0-1	MOD 2 parametresi. Transperant (şeffaf) seçeneği. 0 disables, 1 enasles.

Değeri Adı	Parametre	Anlamı
& 17 23 ETB	0-3	MOD 4 parametresi. 0 normal grafik ink moduna, 1 XOR grafik ink moduna, 2 AND grafik ink moduna, 3 OR grafik ink moduna ayarlar.
& 18 24 CAN		Pen ve sayfa inklerini değiştirir.
& 19 25 EM	0-255 0-255 0-255 0-255 0-255 0-255 0-255 0-255 0-255	Kullanıcı tanımlanabilir. Karakter için matris ayarlar. SYMBOL komutuna eşdeğerdir. 9 parametre alır, ilk parametre hangi karakterin matrisi ayarlanacaksa onu belirler. Diğer 8'i matrisi belirler. İlk bayt'ın anlamlı biti karakter hücresinin üst sol pixeli, son baytın en az anlamlı biti karakter hücresinin alt sağ pixelidir.
& 1A 26 SUB	1-80 1-80 1-25 1-25	Pencere ayarlar. WINDOW komutunun eşdeğeridir. İlk 2 parametre pencerenin sağ ve sol kenarlarıdır. Küçük değer sol, büyük değer sağ kenardır. Son 2 parametre de küçük değer üst, büyük değer alt kenar kenardır.
& 1B 27 ESC		Etkisi yok ihmal edildi.
& 1C 28 FS	0-15 0-31 0-31	Ink'i bir renk çiftine ayarlar. INK komutunun eşdeğeridir. İlk parametre (MOD 16) ink'i belirtir, diğer ikisi (MOD 32) gereken renklerdir. (27'den 31'e kadar olan parametre değerleri tanımlanmamış renklerdir)
& 1D 29 GS	0-31 0-31	Çerçeveyi bir renk çiftine ayarlar. BORDER komutu ile aynıdır. Parametreler (MOD 32) iki rengi belirler. (27'den 31'e kadar olan parametre değerleri tanımlanmamış renklerdir.)
& 1E 30 RS		Kursörü pencerenin en sol köşesine götürür.

& 1F 31 US	1-80 1-25	Kursörü hali hazırdaki pencerenin içinde verilen bir pozisyona götürür. LOCATE ile aynıdır. 1. parametre kolonu, 2. parametre satırı belirler.

6128'in işletilmesi, karmaşık, gerçek zaman çalışma sistemi tarafından önceden hazırlanmıştır. Çalışma sistemi bilgisayardaki giriş ve çıkış trafiğini yönetir.

Öncelikle donanım ve BASIC çeviricisi arasında bağlantı kurulur, (örneğin; ink çakma (flashing) fonksiyonu ki BASIC parametreleri geçirir) ve çalışma sistemi iş ile birlikte çalışır. Bir kısmı gerekeni hesaplarken diğer kısmı bu olayların zamanını hesaplar.

Makine çalışma sistemi genellikle "firmware" olarak nitelendirilebilir ve BASIC'deki yüksek seviye komutları tarafından çağrılan makine kodu dizileri ihtiva eder.

Eğer siz makina bellek adresinde birşey araştırmayı tecrübe etmek isterseniz (POKE) veya altprogramları çağırmayı (CALL) denemelisiniz. Bunu yapmadan önce programlarınızı koruyun ve listeleyin yoksa pişman olursunuz. 6128'in geniş firmware çalışma sistemi SOFT G68'de tarif edilmiştir ve bu tanıtıcı kitabın gayesi dışında tutulmuştur.

Geniş anlamda makina kodlarını kullanan bir program için assembler kullanmak gereklidir. DEVPAC assembler, Z80 editörü, disassembleri ve monitörü ile birlikte tekrar yerleştirilebilir assembler içerir.

Bölüm 2: Kesintiler (İnterrupts)

Bu kitapçığın daha evvelki konularında tarif edilmiş AFTER ve EVERY yapıları tarafından örneklenmiş bir kaç çoklu işlemlerini içeren çalışma sistemini kapsayan Z80 kesintilerinin geniş kullanımını 6128 yapar. Olay zamanlayıcılarının başta gelenleri:

Break ((ESC) (ESC))
Timer 3
Timer 2 (ve 3 ses kanalı dizisi)
Timer 1
Timer 0

Aralar kesinti noktasındaki muhtemel orta değişken durumlarının neticesi ele alındıktan sonra ilave edilmelidir. Ara alt programının kendisi, ana programdaki değişken durumları ile istenmeyen tesirlerini yok etmelidir.

Ses dizileri, eşit öncelikli bağımsız aralıklara sahiptir. Birkez ses aralığı başladığında herhangi bir diğer ses aralığı tarafından kesilmez. Bu başlatılmış ses aralığı dizileri yukarıda bahsedilmiş etkilerden muaf değişkenleri paylaşır.

Ses dizilerinin kesintisine yol verilmişse (ON SQ GOSUB kullanarak), eğer bu kanal için ses dizesi dolu değilse hemen keser, aksi takdirde ses sona erince keser ve dizede daha fazlası için yer vardır. Ara verme hareketi hadiseyi zayıflatır, böylece altprogram, eğer başka kesintiler gerekiyorsa.....
.....

Bölüm 3: BASIC ASCII ve Grafik Karakterleri

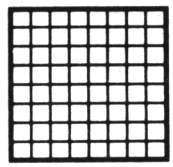
ASCII

Aşağıdaki tablo, uygun ASCII kodlarıyla birlikte ondalık sekizlik (octal), onaltılık (hex) notasyonlarını kullanan ASCII referans karakter setini vermektedir.

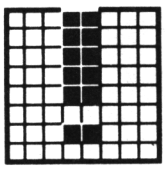
DEC	OCTAL	HEX	ASCII characters
0	000	00	NUL ([CTRL]@)
1	001	01	SOH ([CTRL]A)
2	002	02	STX ([CTRL]B)
3	003	03	ETX ([CTRL]C)
4	004	04	EOT ([CTRL]D)
5	005	05	ENQ ([CTRL]E)
6	006	06	ACK ([CTRL]F)
7	007	07	BEL ([CTRL]G)
8	010	08	BS ([CTRL]H)
9	011	09	HT ([CTRL]I)
10	012	OA	LF ([CTRL]J)
11	013	OB	VT ([CTRL]K)
12	014	OC	FF ([CTRL]L)
13	015	OD	CR ([CTRL]M)
14	016	OE	SO ([CTRL]N)
15	017	OF	SI ([CTRL]O)
16	020	10	DLE ([CTRL]P)
17	021	11	DC1 ([CTRL]Q)
18	022	12	DC2 ([CTRL]R)
19	023	13	DC3 ([CTRL]S)
20	024	14	DC4 ([CTRL]T)
21	025	15	NAK ([CTRL]U)
22	026	16	SYN ([CTRL]V)
23	027	17	ETB ([CTRL]W)
24	030	18	CAN ([CTRL]X)
25	031	19	EM ([CTRL]Y)
26	032	1A	SUB ([CTRL]Z)
27	033	1B	ESC
28	034	1C	FS
29	035	1D	GS
30	036	1E	RS
31	037	1F	US
32	040	20	SP
33	041	21	!
34	042	22	"
35	043	23	#
36	044	24	$
37	045	25	%
38	046	26	&
39	047	27	'
40	050	28	(
41	051	29	)
42	052	2A	*
43	053	2B	+
44	054	2C	,
45	055	2D	-
46	056	2E	.
47	057	2F	/
48	060	30	0
49	061	31	1
50	062	32	2
51	063	33	3
52	064	34	4
53	065	35	5
54	066	36	6
55	067	37	7
56	070	38	8
57	071	39	9
58	072	3A	:
59	073	3B	;
60	074	3C	<
61	075	3D	=
62	076	3E	>
63	077	3F	?
64	100	40	@
65	101	41	A
66	102	42	B
67	103	43	C
68	104	44	D
69	105	45	E
70	106	46	F
71	107	47	G
72	110	48	H
73	111	49	I
74	112	4A	J
75	113	4B	K
76	114	4C	L
77	115	4D	M
78	116	4E	N
79	117	4F	O
80	120	50	P
81	121	51	Q
82	122	52	R
83	123	53	S
84	124	54	T
85	125	55	U
86	126	56	V
87	127	57	W
88	130	58	X
89	131	59	Y
90	132	5A	Z
91	133	5B	[
92	134	5C	\
93	135	5D	]
94	136	5E	↑
95	137	5F	←
96	140	60	`
97	141	61	a
98	142	62	b
99	143	63	c
100	144	64	d
101	145	65	e
102	146	66	f
103	147	67	g
104	150	68	h
105	151	69	i
106	152	6A	j
107	153	6B	k
108	154	6C	l
109	155	6D	m
110	156	6E	n
111	157	6F	o
112	160	70	p
113	161	71	q
114	162	72	r
115	163	73	s
116	164	74	t
117	165	75	u
118	166	76	v
119	167	77	w
120	170	78	x
121	171	79	y
122	172	7A	z
123	173	7B	{
124	174	7C	|
125	175	7D	}
126	176	7E	~

Makinaya Özel BASIC Grafik Karakter Seti

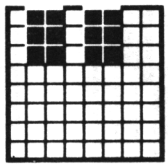
Burada üretilen karakterler, 6128'in ekranına yazmakta kullanılan 8 × 8 hücre marisinin üzerine çizilmiştir.



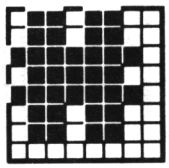
32 &H20
&X00100000



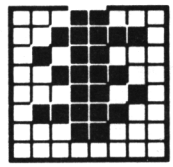
33
&H21
&X00100001



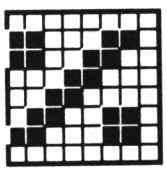
34
&H22
&X00100010



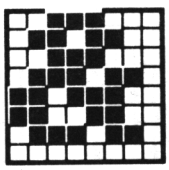
35
&H23
&X00100011



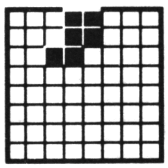
36
&H24
&X00100100



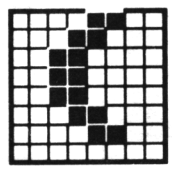
37
&H25
&X00100101



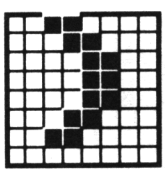
38
&H26
&X00100110



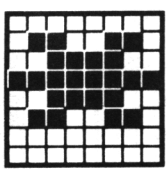
39
&H27
&X00100111



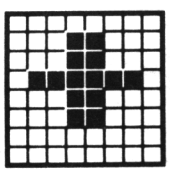
40
&H28
&X00101000



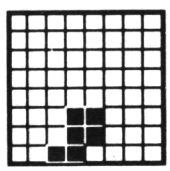
41
&H29
&X00101001



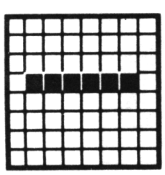
42
&H2A
&X00101010



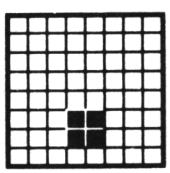
43
&H2B
&X00101011



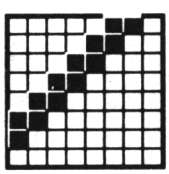
44
&H2C
&X00101100



45
&H2D
&X00101101



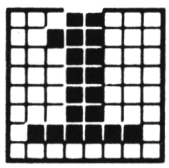
46
&H2E
&X00101110



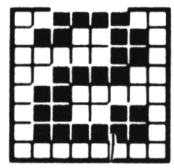
47
&H2F
&X00101111



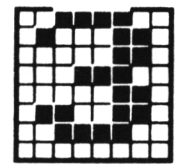
48
&H30
&X00110000



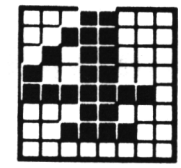
49
&H31
&X00110001



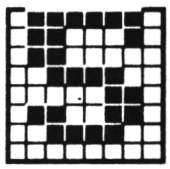
50
&H32
&X00110010



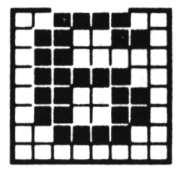
51
&H33
&X00110011



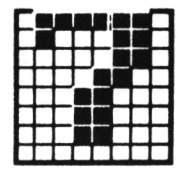
52
&H34
&X00110100



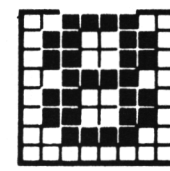
53
&H35
&X00110101



54
&H36
&X00110110



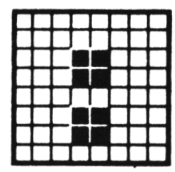
55
&H37
&X00110111



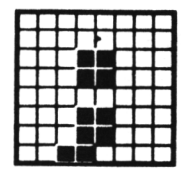
56
&H38
&X00111000



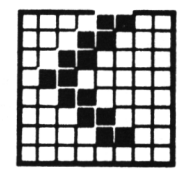
57
&H39
&X00111001



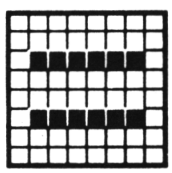
58
&H3A
&X00111010



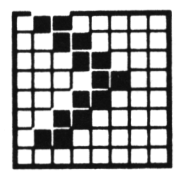
59
&H3B
&X00111011



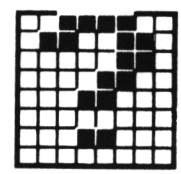
60
&H3C
&X00111100



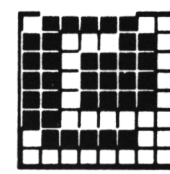
61
&H3D
&X00111101



62
&H3E
&X00111110



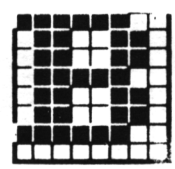
63
&H3F
&X00111111



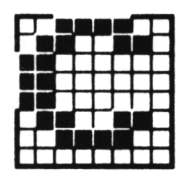
64
&H40
&X01000000



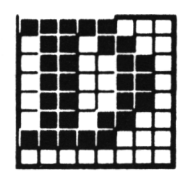
65
&H41
&X01000001



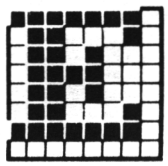
66
&H42
&X01000010



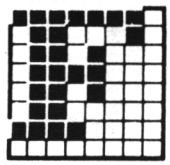
67
&H43
&X01000011



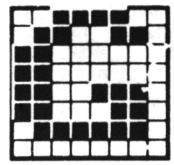
68
&H44
&X01000100



69
&H45
&X01000101



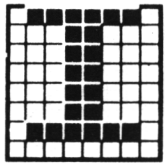
70
&H46
&X01000110



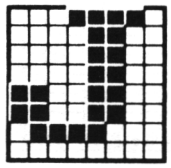
71
&H47
&X01000111



72
&H48
&X01001000



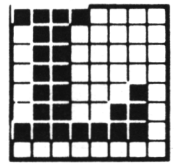
73
&H49
&X01001001



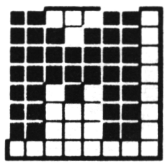
74
&H4A
&X01001010



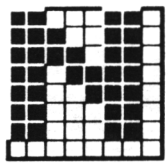
75
&H4B
&X01001011



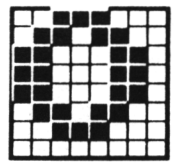
76
&H4C
&X01001100



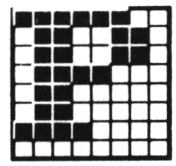
77
&H4D
&X01001101



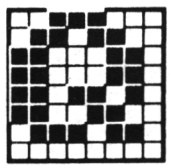
78
&H4E
&X01001110



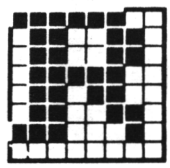
79
&H4F
&X01001111



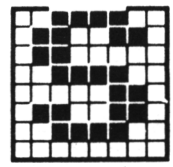
80
&H50
&X01010000



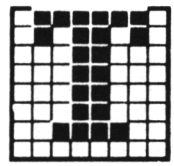
81
&H51
&X01010001



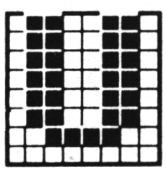
82
&H52
&X01010010



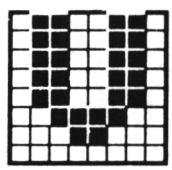
83
&H53
&X01010011



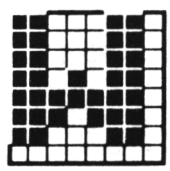
84
&H54
&X01010100



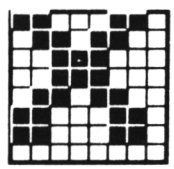
85
&H55
&X01010101



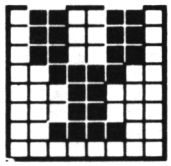
86
&H56
&X01010110



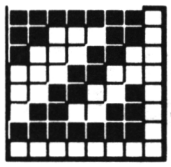
87
&H57
&X01010111



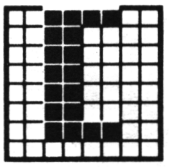
88
&H58
&X01011000



89
&H59
&X01011001



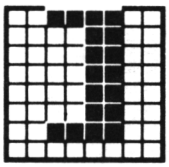
90
&H5A
&X01011010



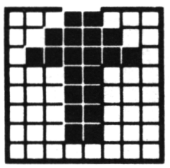
91
&H5B
&X01011011



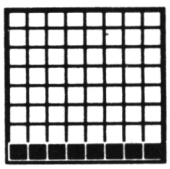
92
&H5C
&X01011100



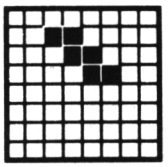
93
&H5D
&X01011101



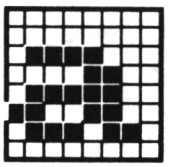
94
&H5E
&X01011110



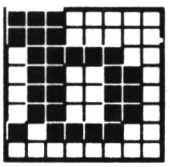
95
&H5F
&X01011111



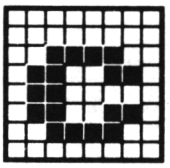
96
&H60
&X01100000



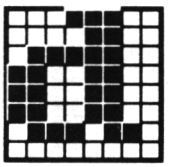
97
&H61
&X01100001



98
&H62
&X01100010



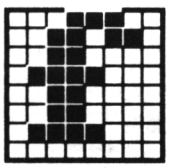
99
&H63
&X01100011



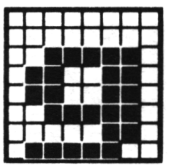
100
&H64
&X01100100



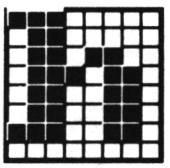
101
&H65
&X01100101



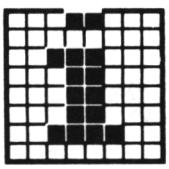
102
&H66
&X01100110



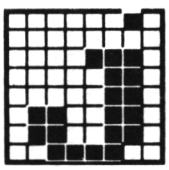
103
&H67
&X01100111



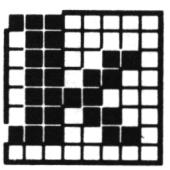
104
&H68
&X01101000



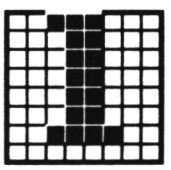
105
&H69
&X01101001



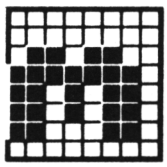
106
&H6A
&X01101010



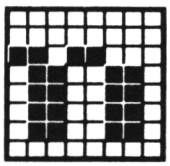
107
&H6B
&X01101011



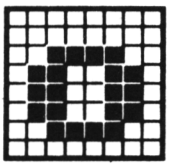
108
&H6C
&X01101100



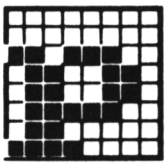
109
&H6D
&X01101101



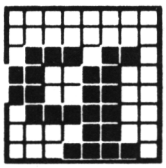
110
&H6E
&X01101110



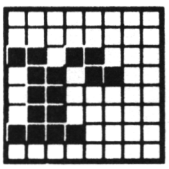
111
&H6F
&X01101111



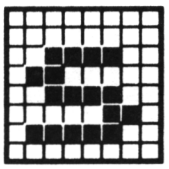
112
&H70
&X01110000



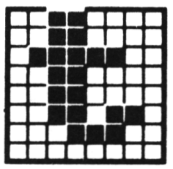
113
&H71
&X01110001



114
&H72
&X01110010



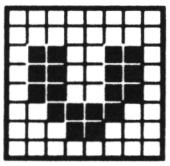
115
&H73
&X01110011



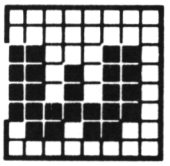
116
&H74
&X01110100



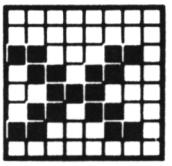
117
&H75
&X01110101



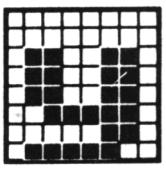
118
&H76
&X01110110



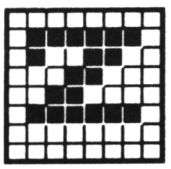
119
&H77
&X01110111



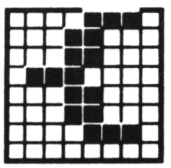
120
&H78
&X01111000



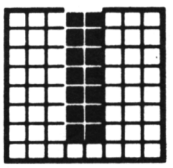
121
&H79
&X01111001



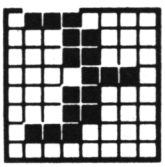
122
&H7A
&X01111010



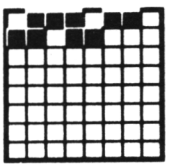
123
&H7B
&X0111101?



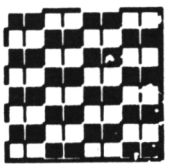
124
&H7C
&X01111100



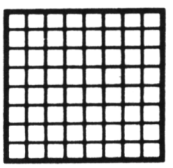
125
&H7D
&X01111101



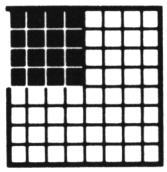
126
&H7E
&X01111110



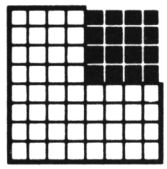
127
&H7F
&X01111111



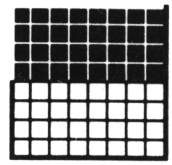
128
&H80
&X10000000



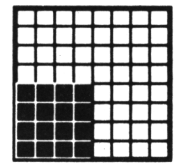
129
&H81
&X10000001



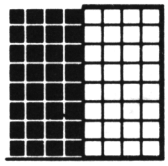
130
&H82
&X10000010



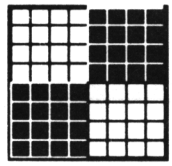
131
&H83
&X10000011



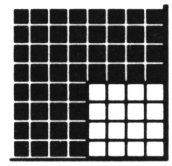
132
&H84
&X10000100



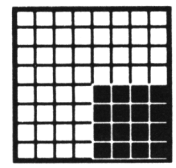
133
&H85
&X10000101



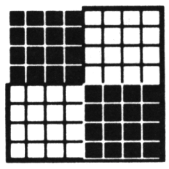
134
&H86
&X10000110



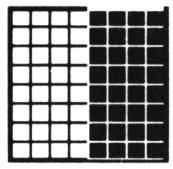
135
&H87
&X10000111



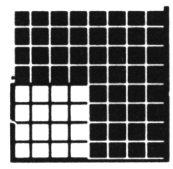
136
&H88
&X10001000



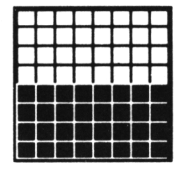
137
&H89
&X10001001



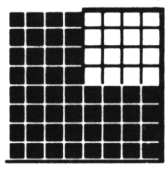
138
&H8A
&X10001010



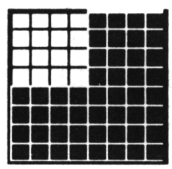
139
&H8B
&X10001011



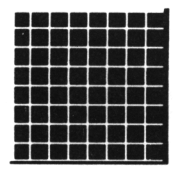
140
&H8C
&X10001100



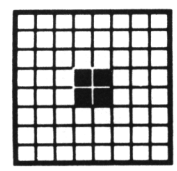
141
&H8D
&X10001101



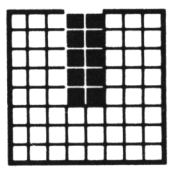
142
&H8E
&X10001110



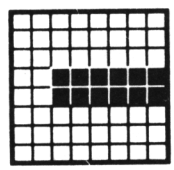
143
&H8F
&X10001111



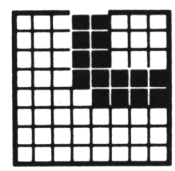
144
&H90
&X10010000



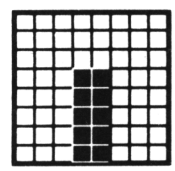
145
&H91
&X10010001



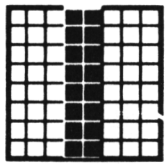
146
&H92
&X10010010



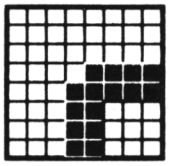
147
&H93
&X10010011



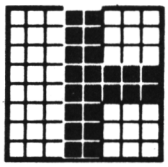
148
&H94
&X10010100



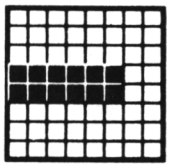
149
&H95
&X10010101



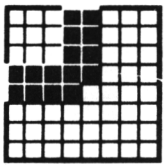
150
&H96
&X10010110



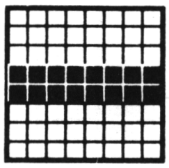
151
&H97
&X10010111



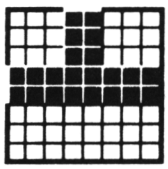
152
&H98
&X10011000



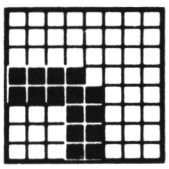
153
&H99
&X10011001



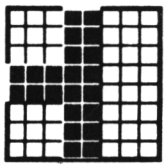
154
&H9A
&X10011010



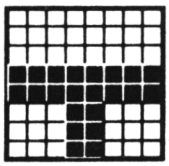
155
&H9B
&X10011011



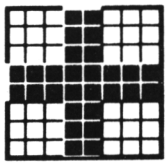
156
&H9C
&X10011100



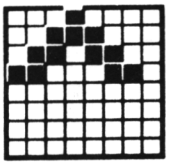
157
&H9D
&X10011101



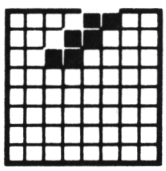
158
&H9E
&X10011110



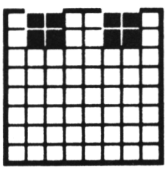
159
&H9F
&X10011111



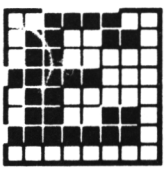
160
&HA0
&X10100000



161
&HA1
&X10100001



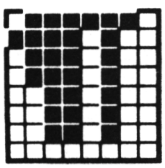
162
&HA2
&X10100010



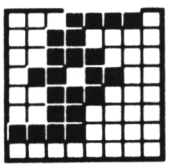
163
&HA3
&X10100011



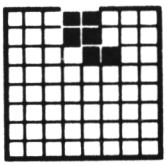
164
&HA4
&X10100100



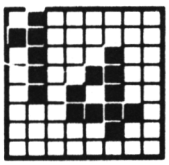
165
&HA5
&X10100101



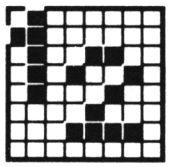
166
&HA6
&X10100110



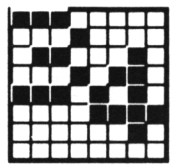
167
&HA7
&X10100111



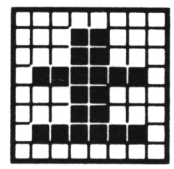
168
&HA8
&X10101000



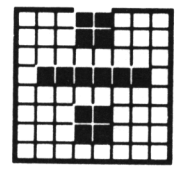
169
&HA9
&X10101001



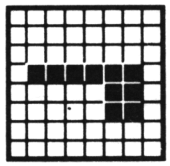
170
&HAA
&X10101010



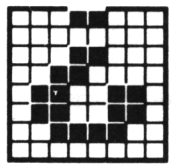
171
&HAB
&X10101011



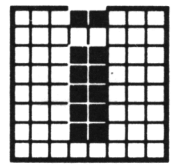
172
&HAC
&X10101100



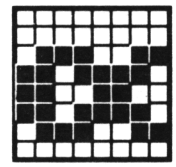
173
&HAD
&X10101101



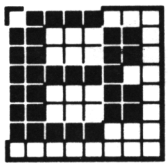
174
&HAE
&X10101110



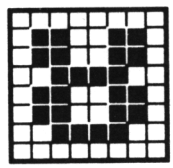
175
&HAF
&X10101111



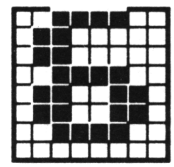
176
&HB0
&X10110000



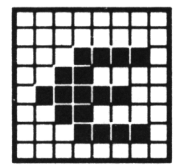
177
&HB1
&X10110001



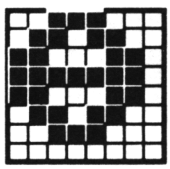
178
&HB2
&X10110010



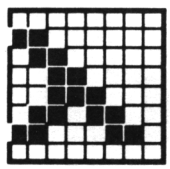
179
&HB3
&X10110011



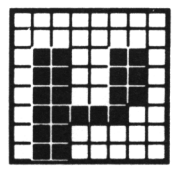
180
&HB4
&X10110100



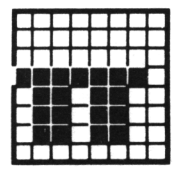
181
&HB5
&X10110101



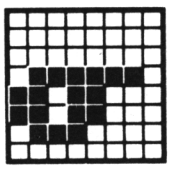
182
&HB6
&X10110110



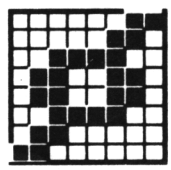
183
&HB7
&X10110111



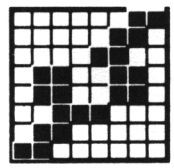
184
&HB8
&X10111000



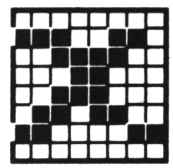
185
&HB9
&X10111001



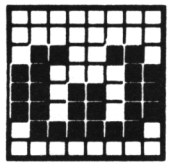
186
&HBA
&X10111010



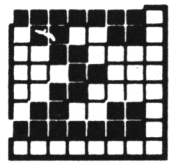
187
&HBB
&X10111011



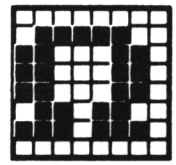
188
&HBC
&X10111100



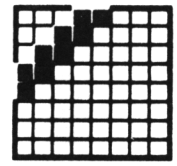
189
&HBD
&X10111101



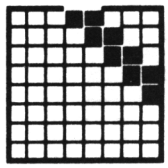
190
&HBE
&X10111110



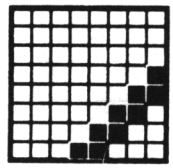
191
&HBF
&X10111111



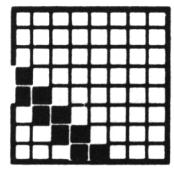
192
&HC0
&X11000000



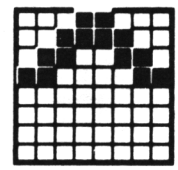
193
&HC1
&X11000001



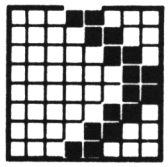
194
&HC2
&X11000010



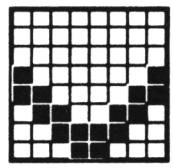
195
&HC3
&X11000011



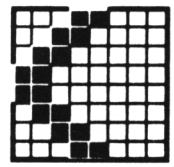
196
&HC4
&X11000100



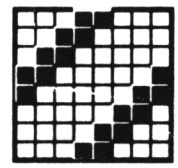
197
&HC5
&X11000101



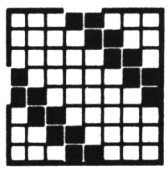
198
&HC6
&X11000110



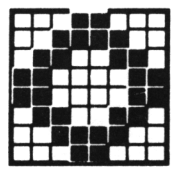
199
&HC7
&X11000111



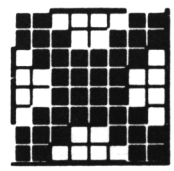
200
&HC8
&X11001000



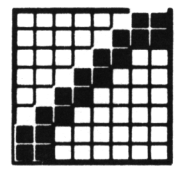
201
&HC9
&X11001001



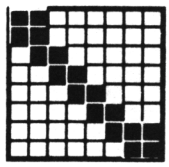
202
&HCA
&X11001010



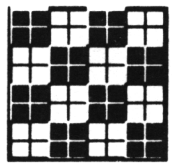
203
&HCB
&X11001011



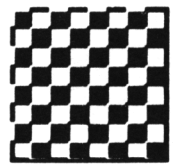
204
&HCC
&X11001100



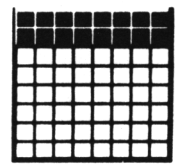
205
&HCD
&X11001101



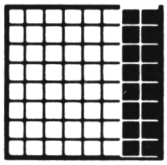
206
&HCE
&X11001110



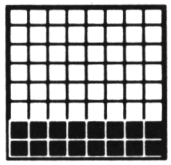
207
&HCF
&X11001111



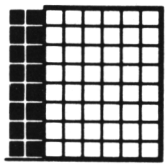
208
&HD0
&X11010000



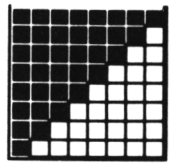
209
&HD1
&X11010001



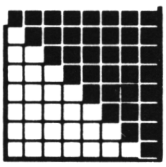
210
&HD2
&X11010010



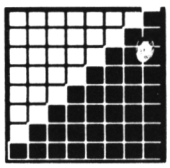
211
&HD3
&X11010011



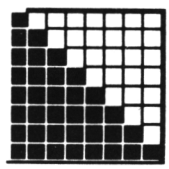
212
&HD4
&X11010100



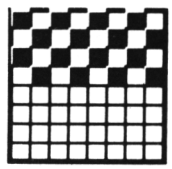
213
&HD5
&X11010101



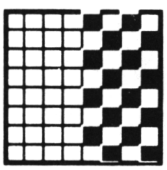
214
&HD6
&X11010110



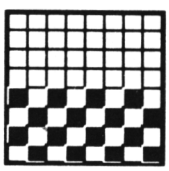
215
&HD7
&X11010111



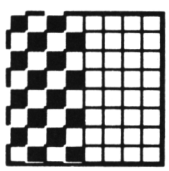
216
&HD8
&X11011000



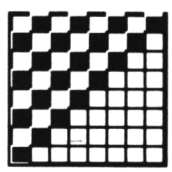
217
&HD9
&X11011001



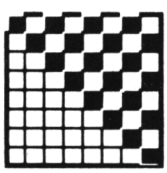
218
&HDA
&X11011010



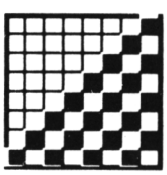
219
&HDB
&X11011011



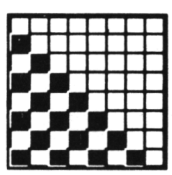
220
&HDC
&X11011100



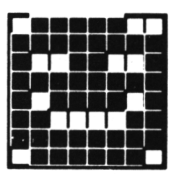
221
&HDD
&X11011101



222
&HDE
&X11011110



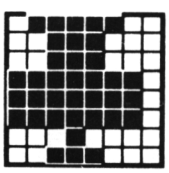
223
&HDF
&X11011111



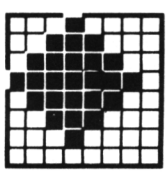
224
&HE0
&X11100000



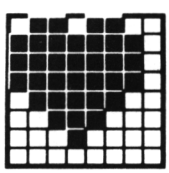
225
&HE1
&X11100001



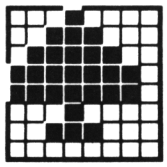
226
&HE2
&X11100010



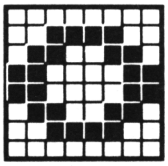
227
&HE3
&X11100011



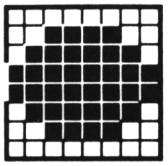
228
&HE4
&X11100100



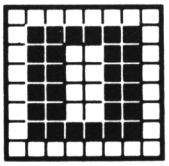
229
&HE5
&X11100101



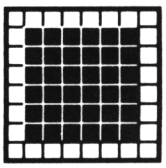
230
&HE6
&X11100110



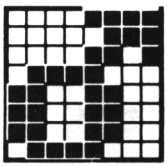
231
&HE7
&X11100111



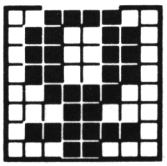
232
&HE8
&X11101000



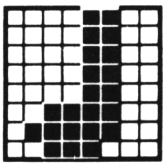
233
&HE9
&X11101001



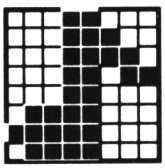
234
&HEA
&X11101010



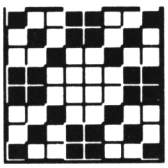
235
&HEB
&X11101011



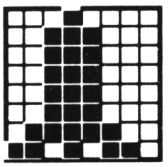
236
&HEC
&X11101100



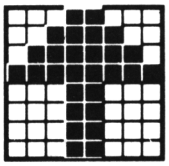
237
&HED
&X11101101



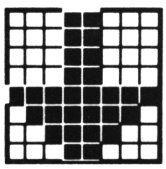
238
&HEE
&X11101110



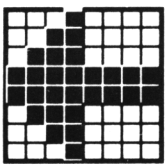
239
&HEF
&X11101111



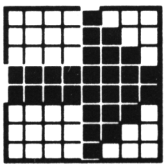
240
&HF0
&X11110000



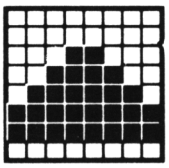
241
&HF1
&X11110001



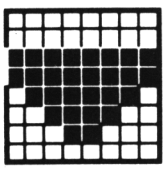
242
&HF2
&X11110010



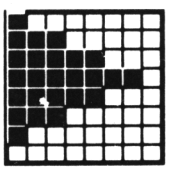
243
&HF3
&X11110011



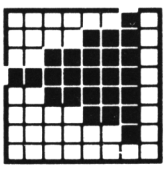
244
&HF4
&X11110100



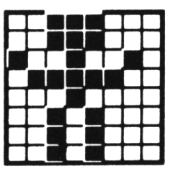
245
&HF5
&X11110101



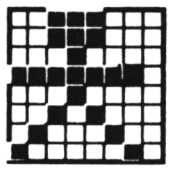
246
&HF6
&X11110110



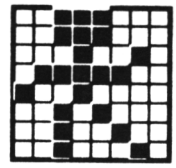
247
&HF7
&X11110111



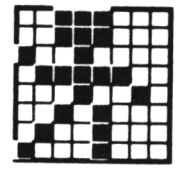
248
&HF8
&X11111000



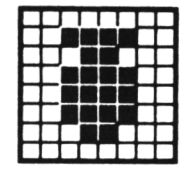
249
&HF9
&X11111001



250
&HFA
&X11111010



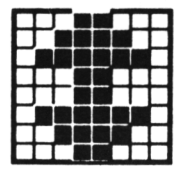
251
&HFB
&X11111011



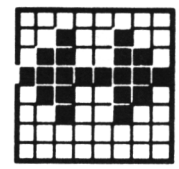
252
&HFC
&X11111100



253
&HFD
&X11111101



254
&HFE
&X11111110

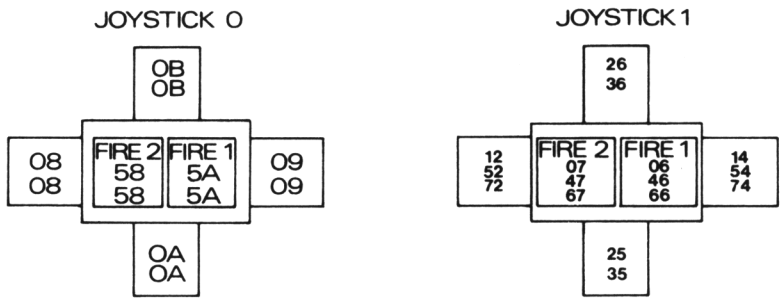


255
&HFF
&X11111111

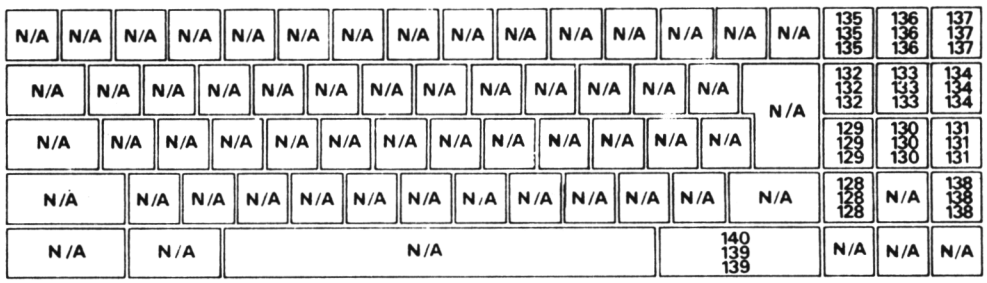
Bölüm 4: Tuş Referansları

Önceden Belirlenmiş ASCII Değerleri (HEX)

N/A	21 31	7E 22 32	23 33	24 34	25 35	26 36	27 37	28 38	29 39	1F 5F 30	3D 2D	1E A3 5E	10 10 10	7F 7F 7F	N/A	N/A	N/A
E1 09 09	11 51 71	17 57 77	05 45 65	12 52 72	14 54 74	19 59 79	15 55 75	09 49 69	0F 4F 6F	10 50 70	00 7C 40	1B 7B 5B	0D 0D 0D		N/A	N/A	N/A
N/A	01 41 61	13 53 73	04 44 64	06 46 66	07 47 67	08 48 68	0A 4A 6A	0B 4B 6B	0C 4C 6C	2A 3A	2B 3B	1D 7D 5D			N/A	N/A	N/A
N/A	1A 5A 7A	18 58 78	03 43 63	16 56 76	02 42 62	0E 4E 6E	0D 4D 6D	3C 2C	3E 2E	3F 2F	1C 60 5C	N/A			N/A	F8 F4 F0	N/A
N/A	E0 E0 E0			20 20							N/A				FA F6 F2	F9 F5 F1	FB F7 F3



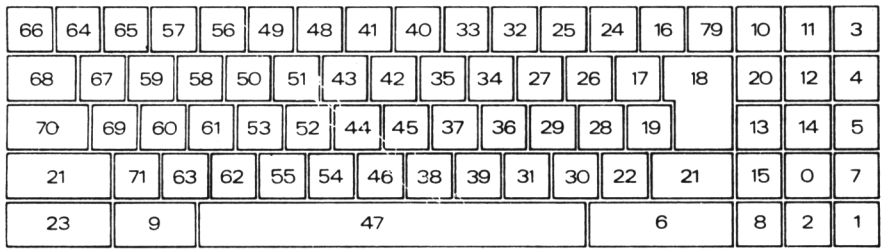
GENİŞLETİLMİŞ KARAKTERLER, TESPİT EDİLMİŞ YERLERİ VE DEĞERLERİ

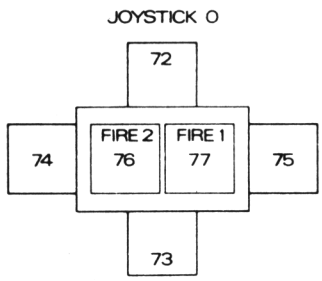


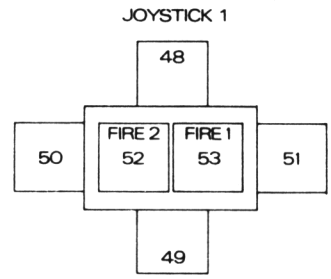
EXPANSION CHARACTER	DEFAULT SETTING	
	CHARACTER	ASCII VALUE
0 (128)	0	&30
1 (129)	1	&31
2 (130)	2	&32
3 (131)	3	&33
4 (132)	4	&34
5 (133)	5	&35
6 (134)	6	&36
7 (135)	7	&37
8 (136)	8	&38
9 (137)	9	&39
10 (138)	.	&2E
11 (139)	[RETURN]	&0D
12 (140)	RUN"[RETURN]	&52 &55 &4E &22 &0D

Note: Expansion characters 13 to 31 (141 to 159) have a null value by default. They have values assigned using the BASIC command KEY, and are assigned to keys using the command KEY DEF.

TUŞ VE OYUN ÇUBUĞU NUMARALARI







Bölüm 5: Ses

Notalar ve ton periodları

Aşağıdaki tablo her zamanki ıslah edilmiş ölçülerdeki arzu edilen ton periodlarındaki notaları sekiz oktav içinde verir.

Üretilen frekans, tam olarak gerekli frekans değildir çünkü (ton periodu) ayarları tam sayı olmalıdır. Göresel hata, gereken ve olan frekans farkının yüzde oranıdır.

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	16.352	3822	-0.007%	
C#	17.324	3608	+0.007%	
D	18.354	3405	-0.007%	
D#	19.445	3214	-0.004%	
E	20.602	3034	+0.009%	
F	21.827	2863	-0.016%	Octave -4
F#	23.125	2703	+0.009%	
G	24.500	2551	-0.002%	
G#	25.957	2408	+0.005%	
A	27,500	2273	+0.012%	
A#	29.135	2145	-0.008%	
B	30.868	2025	+0.011%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	32.703	1911	-0.007%	
C#	34.648	1804	+0.007%	
D	36.708	1703	+0.022%	
D#	38.891	1607	-0.004%	
E	41.203	1517	+0.009%	
F	43.654	1432	+0.019%	Octave -3
F#	46.249	1351	-0.028%	
G	48.999	1276	+0.037%	
G#	51.913	1204	+0.005%	
A	55.000	1136	-0.032%	
A#	58.270	1073	+0.039%	
B	61.735	1012	-0.038%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	65.406	956	+0.046%	
C#	69.296	902	+0.007%	
D	73.416	851	-0.037%	
D#	77.782	804	+0.058%	
E	82.407	758	-0.057%	
F	87.307	716	+0.019%	Octave -2
F#	92.499	676	+0.046%	
G	97.999	638	+0.037%	
G#	103.826	602	+0.005%	
A	110.000	568	-0.032%	
A#	116.541	536	-0.055%	
B	123.471	506	-0.038%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	130.813	478	+0.046%	
C#	138.591	451	+0.007%	
D	146.832	426	+0.081%	
D#	155.564	402	+0.058%	
E	164.814	379	-0.057%	
F	174.614	358	+0.019%	Octave -1
F#	184.997	338	+0.046%	
G	195.998	319	+0.037%	
G#	207.652	301	+0.005%	
A	220.000	284	-0.032%	
A#	233.082	268	-0.055%	
B	246.942	253	-0.038%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	261.626	239	+0.046%	Middle C
C#	277.183	225	-0.215%	
D	293.665	213	+0.081%	
D#	311.127	201	+0.058%	
E	329.628	190	+0.206%	
F	349.228	179	+0.019%	Octave 0
F#	369.994	169	+0.046%	
G	391.995	159	-0.277%	
G#	415.305	150	-0.328%	
A	440.000	142	-0.032%	International A
A#	466.164	134	-0.055%	
B	493.883	127	+0.356%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	523.251	119	-0.374%	
C#	554.365	113	+0.229%	
D	587.330	106	-0.390%	
D#	622.254	100	-0.441%	
E	659.255	95	+0.206%	
F	698.457	89	-0.543%	Octave 1
F#	739.989	84	-0.548%	
G	783.991	80	+0.350%	
G#	830.609	75	-0.328%	
A	880.000	71	-0.032%	
A#	932.328	67	-0.055%	
B	987.767	63	-0.435%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	1046.502	60	+0.462%	
C#	1108.731	56	-0.662%	
D	1174.659	53	-0.390%	
D#	1244.508	50	-0.441%	
E	1318.510	47	-0.855%	
F	1396.913	45	+0.574%	
F#	1479.978	42	-0.548%	Octave 2
G	1567.982	40	+0.350%	
G#	1661.219	38	+0.992%	
A	1760.000	36	+1.357%	
A#	1864.655	34	+1.417%	
B	1975.533	32	+1.134%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	2093.004	30	+0.462%	
C#	2217.461	28	-0.662%	
D	2349.318	27	+1.469%	
D#	2489.016	25	-0.441%	
E	2637.021	24	+1.246%	
F	2793.826	22	-1.685%	
F#	2959.955	21	-0.548%	Octave 3
G	3135.963	20	+0.350%	
G#	3322.438	19	+0.992%	
A	3520.000	18	+1.357%	
A#	3729.310	17	+1.417%	
B	3951.066	16	+1.134%	

Üstteki rakkamlar şu şekilde hesaplanmıştır:

```
FREQUENCY =440*(2↑(OCTAVE+((N-10)/12)))
   PERIOD =ROUND(625000/FREQUENCY)
```

N harfi C için 1, C # için 2, D için 3 v.s.

Bölüm 6: BASIC Hata Mesajları

1 Unexpected NEXT

Bir NEXT komutuna bir FOR döngüsü dışında rastlanmışsa veya NEXt komutundaki kontrol değişkeni FOR'dakiyle uyuşmuyorsa.

2 Syntax Error

Basic verilen satırı anlamamış çünkü yapısı geçerli değil.

3 Unexpected RETURN

Alt programda olmayan bir RETURN komutuna rastlanmış.

4 DATA exhausted

Son DATA'dan sonrasını okumaya kalkan bir READ komutu.

5 Improper argument

Bu genel bir hatadır. Fonksiyonun arguman değeri veya komut parametresi herhangi bir şekilde geçersizdir.

6 Overflow

Bir aritmetik işlemin sonucu taşmış. Bu floating point taşması olabilir ki bazı operasyonlarda 1.7E'38'den daha büyük değer çıkabilir. Değişik olarak bu floating point sayının 16 bitlik işaretli tam sayıya çevirirken hatalı sonuç olabilir.

7 Memory full

Çalıştırılacak program veya değişkenlerin çok büyük olması veya bir hafıza emri eğer BASIC'ın hafızasını çok fazla alçak yada olanaksız derecede yüksek değere ayarlanmaya girişilmişse bu hatayı verir. Unutmayın ki açık bir dosyanın ona ayrılmış bir buffer'ı (tamponu) vardır ve buda belleğin kullanılacağı değerleri kısıtlar

8 Line does not exit

Referans verilen satır bulunamıyor.

9 Subscript out of range

Bir matris referansında elemanlarından biri çok küçük veya çok büyük.

10 Array alraedy dimansioned

DIM bildiriminde matrislerden biri daha önceden belirtilmemiş

11 Division by zero

Bu gerçek bölmede, tam sayı bölmesinde, tam sayı yada üsleme işlemlerinde olur. 0 ile bölme bu sonucu verir.

12 Invalid direct command

Son yapılan emir girişi direkt modda geçersizdir.

13 Type mismatch

Karakter değeri gereken yere numerik bir değer verilmiştir veya tam tersi veya geçersiz bir şekilde oluşturulmuş sayı READ veya INPUT'da bulunmuştur.

14 String space full

Hafızada başka yer kalmayacak kadar karakter dizileri üretilmiş.

15 String too long

Dizi 255 karakter uzunluğunu aşmış. Birkaç dizi eklenerek üretilmiş olabilir.

16 String expression to complex

Bir karakter dizi ifadesi birkaç ara karakter dizi değerleri üretebilir. Bu değerlerin sayısı mantıklı bir limiti aşarsa bu hata ile sonuçlanır.

17 Cannot CONTiune

Bir sebepten dolayı son program CONT kullanılarak yeniden başlatılamaz. Unutyanı ki CONT, STOP komutundan sonra yeniden başlatmak içindir, (ESC) (ESC) veya bir hata ve programdaki her hangi bir değişiklik yeniden başlatmayı imkansız yapar.

18 Unknown user function

Yeni istenen FN için DEF FN işletilmemiştir.

19 RESUME missing

Hata işleme modundayken programın sonuyla karşılaşmıştır. (örneğin: ON ERROR GOTO sırasında)

20 Unexpected RESUME

RESUME sadece hata işlemi modunda geçerlidir. (yani ON ERROR GOTO sırasında)

21 Direct command found

Bir dosya yüklenirken satır numarası olmayan bir satır bulunmuştur.

22 Operand missing

BASIC tamamlanmamış bir ifadeyle karşılaşmıştır.

23 Line too long

BASIC iç formuna çevrilince çok büyük olan bir satır.

24 EOF met

Giriş kanalındaki bir dosyanın geçmiş sonunu okuma girişiminde bulunmuştur.

25 File type error

Okunan dosya uygun bir tipte değildir. OPENIN sadece ASCII yazı dosyalarını açmaya hazırlanmıştır. Aynı şekilde LOAD, RUN, vs. sadece SAVE tarafından üretilen kütük tipleriyle uğraşmak için hazırlanmıştır.

26 NEXT missing

FOR komutuna uyan bir NEXT bulunamamıştır. Bu mesajdaki satır numarası bu hatanın olduğu FOR komutunu bildirir.

27 File already open

Bir önce açılan dosya kapatılmadan OPENIN veya OPENOUT komutu uygulanmış.

28 Unknown command

Basic dışarıdan bir emre alıcı bulamıyor, örneğin baş tarafında çubuk işareti "|" olan bir komut.

29 WEND missing

WHILE komutuna uygun bir WEND bulunamış.

30 Unexpected WEND

WHILE döngüsünde olmayan bir WEND ile karşılaşıldı veya WEND son WHILE döngüsüne uymuyor.

31 File not open

(disk hataları adlı bölüme bakın.)

32 Broken in

(disk hataları adlı bölüme bakın).

AMSDOS Disk Hataları

Dosyalama operasyonlarının yürütülmesi sırasında birkaç hata görülebilir. Bu şekildeki bütün hatalar BASIC 32 hata numarası olarak ele alınır, bununla beraber çok özel bilgiler bu hata numarası gözlendiğinde DERR fonksiyonu tarafından geri döndürülür. Bu geri dönüş değerleri şöyledir:

AMSDOS hatası	DERR değeri	Hata sebebi
0	0 veya 22	(ESC) tuşuna basılmış
14	142(128 + 14)	Akış uygun bir durumda değil.
15	143(128 + 15)	Dosya için ayrılan yerin sonuna gelinmiş. miş.
16	144(128 + 16)	Genellikle yanlış bir dosya adı olan kötü komut.
17	145(128 + 17)	Daha önce mevcut bir kütük.
18	146(128 + 18)	Dosya mevcut değil.
19	147(128 + 19)	Rehber dolu.
20	148(128 + 20)	Disk dolu.
21	149(128 + 21)	Dosyalar açıkken disk değiştirilmiş.
22	150(128 + 22)	Dosya yalnız oku dur.
26	145(128 + 26)	Dosya için ayrılan yerin sonuna gelinmiş.

Eğer AMSDOS bir hata rapor etmişse bit 7 ayarlanırken DERR değeri 128 tarafından ofsett yapılır. DERR tarafından gönderilen diğer değerler disk kontrolcusu tarafından çıkartılır ve bit anlamlıdır, daima bit 6 ayarlanır. Bit 7 hatanın AMSDOS tarafından rapor edilip edilmediğini belirtir. Her bitin anlamı şöyledir.

Bit	Anlamı
0	Adres işareti yok.
1	Yazılmaz-Disk yazı korumalı.
2	Data yok-bölge bulunamadı.
3	Sürücü hazır değil-sürücüde disk yok.
4	Üst üste işletme hatası.
5	Data hatası-CRC hatası
6	Disk kontrolcusu tarafından, datayı bildirmek için daima 1'e ayarlanır.
7	AMSDOS tarafından hata halihazırda rapor edilmemiş.

Eğer dosya açılmamışken girişe teşebbüs edilirse ERR 31'e döner. ERR ve DERR kullanılırken uygulanan yol ERR'ın değerinin 31 veya 32 olduğun inceleyen kısa program çağıran bir ON ERROR GOTO ilave etmektir. Eğer 32 ise hatanın cinsi hakkında bilgi edinmek için DERR sorguya çekilir. örneğin:

```
10 ON ERROR GOTO 1000
20 OPENOUT "myfile.acs"
30 WRITE #9, "test-data"
40 CLOSEOUT
50 END
1000 amsdoserr = (DERR AND &7F):REM mask of bit 7
1010 IF ERR<31 THEN END
1020 IF ERR = 31 THEN PRINT "20.satırı doğru yazdığınıza emin misiniz?."
END
1030 IF amsdoserr = 20 THEN PRINT" disk doludur, öneri yeni bir data diski
kullanın":END
1040 IF amsdoserr = & x01001000 THEN PRINT" diski sürücüye koyun ve bir
tuşa basın":WHILE INKEY$ = " ":WEND:RESUME
1050 END
```

Bölüm 7: BASIC Komutları

Aşağıdakiler AMSTRAD CPC6128 BASIC komutlarının listesidir. Bunlar ayrılmıştır ve değişken adı olarak kullanılmazlar.

ABS, AFTER, AND, ASC, ATN, AUTO

BIN$, BORDER

CALL, CAT, CHAIN, CHR$, CINT, CLEAR, CLG, CLOSEIN, CLOSEOUT, CLS, CONT, COPYCHR$, COS, CREAL, CURSOR

DATA, DEC$, DEF, DEFINT, DEFREAL, DEFSTR, DEG, DELETE, DERR, DI, DIM, DRAW, DRAWR

EDIT, EI, ELSE, END, ENT, ENW, EOF, ERASE, ERL, ERR, ERROR, EVERY, EXP

FILL, FIX, FN, FOR, FRAME, FRE

GOSUB, GOTO, GRAPHICS

HEX$, HIMEM

IF, INK, INKEY, INKEY$, INP, INPUT, INSTR, INT

JOY

KEY

LEFT$, LEN, LET, LINE, LIST, LOAD, LOCATE, LOG, LOG10, LOWER$

MASK, MAX, MEMORY, MERGE, MID$, MIN, MOD, MODE, MOVE, MOVER

NEXT, NEW, NOT

ON, ON BREAK, ON ERROR GOTO O, ON SQ, OPENIN, OPENOUT, OR, ORIGIN, OUT

PAPER, PEEK, PEN, PI, PLOT, PLOTR, POKE, POS, PRINT

RAD, RANDOMIZE, READ, RELEASE, REM, REMAIN, RENUM, RESTORE, RESUME, RETURN, RIGHT$, RND, ROUND, RUN

SAVE, SGN, SIN, SOUND, SPACE$, SPC, SPEED, SQ, SQR, STEP, STOP, STR$, STRING$, SWAP, SYMBOL

TAB, TAG, TAGOFF, TAN, TEST, TESTR, THEN, TIME, TO, TROFF, TRON

UNT, UPPER$, USING

VAL, VPOS

WAIT, WEND, WHILE, WIDTH, WINDOW, WRITE

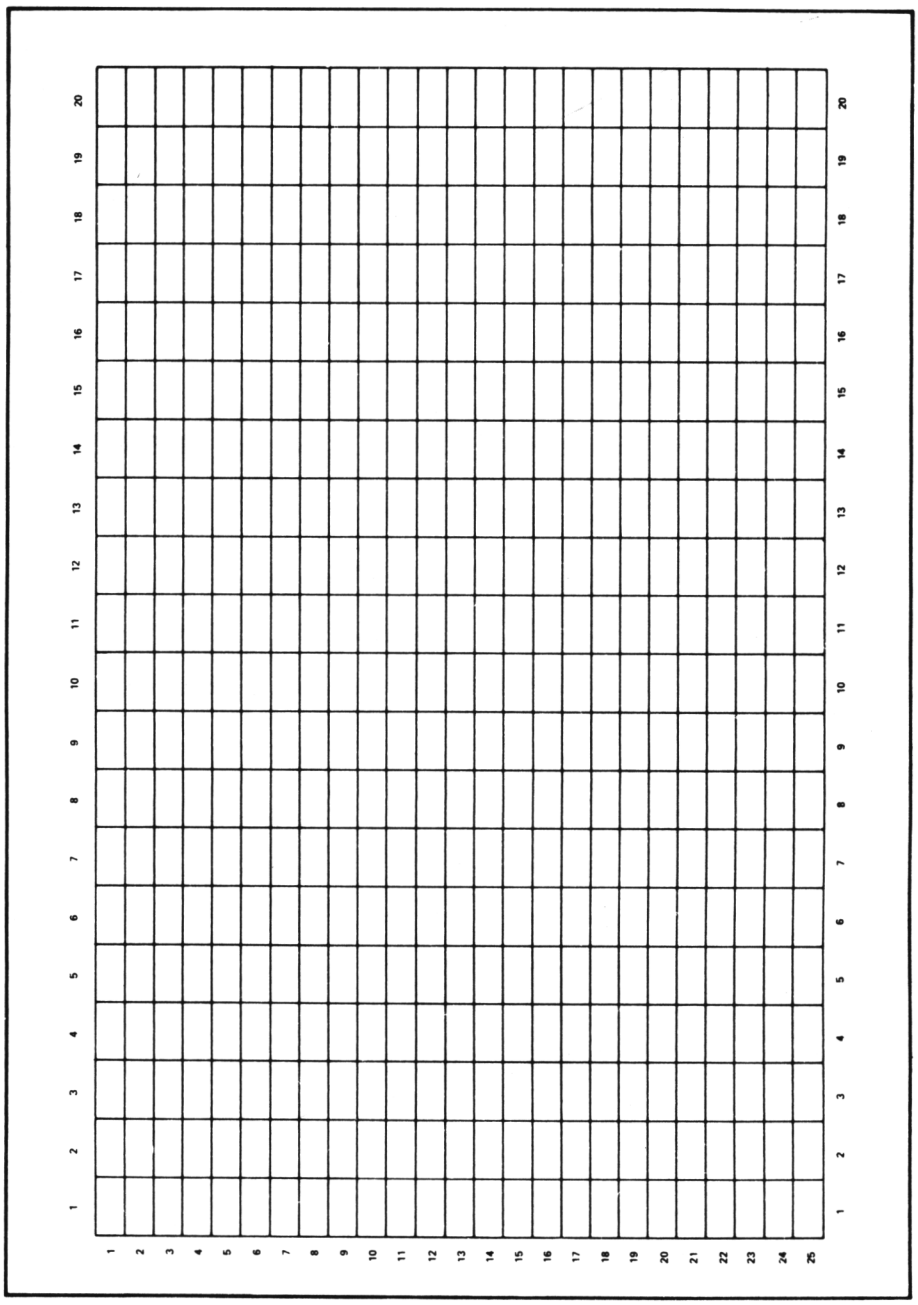
XOR, XPOS

YPOS

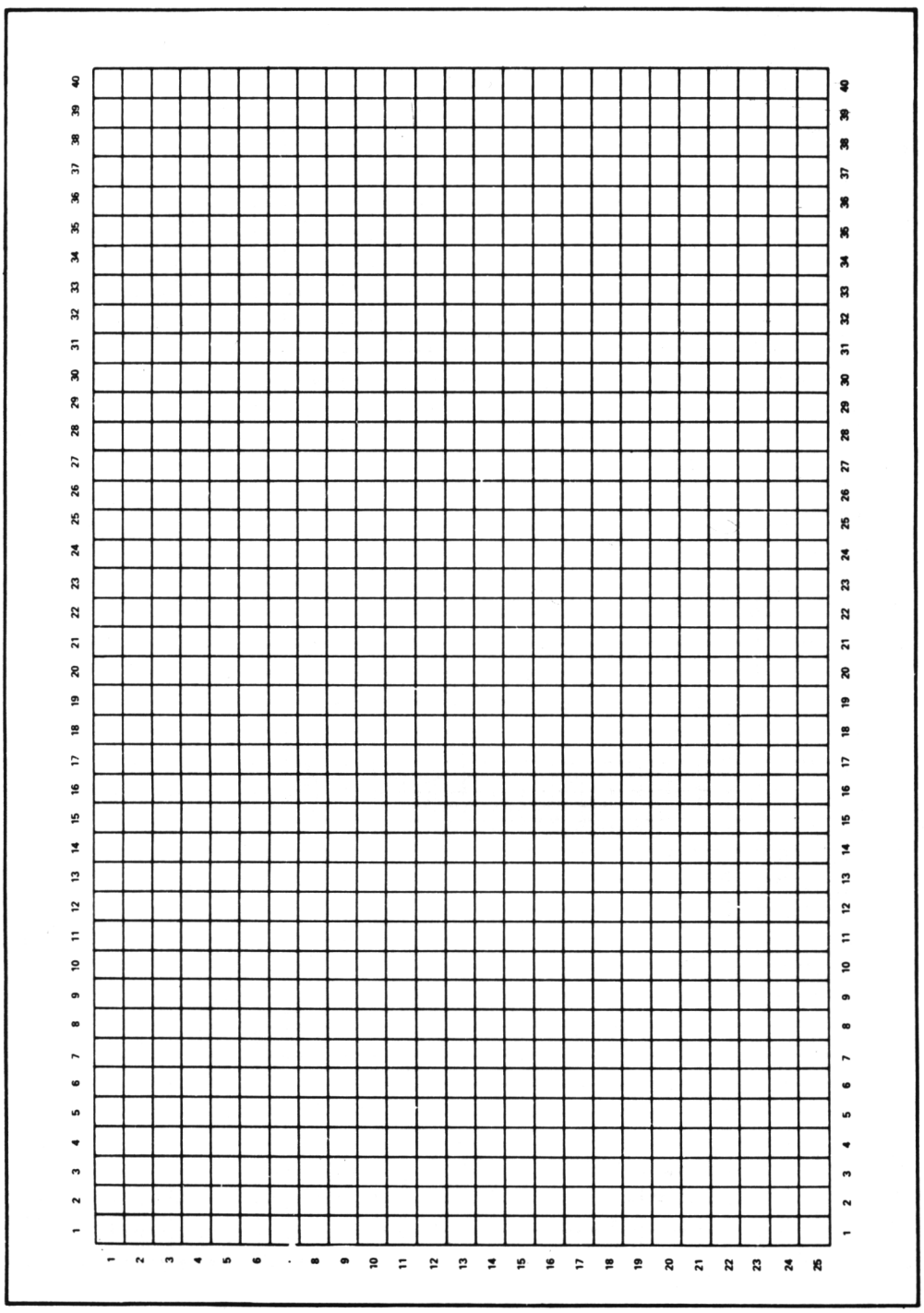
ZONE

Bölüm 8: Planlar

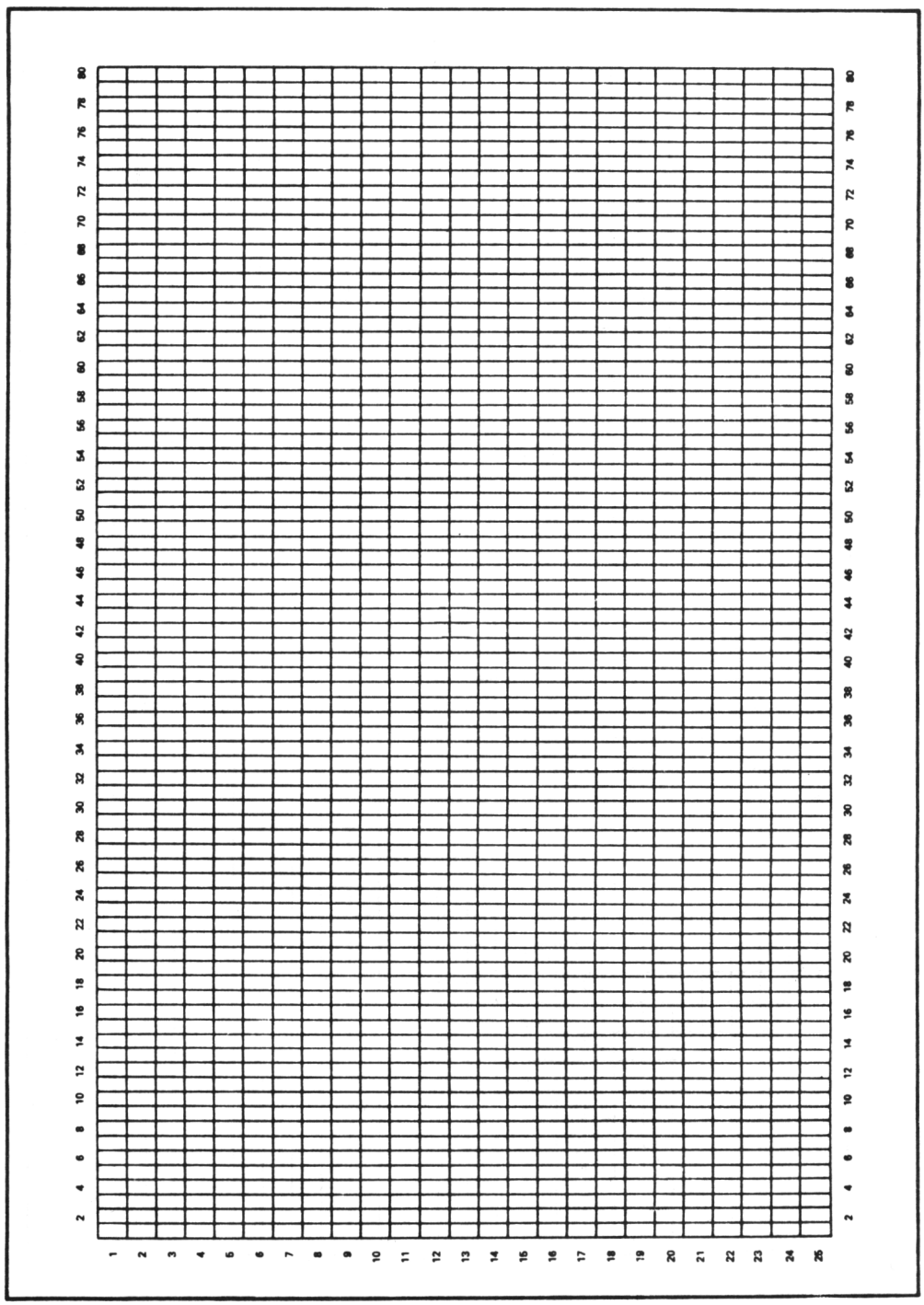
Metin ve Pencere Planı -MODE 0 (20 Kolon)



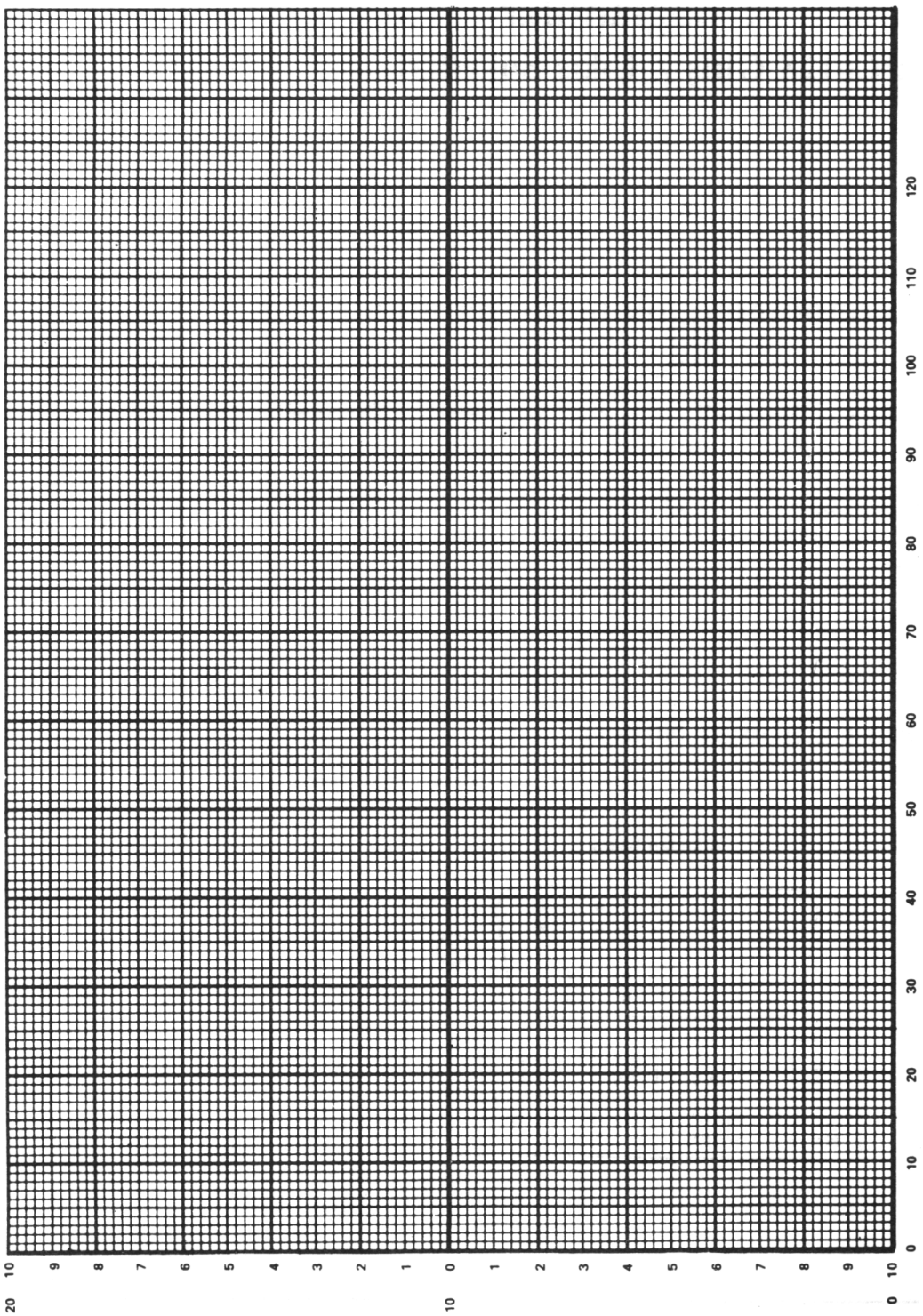
Metin ve Pencere Planı - MODE 1 (40 Kolon)



Metin ve Pencere Planı - MODE 2 (80 Kolon)

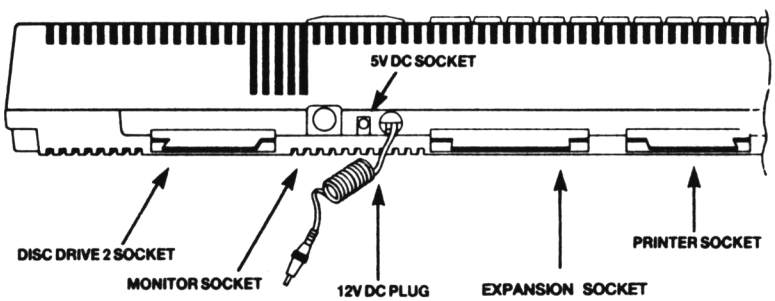


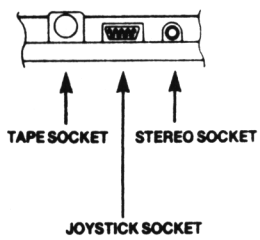
Ses Envelopu / Müzik Planı



Bölüm 9: Bağlantılar

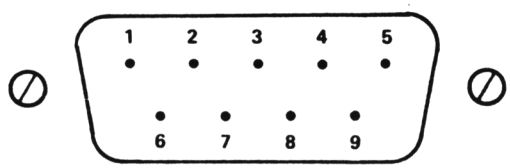
CPC6128 Giriş / Çıkış Soketleri





Oyun Çubuğu Soketi

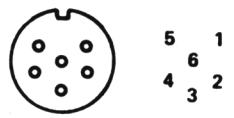
Arkadan bakış



PIN 1	UP	PIN 6	FIRE 2
PIN 2	DOWN	PIN 7	FIRE 1
PIN 3	LEFT	PIN 8	COMMON
PIN 4	RIGHT	PIN 9	COM 2
PIN 5	SPARE		

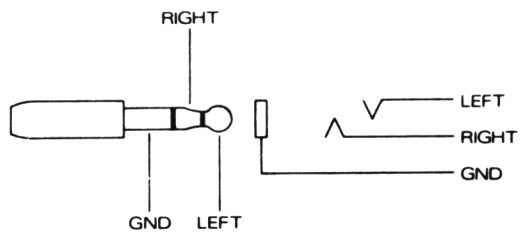
Monitör Soketi

Arkadan bakış



PIN 1	RED	PIN 4	SYNC
PIN 2	GREEN	PIN 5	GND
PIN 3	BLUE	PIN 6	LUM

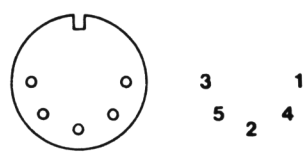
Stereo Soketi



PIN 1 LEFT CHANNEL
PIN 2 RIGHT CHANNEL
PIN 3 GND

Teyp Soketi

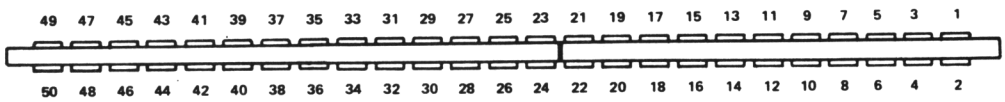
Arkadan bakış



PIN 1 REMOTE SWITCH	PIN 4 DATA IN
PIN 2 GND	PIN 5 DATA OUT
PIN 3 REMOTE SWITCH	

Genişleme Soketi

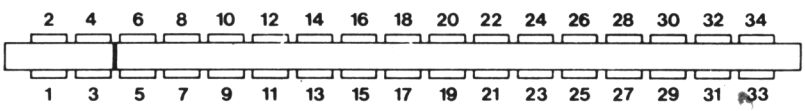
Arkadan bakış



PIN	1	SOUND	PIN	18	A0	PIN	35	$\overline{INT}$
PIN	2	GND	PIN	19	D7	PIN	36	$\overline{NMI}$
PIN	3	A15	PIN	20	D6	PIN	37	$\overline{BUSR2}$
PIN	4	A14	PIN	21	D5	PIN	38	$\overline{BUSAK}$
PIN	5	A13	PIN	22	D4	PIN	39	$\overline{READY}$
PIN	6	A12	PIN	23	D3	PIN	40	$\overline{BUS\ RESET}$
PIN	7	A11	PIN	24	D2	PIN	41	$\overline{RESET}$
PIN	8	A10	PIN	25	D1	PIN	42	$\overline{ROMEN}$
PIN	9	A9	PIN	26	D0	PIN	43	$\overline{ROMDIS}$
PIN	10	A8	PIN	27	+ 5v	PIN	44	$\overline{RAMRD}$
PIN	11	A7	PIN	28	$\overline{MREQ}$	PIN	45	RAMDIS
PIN	12	A6	PIN	29	$\overline{M1}$	PIN	46	CURSOR
PIN	13	A5	PIN	30	$\overline{RFSH}$	PIN	47	L. PEN
PIN	14	A4	PIN	31	$\overline{IORQ}$	PIN	48	$\overline{EXP}$
PIN	15	A3	PIN	32	$\overline{RD}$	PIN	49	GND
PIN	16	A2	PIN	33	$\overline{WR}$	PIN	50	ϕ
PIN	17	A1	PIN	34	HALT			

2. Disk Sürücüsü Soketi

Arkadan bakış



PIN 1	$\overline{READY}$	PIN 18	$\overline{GND}$
PIN 2	$\overline{GND}$	PIN 19	MOTOR ON
PIN 3	SIDE 1 SELECT	PIN 20	GND
PIN 4	$\overline{GND}$	PIN 21	N/C
PIN 5	READ DATA	PIN 22	$\overline{GND}$
PIN 6	$\overline{GND}$	PIN 23	DRIVE SELECT 1
PIN 7	WRITE PROTECT	PIN 24	GND
PIN 8	$\overline{GND}$	PIN 25	N/C
PIN 9	TRACK 0	PIN 26	$\overline{GND}$
PIN 10	$\overline{GND}$	PIN 27	INDEX
PIN 11	WRITE GATE	PIN 28	GND
PIN 12	$\overline{GND}$	PIN 29	N/C
PIN 13	WRITE DATA	PIN 30	GND
PIN 14	$\overline{GND}$	PIN 31	N/C
PIN 15	STEP	PIN 32	GND
PIN 16	$\overline{GND}$	PIN 33	N/C
PIN 17	DIRECTION SELECT	PIN 34	GND

Yazıcı Soketi



PIN	1	$\overline{\text{STROBE}}$	PIN	19	GND
PIN	2	D0	PIN	20	GND
PIN	3	D1	PIN	21	GND
PIN	4	D2	PIN	22	GND
PIN	5	D3	PIN	23	GND
PIN	6	D4	PIN	24	GND
PIN	7	D5	PIN	25	GND
PIN	8	D6	PIN	26	GND
PIN	9	GND	PIN	27	GND
PIN	11	BUSY	PIN	28	GND
PIN	14	GND	PIN	33	GND
PIN	16	GND	All other pins		NC

Bölüm 10: Yazıcılar

Yazıcı Bağlantısı

6128 standart centronics ara birimli yazıcıların bağlanmasına ve kullanılmasına müsaittir.

Yazıcı kablosu, bilgisayarın arkasındaki PRINTER soketi ile paralel yazıcının konnektörü arasında birebir bağlantı yapacak şekilde teşkil edilmiştir. Bilgisayarın baskılı devresinde, yazıcı konnektöründen 2 tane eksik terminal vardır. Buna sebep standart baskılı devre edge konnektörü kullanmaya imkan vermesidir.

Gerçek pin ara birimi detayları bu dersin 9. bölümünde anlatılmıştır.

Kablonun yapımından dolayı, bilgisayardaki 1 nolu pin yazıcıdaki 1 nolu pin'e, bilgisayardaki 19 nolu pin yazıcıdaki 19 nolu pin'e bağlanır. Fakat yazıcıdaki 18 ve 36 nolu pinler bilgisayara bağlanmamaktadır.

Bilgisayarın PRINTER soketinin yukarı sırasında 17 çıkıntı vardır ve alt sıra çıkıntıları 19'dan itibaren (18 yerine) numaralandırılmıştır. Bu, kullanılan her telin yazıcı soketinin pinlerindeki gibi, mutlaka bilgisayarın edge konnektöründe aynı numaralara bağlanması demektir.

Bilgisayar BUSY (pin 11) sinyalini yazıcı ile senkronize olmak için kullanır ve eğer yazıcı OFF-LINE ise bekler.

Kullanıcı tesis etme komutları gerekmez ve çıkış, 8 kalanının belirtilmesiyle yazıcıya gönderilir.

Yazıcı Konfigürasyonu

AMSTRAD DMP1 tarafından götürülen ve ekranda gözüken özel karakterler, ekran ve yazıcı için karakter kodları değişik olsa dahi yazılırlar. Bu sembollerin çoğunluğu, yazıcı bunun yabancı lisan koduna anahtarlandığında elde edilebilir. Örneğin:

```
PRINT CHR$ (&AØ)
      ^
PRINT #8, CHR$ (&AØ)
          ^
```

yazıcıda yazılır.

Uzatma işaretinin ( ) karakter kodu,DMP1'de &5E olmasına rağmen çalışır.Diğer bir deyişle yazıcı programı AØ'yı yazıcı tercüme tablosunda tutulan kodlardan birisigibi tanımlar ve&5E'ye çevirerek aynıkarakterin ekranda görünmesini ve yazıcıda yazılmasını sağlar.&5E kodu DMP1'de hangi lisan moduna ayarlanırsa ayarlansın uzatma işareti üretilir. Diğer karakterler tabloda gösterilmiştir.

CHR$	Character On Screen	Printer Translation	U.K.	U.S.A.	France	Germany	Spain
&A0	^	&5E	^	^	^	^	^
&A2	..	&7B	†	†	†	†	..
&A3	£	&23	£	#	#	#	Pt
&A6	§	&40	†	†	†	§	†
&AE	¿	&5D	†	†	†	†	¿
&AF	¡	&5B	†	†	†	†	¡

+ yazılan karakter için DMP1 talimat kitabının 18. sayfasına bakınız.

Yukarısı daha önceden belirlenmiş çevirinin özet formudur, istenirse değiştirilebilir. Daha fazla detaylar için SOFT 968 Firmware'e bakınız.

Bölüm 11: Oyun Çubuğu

Bilgisayarda yapılmış yazılım bir veya iki oyun çubuğu kullanılmasına izin verir. Bunlar tuştakımı gibi ele alınır ve INKEY, INKEY$ tarafından klavye tuşlarıymış gibi dikkate alınır.

Oyun çubuğunun ana ateş düğmesi "ateş 2" olarak tarif edilir.

JOY (Ø) ve JOY (1) fonksiyonları sırasıyla 1'nci ve 2'nci oyun çubuklarını doğrudan yoklarlar. Fonksiyon son klavye taramasında oyun çubuğu anahtarlarının durumunu gösteren bitler seviyesinde bir sonuçtur.

Aşağıdaki tablodu her iki oyun çubuğundan dönen değerler belirtilmiştir. JOY değerlerini, tuş parametre gibi alan talimatlarda kullanılan değerler takip eder.

KONUM	JOY KOMUTU		TUŞ DEĞERLERİ		
	BİT	GERİ DÖNEN DEĞER	İLK OYUN ÇUBUĞU	İKİNCİ OYUN ÇUBUĞU	EŞDEĞER TUŞ
Up	0	1	72	48	"6"
Down	1	2	73	49	"5"
Left	2	4	74	50	"R"
Right	3	8	75	51	"T"
Fire 1	4	16	76	52	"2"
Fire 2	5	32	77	53	"F"

İkinci oyun çubuğu için olan değerler geri döndüğü zaman bilgisayar bu değerlerin oyun çubuğunun mu yoksa tuş takımından mı üretildiğini ayıramaz. Bu tuş takımının ikinci oyun çubuğu yerine kullanılabileceği anlamına gelir.

Bölüm 12: Disk Organizasyonu

AMSDOS üç değişik disk formatlamaya dayanır, bunlar: SİSTEM format DATA ONLY format ve IBM format. AMSDOS altında, disk açık olmayan dosyalar girildiğinde otomatik olarak her seferinde araştırılır. Bu otomatik araştırmaya izin vermemek için her formatın tek sektör numarası vardır.

3 inch'lik diskin iki yüzü vardır, fakat kullanıcının hangi yüzü soktuğuna bağlı olarak bir yüzüne girilebilir. Her iki yüzünde de değişik formatlar olabilir.

Bütün Formatlar İçin Ortak

Tek yüzü
512 byte fiziksel sektör boyu
40 kanal (0'dan 39'a kadar numaralandırılmış.)
1024 byte CP/M blok boyutu
64 rehber girişi

SİSTEM Format

41'den 49' her kanal için 9 sektör 2 ayrılmış kanal

Sistem formatı ana format dayanaklıdır ki CP/M sadece sistem diskinden yüklenebilir. CP/M 2.2'de sistem format diskini warm boot için sokulmasını gerektirir. Ayrılmış kanallar aşağıdaki gibi kullanılırlar.

	CP/M 2.2.	CP/M Plus
kanal 0 sektör & 41	CP/M için boot sektörü	:CP/M için boot sektörü
kanal 0 sektör & 42	konfigürasyon sektörü	: kullanılmaz
kanal 0 sektör & 42-& 47	kullanılmaz	: kullanılmaz
kanal 0 sektör & 48-& 49	CCP ve BDOS	: kullanılmaz
kanal 1 sektör & 41-& 49	CCP ve BDOS	: kullanılmaz

VENDOR formatı, sistem formatının özel bir şeklidir. Burada herhangi bir sistem yazılımı ve sistemin kendine ayırdığı iki track bulunmaz. Bu format genelde yazılım firmaları program dağıtımında kullanırlar.

SADECE DATA Format

&C1'den &C2'a kadar numaralanmış her kanalda 9 sektör 0 ayrılmış kanal

Bu format şeklinin CP/M 2.2 ile kullanılması tavsiye edilmez çünkü bundan warm boot yapılması mümkün değildir. Bununla beraber eğer CP/M Plus veya AMSDOS kullanıldığında DATA format biraz daha fazla disk yeri meydana getirir.

IBM Format

&C1'den &C8'e kadar numaralanmış her kanalda 8 sektör 1 ayrılmış kanal

Bu format mantıksal olarak IBM PC'de CP/M tarafından kullanılan tek yüzlü formatın aynısıdır. 6128, IBM format disklerini okuyup yazabilir fakat bunları kopya edemez veya meydana getiremez.

Bölüm 13: Yerleşmiş Sistem Uzantıları (RSX)

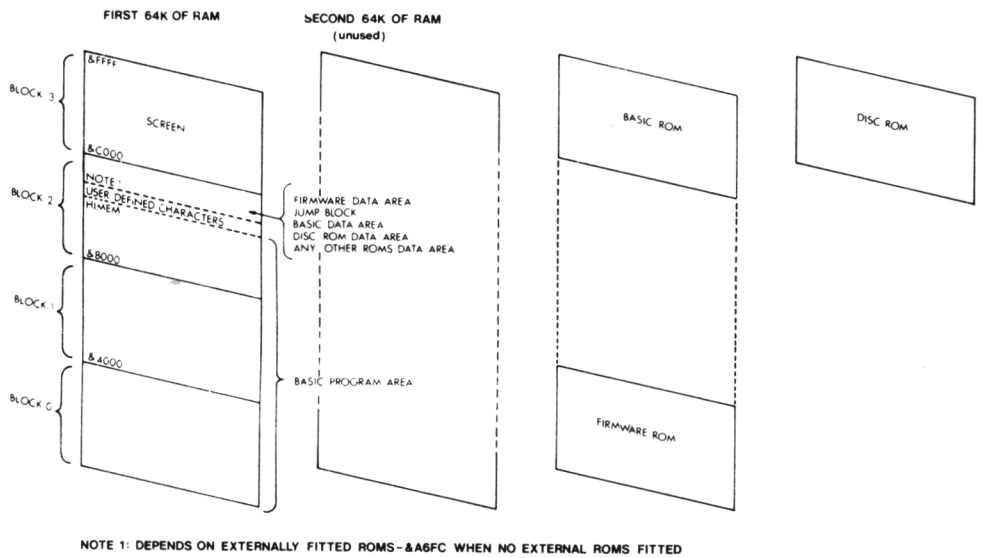
Harici komutlar DERS 5'de tanıtıldı. (AMSDOS hakkında) Temel olarak, bir harici komut BASIC'in repertuarını | ön işareti ile belirtilmiş yeni komutlar ekleyerek genişletme yoludur. AMSDOS yeni komutları için makina talimatları ROM'a dahil edilmiştir ve 6128 BASIC'e başladığında komutları eklemek için gerekli idare otomatik olarak ele alınır.

BASIC başladıktan sonra makina talimatlarını RAM'a yükleyerek birçok harici komutu ilave etmek mümkündür. Bu tür komutlar RSX diye adlandırılırlar ve ROM tabanlı genişletilmiş komutlar gibi çalışırlar. RSX komutlar 6128'e Basic başladıktan sonra her seferinde diskten yüklenmelidirler. Normal olarak RSX komutlar bazı akıllı ek ünitelerin kontrolü için kullanılır. (Light pen, speech synthesiser)

Ders 8'de RSX komutlar kullanarak 6128'in ikinci 64K hafızasına nasıl erişildiği anlatılmaktadır.

Bölüm 14: Hafıza

CPC 6128, 128k'lık RAM VE 48K'lık ROM ihtiva eder. Bu aşağıda görüldüğü gibi BASIC 1.1 için elde edilebilir. İlk 64K'lık RAM nominal olarak her biri 16K olan dört bloğa ayrılmıştır. Blok 0'dan 3'e kadar numaralandırılmıştır. Ekran blok 3'ü kullanır ve blok 2'nin üst kısmı belirtildiği gibi sistem değişkenleri ile doldurulur.



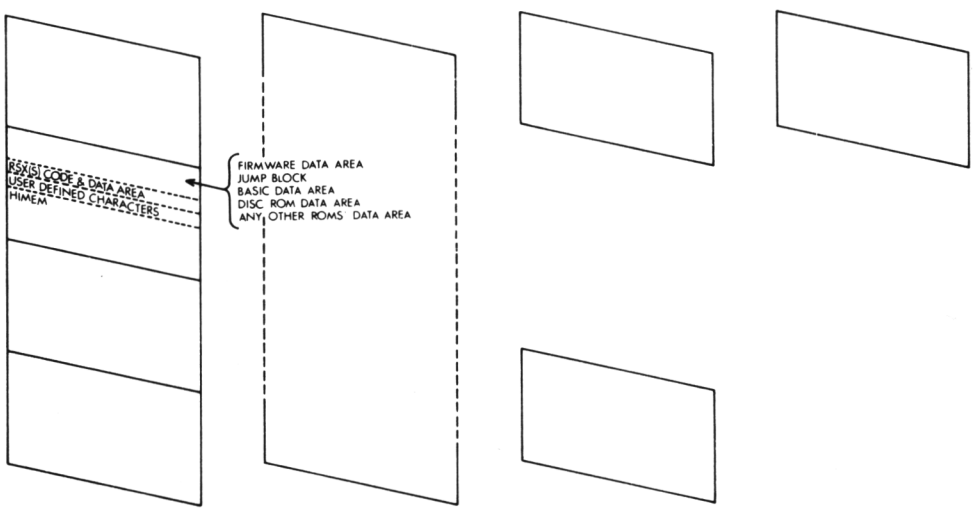
Basıc 1.1 için bellek haritası

Tanımlanabilir kullanıcı karakterleri başlangıçta hemen HIMEM'in üzerinde yer alırlar. HIMEM, MEMORY komutlarıyla değiştirilebilir fakat AMSDOS dosyalar açıkken tampon yaratmak için otomatik olarak 4K düşürülebilir. Tanımlanabilir kullanıcı karakterleri ancak HIMEM ayarlandığı zaman değişmemişse değiştirilebilirler. (Aksi takdirde SYMBOL AFTER 256 komutu tarafından "tanımlanabilir kullanıcı karakter yok" diye ayarlanır.) BASIC başladığında tanımlanabilir kullanıcı karakterleri sanki SYMBOL AFTER 240 komutu meydana gelmiş gibi ayarlanırlar.

Bu sebepten dolayı HIMEM'i geçici olarak değiştirilmeden önce tanımlanabilir kullanıcı karakter alanını çökertir, sonra yeni bir pozisyonda tekrar kurar. Bu SYMBOL AFTER'ı değiştirmek için sonradan gelen programlara müsade eder.

Aşağıdaki örnek RSX yüklenmesiyle birlikteki HIMEM azaltılması sırasındaki operasyonun planı gösterir.

```
100 SYMBOL AFTER 256 'çöken tanımlanan kullanıcı karakterleri
110 rsxaddress = HIMEM-rsxlenght
120 MEMORY rsxaddress-1
130 LOAD "rsxcode", rsxaddress
140 CALL rsxaddress ' log on rsx
150 SYMBOL AFTER 140 ' tanımlanan karakterleri eski haline koy
```



Memory Map with RSX loaded in recommended position

İLAVE I/O

Ancak I/O kapı adresleri bilgisayar tarafından saklanır 7FFF'den aşağı adresler kullanılmamalıdır.

AO-A7 adres kısmının harici I/O cihazlarının tipini yansıtması hedeflenir ve A8, A9 adres satırları, I/O cihazlarının içindeki kayıtlarını seçmek için deşifre edilir. Geri kalan adres satırlarının A11 ve A15 yüksekken sadece A10 düşük olarak deşifre edilmelidir. Bunun için her cihazın çF8??, cF9??, çFA??, çFB?? gibi adreslenmiş kayıtları olması gerekir, burada ?? kominikasyon bağlantıları için DC'den D1'ye kadar ve diğer çevre elemanları için EO'dan FE'ye kadar olmalıdır. Unutmayın ki adres yolunun yukarı yarısının üzerindeki (A15-A8) B kaydına yerleştirilen Z80 talimatları kullanılmalıdır.

Ek ROM lar

Devredeki ROM'un herhangi bir kısmının yerine seçilerek ilave ROM'ların tedariği yapılır. Adres seçici ve banka seçme mantığı genişleme yoluna bir modül içinde bağlanmalıdır. Fakat bütün gerekli sinyaller genişleme yolundan (expension bus) getirilmelidir.

Bölüm 15: CP/M Plus Terminal Benzeticisi

Bu dersin 1. bölümünde kontrol karakterleri tablosu incelenmişti (bağlantı faaliyetleriyle birlikte). Bu faaliyetler, metin BASIC veya CP/M 2.2 tarafından ekrana gönderildiğinde gerçekleşir ve VDU Firmware yazısında elde edilebilen faaliyetleri yansıtmak ve kullanma basitliği için seçilirler. Bu faaliyetler AMSTRAD bilgisayarlarında tekdir ve yazılım için adapte edilmiştir.

CP/M yazılımı ticari ve iş çerçevesinde bir makinadan diğer makinaya uygulanabilir ve kolaylıkla taşınabilir bir yazılım için standart VDU metin faaliyetleri ummak normaldi. 6128 üzerindeki CP/M Plus tamamlayıcısı Zenith Z19/Z29 VDU'ya çok benzer faaliyetleri tedarik eden terminal benzeticisi içerir. CP/M Plus yazılımı; tesisat prosedürü normal olarak bu tip terminal için standart bir seçme ihtiva eder.

CP/M Plus terminal benzeticisi tarafından sunulan faaliyetler değişik kontrol kodları gerekmesine rağmen VDU Firmware yazısının daha önce sunduklarını içerir.

20'den FF'e kadar alandaki karakterler şimdiki kursör pozisyonunda görüntülenir. Eğer kursör sağ tarafdaki kolonda değilse bir kolon sağa hareket ettirilir. Eğre kursör sağ tarafdaki kolondaysa ve yazılma durumundaysa bir sonraki satırın sol kolonuna hareket ettirilir, gerekliyse ekrandaki yazılar yukarıya yuvarlanır.

00'dan F1 arasındaki kontrol kodları olarak çevrilen karakterler şunlardır.

&07 BEL Bip sesi

&08 BS Geri dönüş. Sola bir kolon hareket eder. Eğer kursör en soldaki kolonun üzerindeyse bir üstdeki satırın en sağ kolununa haraket ettirilir.

&OA LF Satır atlama. Kursör bir satır aşağı haraket eder, ekran yuvarlanması gerekiyorsa yapılır.

& OD CR Satır başı. Kursör son satırın en sol kolonuna gider.

& 1B ESC Kaçış sırasını bildirir.

Diğer bütün kontrol kodları ihmal edilir.
Şu kaçış sıraları kabul edilir. Bir kaçışı takip eden diğer herhangi bir karakter görüntülenir ve kursör ilerler. Bu özellik kontrol kodları 00'dan 1F'e kadar olan karakterleri görüntülemekte kullanılabilir (unutmayınki bir çok uygulama lisanları 09 (TAB) kontrol kodunu birkaç başlık olarak genişleterek (ESC) (TAB) genellikle 09 karakterleri için görüntülemez.

[ESC] 0 Vaziyet satırı zayıflar. Disk sistem mesajı CTR çıkışı ile gözükür. CTR ekranın alt satırını kullanır.

[ESC] 1 Vaziyet satırını kuvvetlendirir. Disk sistem mesajları ekranın alt satırında görünür.

(ESC) 2 (n) Karakter ayarlamasını değiştirir (bu dersin 16. bölümüne bakınız). (n) lisan parametresidir ve 07 ile maskelenmiştir. Belirli karakter matrisleri (20 ile F7 arasındaki) 80'den FF'e kadar olan diğer karakterle değiş tokuş yapılır. Bu komutun faaliyeti enternasyonal karakterleri seçebilen yazılıma sahip yazıcıların kontroluna benzer.

(n) = 0 USA
(n) = 1 France
(n) = 2 Germany
(n) = 3 UK
(n) = 4 Denmark
(n) = 5 Sweden
(n) = 6 Italy
(n) = 7 Spain

[ESC] 3 (m) Ekran modunu değiştirir. (m) = ekran modu + 20. Eğer mode 0 ile 2 arasını vermek için 3 ile maskelenir. Mode 3 ihmal edilir. Ekran temizlenir, fakat kursör pozisyonu etkisizdir.

[ESC] A Kursör yukarı. Eğer en üstteyse birşey yapılmaz

[ESC] B Kursör aşağı. Eğer en alttaysa birşey yapılmaz

[ESC] C Kursör ileri. Eğer en sağ kolondaysa birşey yapılmaz.

[ESC] D Kursör geriye. Eğer en sol kolondaysa birşey yapılmaz.

[ESC] E Sayfa sil. Kursör pozisyonu etkisizdir. Bu komut bütün ekranı temizler (24 × 80 moduna dahi ayarlı olsa). Diğer kaçış modları 24 × 80 moduna ayarlanmışsa sadece 24 × 80 alanda etkili olur.

[ESC] H Kursör evi. Kursörü sol kolonun en üst satırına haraket ettirir.

[ESC] I Ters indeks. Kursörü bir satır üste çıkartır. Eğer gerekirse ekranı aşağıya kaydırır.

[ESC] J Sayfanın sonuna kadar temizler. Kursör pozisyonundaki karakteride siler. Kursör pozisyonu etkilenmez.

[ESC] K Satırın sonuna kadar siler. Kursör pozisyonundaki karakteride siler. Kursör pozisyonu etkilenmez.

[ESC] L Satır sakıştırır. Tüm aşağıdaki satırlar kursörün bulunduğu satar dahil aşağıya kaydırılır. Kursör sahası temizlenir. Kursör pozisyonu etkilenmez.

[ESC] M Satır sil. Kursörün bulunduğu sıra dahil tüm aşağıdaki satırlar yukarı kaydırılır. Kursör pozisyonu etkilenmez.

[ESC] N Karakter sil. Kursörün sağındaki tüm karakterler sola, bir karakter pozisyonuna değiştirilirler. Sıranın sonundaki karakter temizlenir. Kursör pozisyonu etkilenmez.

[ESC] Y (r)(c) Kursörü verilen pozisyona hareket ettirir. Eğer pozisyon ekran kenarının dışındaysa, kursör ekran kenarına, kadar hareket eder. (r) = sıra pozisyonu + 20, (c) = kolon pozisyonu + 20. En üst sol köşe sıra 0, kolon 0'dır.

[ESC] b(cp) Ön plan rengini ayarlar. Ekrandaki tüm karakterleri etkiler. (cp) renk parametresidir ve 3F ile maskelenir, sonra iki bitlik üç sayı gibi ele alınır ki her biri üç ilkel rengin yoğunluğunu belirter şöyleki: bit 0,1 mavi, bit 1,2 kırmızı ve bit 3,4 yeşil. 6128 dört tanımlanabilir seviye üzerine üç yoğunluk seviyesini planlamıştır:

CPC 6128	Sıfır yoğunluk	Yarım yoğunluk	Tam yoğunluk
Renk bitleri	00 binary	01 veya 10 binary	11 binary

[ESC] c(cp) Arka zeminin rengini ayarlar. Ekran üzerindeki bütün arka zemin ve çerçeve etkilenir. Renk yukardaki gibi belirtilir.

[ESC] d Sayfanın başlangıcını siler. Kursörün durduğu yerdeki karakter dahil. Kursörün pozisyonu etkilenmez.

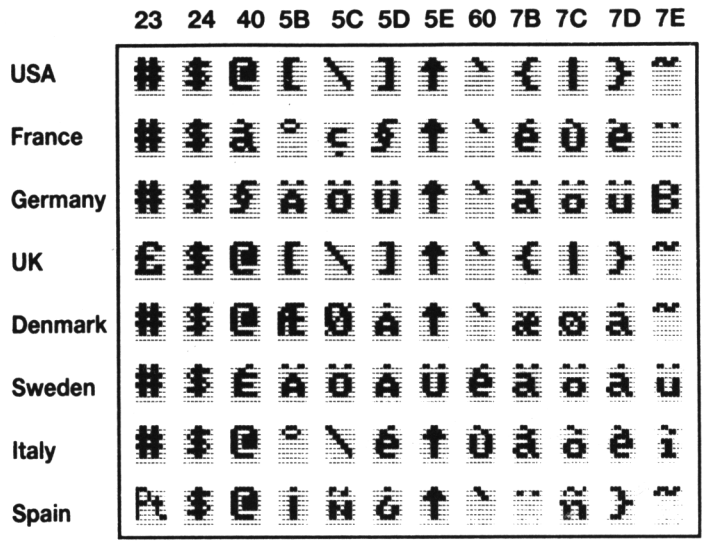
[ESC] e Kursörü yak. İstenmeyen bir parlamayı önlemek için normal yazı sırasında en son karaktere basıldıktan 1/10 sn. sonraya kadar kursör gözükmez.

[ESC] f	Kursörü kapat.
[ESC] j	Kursör pozisyonunu sakla.
[ESC] k	(ESC)j'nin sakladığı gibi kursör pozisyonunu eski haline koyar.
[ESC] l	Satır sil. Kursör pozisyonu değişmez.
[ESC] o	Satırın başlangıcını sil. Kursörün durduğu pozisyondaki karakteride siler. Kursör pozisyonu değişmez.
[ESC] p	Ters video modunu gir. Yazılabilen karakterler ön plan ve arka zemin renkleri ters çevrilirken yazılır.
[ESC] q	Ters video modundan çık.
[ESC] r	Altını çizme moduna girer.
[ESC] u	Altını çizme modundan çıkar.
[ESC] v	Satır sonunda katlan.
[ESC] w	Satırın sonunda çıkar.
[ESC] x	24x80 moduna gir. Bazı uygulama programları standart 24x80 ekranını gerektirir. Bu komut, ekran hangi makinaya ait olursa olsun ve hal satırı girilip girilmemesi farketmeden, ekranın tüm boyutları dikkate alınmaksızın, 24x80 ekranı yapar. Ekran temizlenir.
[ESC] y	24 × 80 modundan çıkar. Ekran temizlenir.

Bölüm 16: CP/M Plus Karakter Tertibi

Bu dersin 10. bölümünde yazıcı çeviri tablosu anlatılmıştı. Bu tablonun amacı belirli karakterleri, BASIC ve CP/M 2.2 karakter tertibinden lisan seçimi yapılmış yazıcı tarafından yazılabilecek forma çevirmektir. Bu işlem, BASIC karakter setinde birkaç yabancı yazım karakterleri bulunduğundan kısıtlıdır.

Her ne kadar bu yazıcı çevirici şeması CP/M Plus altında çalışıyorsa da karakter tertibi, ekran karakterleri ile yazılan karakterler arasındaki uygunluğu sağlamaya müsade etmek için fazlalaştırılmıştır. Aşağıdaki tablo bu bağlantıyı gösterir:



CP/M Plus International Character Set

Makinanın yabancı çevre ile çalışması iki faaliyeti gerektirir.

1- Yazıcının kendisi gerekli lisana ayarlanmalıdır. (Genellikle küçük bir (DIP) anahtar ile olmakla beraber bazı yazıcılar kontrol kodları gönderilmesine izin verirler.)

2- İstenilen ekran karakter tertipleri;

LANGUAGE (n) geçiş komutu

veya

[ESC] 2 (n) terminal benzeticisine gönderilerek yapılır.

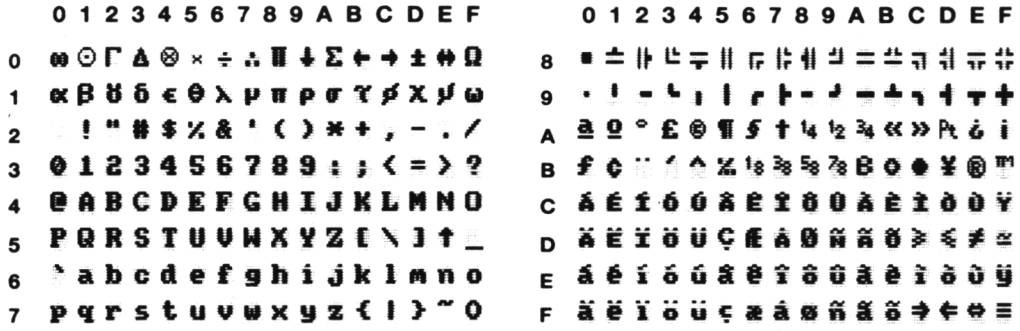
Pratikte, başlangıç konumu, LANGUAGE ve SETLST faaliyetlerini kullanarak PROFILE.SUB operasyonunun bir kısmı gibi gerçekleştirilebilir.

7 bit yazılım

Yabancı lisan faaliyeti çok kuvvetli olmakla beraber, önemle ifade edilmelidir ki yerel yabancı karakterler, değiştirilen normal (veya USA) karakterler, bundan sonra görüntülenemezler. Yaklaşık bütün yazılımlar (birçok CP/M Plus faydaları, kelime işlemcileri ve lisanlar) sadece 7 bit karakter teşkili ile çalışırlar.

8 bit karakter tertibi ile çalışmak...

BASIC karakter tertibi 256 değişik sembol içerir ve değerler 128'den 255'e kadar (80'den FF'e kadar) olan ve oyunlarda, ev hesaplarında kullanılan değişik grafik sembollerini kapsar. CP/M Plus da 256 karakteri içerir fakat ikinci 128'i BASIC'de olanlardan farklıdır, CP/M Plus çerçevesinin iş ve enternasyonal tadını yansıtır.



Characters 0 to 127 (&00 to &7F) Characters 128 to 255 (&80 to &FF)

Standart CP/M Plus Karakter Seti

8 bit karakterle çalışabilir yazılım, bu karakterleri yukarıda görülen ekran resmindeki gibi sadece kutuları doldurmakta değil lisanda belirli bir değişiklik yapmadan bütün yabancı dil karakterlerini bir kerede girebilir. Bu yol ile bütün çubuk ve parantez tipi karakterler elde edilebilir.

Şunu unutmayın ki, hali hazırda çok az 8-bit yazılım mevcuttur ve bu 0'dan 31'e kadar ve 128'den 255'e kadar olan karakterler yukardaki ekrandaki şekilde gözükürler. Diğer taraftan yazıcıda nasıl çıkacağı, yazıcının özelliğine bağlıdır.

KONU 8
Bellek Bankası Yönetimi

İlgili Konular:

* Ekran Görüntülerinin saklanması
* Sahte-dosya çalışması

BASIC 1.1 için bellek planı (Ders 7 bölüm 14'de gösterilmiştir) 128K RAM'ın 64'K'sının kullanılmadığını gösterir. BASIC ve firmware, disk ROM ile birlikte olan ROM'da bulunurlar ve normal belleği 64K'dan 112K'ya (64K RAM, 48K ROM) yükseltir.

Her 16K'lık bölüm 'blok' diye adlandırılan ve herhangi bir dört blok seçimi (toplam 64K yapmak için) 'banka' diye adlandırılır. Bu sebeple blok seçim tekniği 'banka anahtarlaması' diye isimlendirilir.

Z80 mikroişlemcisi, herhangi bir zamanda sadece 64K'lık belleğe yer bulabilir. Bundan dolayı çalışma sisteminin firmware ROM'u RAM'ın 0 bloğu yerine harekete geçiren ve BASIC ROM veya disk ROM'u 3 blok yerine harekete geçiren talimatları vardır. Bu anahtarlamalar BASIC veya firmware gerektiği zaman otomatik olarak olur. RAM'ın banka anahtarlanması sadece bu kavramı ROM'dan ziyade RAM'ın kaplanmasını ilave için genişletir. Anahtarlama bir assebler programı tarafından yürütülür.

BANKMAN.BAS programı sistem disk paketinin 1. yüzünde önceden hazırlanmıştır. Eğer program BASIC başladıktan sonra işletilirse standart bellek banka yönetimini RSX kodda tesis eder. Program 'BELLEK BANKASI YÖNETİMİ' diye bilinir.

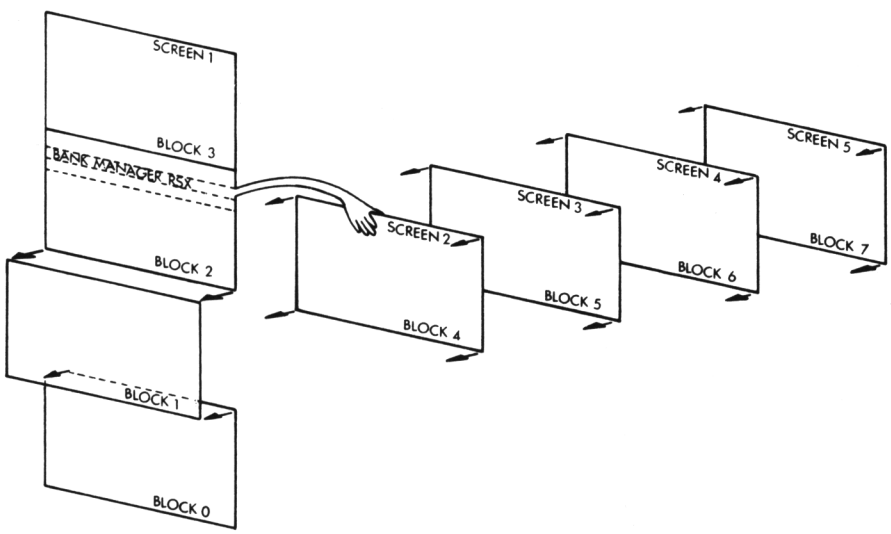
64K'lık hafızanın bir kullanımı da resim ekranı için geçici depo yeri gibidir. Örneğin: Birçok değişik ekranları depolayan veya hazırlanmış değişik ekranları olan video oyunları kapsayan tür uygulamalar.

İkinci 64K'nın diğer bir kullanılışı dizi matris alanına veya sadece RAM diske uzaması gibi ele alınan değişken çalışma alanı uzamasıdır.

Bölüm 1: Ekran Görüntülerinin Depolanması

Ekranınızı Seçin

Bellek bankası yöneticisi Blok 1'i anahtar dışı yapıp, ikinci 64K'daki dört bloktan birini Blok 1'in yerine anahtarlayabilir. Aşağıdaki çizim bu faaliyeti göstermektedir. İkinci 64K'daki her bloğun aynı adres alanını doldurduğuna dikkat eden. (4000'den 7FFF'e kadar) Blok 1'in içeriği (muhtemelen BASIC programınızın ortasıdır) korunur ve Bellek Bankası Yöneticisi bitince eski haline konulur. Aşağıda beş adetten başka üç tane daha banka seçimi mümkündür, fakat bunlar CP/M Plus'ın yerine getirilmesinde faydalıdır.



Hardware bank switching

Bellek Banka Yöneticisi, bilgilerle dolu bir bloktan diğerine hareket etmek için iki komuta gereksinir. 4. ve 7. blokların içeri ve dışarı anahtarlanmaları gerekirse otomatik olarak yapılırlar ve bellek şeması, içeri anahtarlanmış blok 1 ile bırakılır.

Komutlar:

ISCREENSWAP, [(ekran bölümü),] (ekran numarası), (ekran numarası)

iki bloğun içeriğini değiştirir.

ISCREENCOPY, [(ekran bölümü),] (gidilecek ekran numarası), (kaynak ekran numarası)

bir bloğun içeriğini diğer bir blok içine kopya eder.

Tercihe bağlı (ekran bölümü) parametresi, bloğun 1/64'ünün (16K'dan 256 bayt'ının) kopyası için yazılıma sebep olur. Bu sebeple (ekran bölümü) 0 ile 63 arasındaki değerleri alır. Bu mode'deki çalışma, eğer ekran hareketiyle birlikte diğer bir işlem koymak gerekirse kullanışlıdır. Bir ekran değiş tokuşu saniyenin 150/300'de biri (150 zaman sayımı) sürede olur.

(Ekran numarası)'nın 1 (normal ekran), 2, 3, 4 veya 5 olması gerekir. Kopya ve değiş tokuş işlemleri kaynak veya gidilecek yer ekran 1 ise daha hızlı çalışır. Ekran donanım yardımıyla yuvarlanma etkisine dikkat ediniz. Ekran görüntüleri o şekilde yapılmalıdır ki ekran 1 ile aynı donanım pozisyonunda olsun. En basit pozisyon MODE komutuyla teşkil etmektir.

Ekran Anahtarlama Komutlarının Kabiliyetini Tecrübe Etmek

Önce, Bellek Bankası Yöneticisi programı sistem disk paketinin 1. yüzünden şu şekilde çalıştırılır:

```
RUN "BANKMAN"
```

Sonra şunu yazın:

```
MODE 1
```

Ekran temizlenir. Ardından da şunu yazınız:

```
' Bu esas ekran
ISCREENCOPY, 3,1 'Esas ekranı bellek 3'e gönderdi
CLS
```

Ekran tekrar temizlenir. Şunu yazınız:

```
' Bu ortadaki ekran
ISCREENCOPY,2,1 'Orta ekranı bellek 2'ye gönder
ISCREENSWAP,2,1 'Bellek 2 ve bellek 3'ü değiştir
ISCREENCOPY,1,3 'Orta ekranı bellek 3'den eski haline koy
ISCREENCOPY,1,2 'Esas ekranı bellek 2'den eski haline koy
```

Son olarak, bu konu üzerinde, Ders 9'un son bölümünde kapsamlı bir 'Ekran tasarımıcısı' proramı vardır.

Bölüm 2: Sahte-Dosya Çalışması

RAM disk olarak ele alındığında ikinci 64K'nın tümü, sabit uzunlukta kayıtlar içeren 'RAMdosya'ya bölünür. Kayıt uzunluğu 0 ve 255 bayt arasında olabilir. (Tavsiye edilen en az 2 bayt'tır.) (RAM kayıt uzunluğu) bir kere kurulduğunda, her kayıt kendi (RAM kayıt numarası) tarafından girilir. Bir kayıt uzunluğu kullanarak RAM dosyası yazmak ve diğerini kullanarak geri okumak kabul edilir.

NOT: RAM dosyası sadece veri içermelidir. Program talimatları ihtiva etmesi gibi bir faaliyeti yoktur.

Rastgele okuma disk dosyalama planları ile ortak olduğu gibi 'son kayıt numarası' kavramı vardır. Bu önceden belirlenmiş kayıt numaralarıdır ki RAM dosyasında otomatik basamaklandığı zaman faydalıdır.

KOMUT

```
I BANKOPEN, (RAM kayıt uzunluğu)
```

Bütün kayıtları sabit uzunluğa ayarlar, son kayıt numarasına sıfır başlangıç değeri verir fakat herhangi bir şekilde belleği temizleyemez.

KOMUT:

I BANKWRITE, @(geri dönüş kodu), (dizi ifadesi), (,[RAM kayıt numarası)] (dizi ifadesi)ni RAM dosyasına yazar.

(RAM kayıt numarası) hangi kaydın yazıldığını belirtir. Eğer parametre ihmal edilirse son kayıt numarası kullanılır. Son kayıt numarası bir sonraki kayıt noktasına ayarlanır. Eğer (dizi ifadesi) kayıdı tamamen doldurmazsa eski karakterler (üzerine yazılmayan) kayıdın sonunda kalır. Eğer (dizi ifadesi) kayıttan uzunsa artan karakterler bir sonraki kayıda dökülmesini bertaraf etmek için atılırlar. (geri dönüş kodu) bir tamsayı değişkenidir, yazılan kayıt numarasını getirir (eğer operasyon başarılı ise), veya yazma operasyonu bir şekilde başarısız olmuşsa negatif bir hata kodu verir.

-1 Dosya sonu hatası. Gerekli kayıt numarasının adresi 64K'yı aşmış.

-2 Banka anahtarlama hatası. (Hiçbir zaman olmamalı)

ÖRNEKLER:

```
IBANKOPEN, 1Ø
IBANKWRITE, @r%,"123 test",Ø
IBANKWRITE, @r%,w$
```

KOMUT:
IBANKREAD, @(geri dönüş kodu), @(dizi değişkeni) [,(RAM kayıt numarası)]

RAM dosyasından (dizi değişkeni)'ne kayıt okur.

(RAM kayıt numarası) hangi kayıdın okunacağını belirler. Eğer parametre konmazsa, son kayıt numarası kullanılır. Son kayıt numarası bir sonraki kayıt numarasına ayarlanır. Eğer kayıt içeriği (dizi değişkeni)'ni tamamen doldurmuyorsa eski karakterler (üzerine yazılmamış) (dizi değişkeni)'nin sonunda kalır. Eğer kayıt içeriği (dizi değişkeni)'nin boyundan uzunsa artan karakterler atılır. Harici komut sırasında dizi değişkeninin boyunu arttırmak mümkün değildir. (geri dönüş kodu) bir tamsayı değişkenidir. Okunan kayıt numarasını eğer işlem başarılıysa, geri getirir veya işlem, herhangi bir sebepten başarısızca negatif hata kodu verir.

-1 Dosya sonu hatası. İstenen kayıt numarasının adresi 64K'yı aşmış.

-2 Banka anahtarlama hatası. (Hiçbir zaman olmamalı)

ÖRNEK: IBANKREAD, @r%,i$,Ø

Arama...

Depolanmış kayıtları, belli bir giriş yeri aramak için araştırmak mümkündür.

KOMUT IBANKFIND, @(geri dönüş kodu), (dizi için araştır) [,(başlangıç kayıt numarası) [,(bitiş kayıt numarası]]

Bütün RAM kayıtlarını tarar. (başlangıç kayıt numarası) aramanın hangi kayıttan başlayacağını gösterir. Eğer parametre konmamışsa son kayıt numarası kullanılır.

Araştırma, (RAM kayıt uzunluğu)'nda basamak basamak, tam bir kopyası bulununcaya kadar ikinci 64K boyunca ilerler.

Eğer, (bitiş kayıt numarası) belirtilmemişse araştırma, bu kayıt incelendikten sonra sona erer. (Tam bir kopya bu noktadan önce bulunmamışsa)

Eğer araştırma başarılıysa, son kayıt numarası, kopyanın bulunduğu kayıttaki noktaya ayarlanır. (Aksi takdirde etkisiz kalır)

(geri dönüş kodu), kopyanın bulunduğu kayıt numarasını getiren bir tamsayı değişkenidir. (Eğer operasyon başarılıysa) veya operasyon başarısızsa (herhangi bir sebepten dolayı) negatif bir hata kodu verir.

-1 Dosya sonu. Başlanıç kayıt numarasının adresi 64K'yı aşarsa veya bitiş kayıt numarasını aşarsa.

-2 Banka anahtarlama hatası. (Hiçbir zaman olmamalı)

-3 Tam bir kopya bulunamadı.

(dizi için araştır), rastgele kartlar içerebilir, (sıfırlarla işaretlenmiş -CHR$(Ø)) ve karşılaştırma (RAM kayıt uzunluğu) ile veya (dizi için araştır)'ın uzunluğu arasında hangisi kısa diye yapılır.

ÖRNEKLER:

```
IBANKFIND, @r%,"123 test",Ø

IBANKFIND, @r%,f$,1ØØ,2ØØ
```

Hatalı Uyuşmalardan Sakınmak...

Yanlış parametre numarası gibi açık hatalar bir 'kötü komut' (Bat Command) gibi rapor edilirler. Bununla beraber harici komutlar mekanizması 'hatalı uyuşma tipi' (Type mismatch) hataları algılamaz ve kullanıcı, parametrelerin doğru şekilde uygulanmasını sağlamalıdır.

Aşağıdaki program, 7 harfli kelimenin anagramlarını içeren bir data ortamını soruşturmak ve kurmak için RAM disk komutlarını kullanır. Uygunluk için araştırır ve siz rastgele kartlar kullanmalısınız. (Anagram: Harflerin sırasını değiştirmek suretiyle bir kelimenin manasını başkalaştırmadır.)

Örneğin, FIGURES kelimesinin ?RUGS?? ile uyuşan anagramları: FRUGSIE, FRUGSEI, IRUGSFE, IRUGSEF, ERUGSFI ve ERUGSIF'dır. Bunları oluşturmak için database biraz zaman alacaktır fakat 64K, doldurmak için çok fazla bir bellektir.

```
10 'ANAGRAMS by ROLAND PERRY
20 ' copyrihgt (c) AMSOFT 1985
30 '
40 'Remember to RUN "BANKMAN" before running program
50 '************************************************
60 '
70 MODE 2
80 DEFINT a-z
90 r% = 0:IBANKOPEN,7
100 INPUT "What 7 letter word to scramble ";s$
110 IF LEN (s$) < >7 THEN 100
120 PRINT"Please wait..."
130 LOCATE 1,15:PRINT "Computing:"
140 FOR c1 = 1 TO 7
150 FOR c2 = 1 TO 7
160 IF c2 = c1 THEN 370
170 FOR c3 = 1 TO 7
180 IF c3 = c2 OR c3 = c1 THEN 360
190 FOR c4 = 1 TO 7
200 IF c4 = c3 OR c4 = c2 OR c4 = c1 THEN 350 THEN 350
210 FOR c5 = 1 TO 7
220 IF c5 = c4 OR c5 = c3 OR c5 = c2 OR c5 = c1 THEN 340
230 FOR c6 = 1 TO 7
240 IF c6 = c5 OR c6 = c4 OR c6 = c3 OR c6 = c2 OR c6 = c1 THEN 330
250 FOR c7 = 1 TO 7
260 IF c7 = c6 OR c7 = c5 OR c7 = c4 OR c7 = c3 OR c7 = c2 OR c7 = c1 THEN 320
```

```
270 o$ = MID$(s$,c1,1) + MID$(s$,c2,1) + MID$(s$,C*,1) + MID$(s$c4,1) +
MID$(s$,c5,1) + MID$(s$   c6,1) + MID$(s$,c7,1)
280 LOCATE 12,5:PRINT x;o$
290 IBANKWRITE,@r%,o$
300 IF r% < Ø THEN STOP
310 x = x + 1
320 NEXT c7
330 NEXT c6
340 NEXT c5
350 NEXT c4
360 NEXT c3
370 NEXT c2
380 NEXT c1
390 lastrec = r%
400 REM now look them up
410 r% =Ø:g$ = SPACE$(7)
420 PRINT*INPUT "What match do you require: use ? as will d card: ",m$
430 m$ = LEF$(m$,7)
440 FOR x = 1 TO LEN (m$)
450 IF MID$(m$,x,1) = "?"THEN MID$(m$,x,1) = CHR$(Ø)
460 NEXT
470 IBANKFIND,@r%,m$,Ø,lastrec
480 IF x% < Ø THEN GOTO 42Ø
490 IBANKREAD,@r%,g$
500 PRINT g$,
510 IBANKFIND,@r%,m$,r% + 1,lastrec
520 GOTO 48Ø
```

KONU 9
Boş Vaktinizde...

Bu ders bazı geçmişteki, genelde hesaplama bölgelerine ve kısmen 6128'e yavaş yavaş göz atar. Bu bölümü bilgisayar ile çalışmadan önce, mutlaka okumanız gerekmez. Fakat bu bilgisayarın içinde neler olup bittiğini biraz anlamaya yardımcı olacaktır.

Bölüm 1: Genel Olarak Konuşursak....

Sıkı Durun!

Akıl durdurucu bilgisayar oyunlarının avantajlarından faydalanmak için 6128'i almış olabilir ve belkide hâlâ 'Hardware' (Donanım) adı altında gelen bilgisayar kavramlarını merak ediyorsunuzdur.

Donanım (Hardware), kaldırabileceğiniz ve taşıyabileceğiniz bir cihazdır. Örneğin Ana bilgisayar tuş takımı, monitör, bağlantı uçları vs. Gerçekten 'Yazılım' programlar, kitaplar, disk ve kaset temelli bilgiler hariç herşeylerdir.

Bilgisayarın davranışı yoluyla bazı özellikler donanımın sayesinde üretilmiştir. TV'de veya monitördeki renkli görüntüler gibi. Bu donanım becerilerini kullanarak özel tasarlanmış karakterleri ve şekilleri ekranda üreterek yazılıma (Software) bağlıdır.

Donanım gerçekte aydınlatmak için TV tüpünün ekranının içindeki ışık neşreden yüzüne elektron demetlerini yöneltir, yazılım bunu nasıl ve ne zaman gerçekleştireceğini emirler ilave ederek zekice donanıma söyler. Zamanlama kontrol ve sıralamayı, uzay gemisinin kalması için veya tuş takımından bir harfi yazdığınızda görünmesi için ilave eder.

Bir Bilgisayarı Diğerlerinden Daha İyi Yapan Nedir?

Yazılımsız donanım işe yaramaz. Donanımsız yazılımda işe yaramaz. Bilgisayarın değeri ikisinin beraberce vazifelerini beraberce yapmalarıyla başlar. Donanım ve yazılım performansını derecelemek için bazı genel olarak kişisel bilgisayarlar için kabul edilen referans noktaları şunlardır.

1. Ekran Hassasiyeti-Görüntüdeki En Küçük Görünebilir Parça.

Programcı için elde edilebilir renklerin sayısı ve tek ekran alanında görüntülenebilen yazı karakterlerinin sayısı dahil faktörlerin konbinasyonudur.

Kolayca anlaşılabileceği gibi, 6128, bu münasebetlerin her birinde benzer fiatlı makinalara oranla daha iyidir.

2. BASIC Yorumlayıcısı (Çeviricisi)

Hakikatte her ev tipi bilgisayarın, donanım özelliklerini kullanmak için kullanıcıya programlar yaratmaya başlamasına izin veren BASIC çeviricisi vardır. Yapısının içinde olan programlama lisanı (BASIC) makinanızla sunulmuştur ve kendisi bir programdır. BASIC, dünyada geniş çapta kullanılan basit bir bilgisayar lisanıdır ve her lisan gibi genel lehçelerin değişikliğinden meydana gelmiştir.

6128'deki tercüme BASIC'ın en uygun lehçelerinden biridir ve CP/M disk çalışma sistem altında çalışan programları işletir. BASIC'ın çok süratli tamamlayıcısıdır, diğer bir deyişle hesaplarını çok çabuk tamamlar. Sizi ilgilendirmemekle beraber bir bilgisayar 3 ile 5'in çarpımını 0,05 saniyede yaparken diğer bir bilgisayar aynı işlemi 0,075 saniyede yapar ki ekrana grafik çizen bir program binlerce basit hesaplamaları çağırır, 0,05 ile 0,075 saniyenin farkı sonunda ekleyerek düşünmeye değer bir performans farklılığı ortaya çıkar.

Siz sık olarak "Makina kodu" teriminin kullanıldığını duyacaksınız. Makine kodu, talimat kodunun işlemciden geçebilen ham halidir. Bu bir şey sorulduğunda daha az bir çalışma zamanı alır ve BASIC çeviricisinden geçerken 5 ile 15 defa arası daha çabuk neticeyi verir. Diğer taraftan aynı işi yaptırmak için makine koduyla yazılan program, BASIC kullanarak yazılandan 5 ile 50 defa daha uzun zaman alır.

Amstrad bilgisayarınızdaki BASIC, diğer bilgisayar sistemleri arasında bulunan en hızlısı ve on özellik dolu olanıdır ve tecrübeli BASIC programcısının yüksek seviye lisansının çeviricisinin tabiatında bulunan ağırlığının üstünden umulmaz dinamik görüntü ve müzikal tesisleri gerçekleştirmesine yardımcı olan bir çok hususiyetleri birleştirmiştir.

3. Genişleme Kabiliyeti

Bir çok bilgisayar, yazıcı, oyun çubuğu, extra disk sürücüsü gibi ilave donanım parçalarına gereksinme duyarlar. Mantığa aykırı gibi görünmesine rağmen bazı başarılı bilgisayarlar, basit bir yazıcı veya oyun çubuğu gibi ilave donanım parçalarının takılması için dahi genişleme ara birimi adı verilen ek birimlere ihtiyaç duyarlar.

6128 bilgisayarının özelliği, yapılmış centronics yazıcı kapısı, ilave disk sürücüsü için bir kapı, bir kaset data (+motor kontrol) soketi, iki oyun çubuğu, stereo ses çıkışı, sesi interface'e takmak için kullanılan geniş kapsamlı çıkış soketi (AMSTRAD model RS232C), bir MODEM bir komuşma sentesizörü (AMSTRAD model SSA2), bir ışıklı kalem vs. gibi çok geniş kapsamlıdır.

4. Ses

Bir bilgisayarın ses özellikleri, sesin sinek vızıltısı gibi veya bir elektronik müzik enstrumanı gibi çıkmasıyla hesaplanır.

6128 bilgisayarı 8 oktav 3 kanal ses üretici kullanır ki genlik ve ton zarflarında tam bir kontrol ile uygun bir müzikal kaliteyi üretir. Bundan başka ses, bir kanal sol çıkış, bir kanal sağ çıkış ve üçüncü kanal ortada kalarak stereo olarak ayrılır.

Bu size sesin hareketi takip etmesini sağlayan program yazma olanağı sağlar.

Neden Yapamasın?

Bu günün modern teknolojisinin kuvvetiyle, kullanıcılar daima 6128 gibi gelişmiş bilgisayarların neden işleri TV'deki gibi yapamadığını merak ederler. Örneğin neden bir bilgisayar ekranda yürüyen bir adam resmini tabii bir şekilde canlandıramıyor?

Niçin bütün bilgisayarın sundukları haraketli şekiller kibrit çöpünden yapılmış gibi?

Bu sorunun cevabı oldukça karışıktır. Fakat sakın bilgisayarınızın ekranının TV ekranının zekasında olduğunu zannetme aldatmacasına düşmeyin. Bir televizyon cihazı tayfın bütün renkleri boyunca aydınlık ve karanlık uçları arasındaki hassasiyetin sonlu alanını tarif eden lineer bilgileri kullanarak çalışır. Bu çalışma, bilgisayar tekiminde, TV resminin görüntü hafızası ev tipi bilgisayar görüntüsünden yirmi defa büyüktür demektir.

Bu problemin sadece bir bölümüdür ki bu resmi canlandırmak için muazzam miktarda bellek yüksek bir süratte (her saniyede 50 defa) işlenmelidir. Bu yapılabilir fakat ev bilgisayarından bir kaç bin defa daha pahalı bir bilgisayar tarafından yapılabilir.

Tuş Takımı Yabancı Gelmiyor....

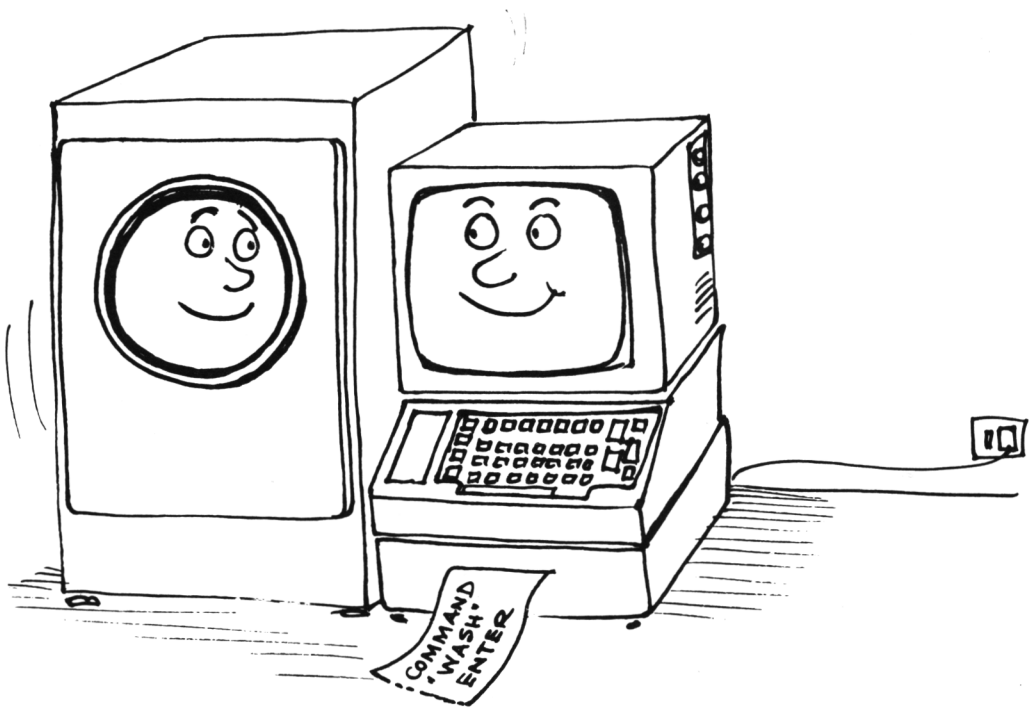
Bilgisayarın elektronik göstergesi olan bir daktilo olduğu gibi yanlış bir fikre kapılmayın. Ekran elektronik bir kağıt değil, bir komut konsoludur. Bu makine hafızasındaki programlama lisanı ile bağlantı imkanı vermesi demektir.

Bilgisayar tuş takımından yazdığınız bütün karakterleri program talimatları gibi çevirmeye uğraşır. (RETURN) tuşuna bastığınızda bilgisayar ne yazdığınıza bakar ve BASIC ile alakalı görmezse girişi kabul etmez ve şu hatayı mesajını verir.

Syntax error

Bununla beraber bilgisayarınızda bulunan bir kelime işlemci sistemi her hangi bir kelime veya kelimeler dizisi yazıp (RETURN) tuşuna bastığınızda, sistemi sanki bir elektronik daktilonun elektronik kağıdı gibi alıştırır. Fakat bunu yapmak için önce kelime işlem programının tamamını makineni hafızasına yüklenmelidir.

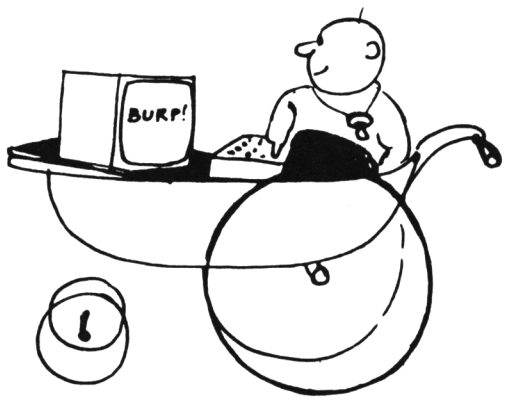
Bilgisayar ev ve ofiste, TV gibi alışılagelmiş bir cihaz haline geleceğe benziyor.



Kim Korkar Anlaşılmaz Dilden

Mütehassıs endüstrilerde olduğu gibi, hesaplama, basit lisan açıklamaları için bir çok kelime gerektiren mükemmel haberleşme kavramının kısa formu olarak kendi anlaşılmaz dilini geliştirmiştir. Bu sadece yüksek teknoloji işi değildir, bir terminolojinin karışık lehçe ve fısıltı kelimelerinin görünebilir buzlu ekranı arkasında kendini saklamanın suçlusudur. Bir çoğumuz, temel meslekler ve sanatların hepsi tarafından anlamayı kaldıran engellerle karşı karşıya gelmişizdir.

Esas farklılık şudur: Geçerli karışık lehçelerin konuşturulması kelimelerin kullanılma yolundan gelir. Hesaplama durumunda olduğu gibi kelimelerin kendisinden değil hesaplama terminolojisi ile yakın yetişen bir çok insan kominikasyonun karmaşıklığını minimize yapmak için en doğru tarzda kelimeleri kullanmak için yollarından çıkarlar. Basit lisanın hesaplamada kullanıldığı gibi yanlış fikre kapılmayın, bu harfi harfine bir konu değildir, fakat tam bir ilimdir ve konuşmanın söz dizisinden farklıdır, kominikasyonun yapısı çok doğrudur ve konuşturan veya iki manalı değildir.



BASIC'in Temeli

Hakikatte tüm ev bilgisayarları, BASIC diye bilinen lisanı kullanırlar. BASIC ile oldukça zor ve güçlü programlar yazılmaktadır.

Bununla beraber hiç şüphe yok ki adı yeni başlayanlara, bilgisayar program lisanlarındaki labirentin başlangıç yeri gibi çekici gelecektir.

BASIC müsade edilen komutlar ifade eden bir bilgisayar lisanıdır ve program çalışırken veriler üzerinde işlemleri gerçekleştirir. 5000-8000 kelime arasında olan normal lisanın kelime haznesine karşılık BASIC'de kelime naznesi sadece 200 civarındadır. BASIC kullanarak yazılan programları, bu kelimelerin kullanılmasıyla ilgili katı kurallar takip etmek zorundadır. Dizilim kesindir ve harfler veya konuşma diline ait ifadelerle bilgisayarla ilişki kurmaya kalkılırsa sonuç şu olur;

```
Syntax error
```

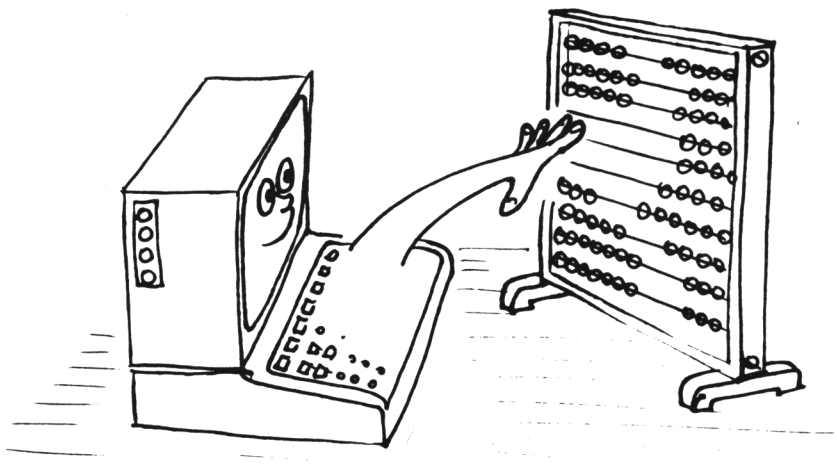
Bu ilk gördüğünde o kadar bağlayıcı değildir. BASIC lisanı öncelikle sayıları (sayısal verileri) işletmek için tasarımlanmıştır. Kelimeler esasen matematiksel işlemlerin (+ /- vs.) uzantılarıdır ve yeni başlayanlar için en önemli kavram bilgisayarın sadece sayısal verilerle çalıştığı gerçeğini kavramasıdır. CPU (merkezi işlem birimi) entegre devresine uygulanan bilgiler sadece sayısal veriler şeklindedir.

Numara Lütfen

Eğer bilgisayar, Shakespeare'in tüm çalışmalarını depolamak için kullanılırsa sistemin hiçbir yerinde tek bir harfe rastlayamazsınız. Her bilgi parçası önce bilgisayarın, yerleştirebileceği ve işleyebileceği gibi sayılara çevrilir.

BASIC kelimeleri, sayılara çevirir sonra bilgisayar sadece toplama, çıkartma ve Boolean mantığını kullanarak verileri karşılaştırır ve belli vasıfları seçer diğer bir deyişle eğer bir sayı diğerinden büyük ve eşit olup olmadığını görmek için veya bir sayının belli bir kiriterde olup olmadığına göre tercih yapar.

Program çalışması sırasında bilgisayar her işlemi basit evet/hayır sorularına çevirir.



Hesaplama hakkında ilk ve en önemli gerçeği elde etmediğiniz için bu işlemler size sıkıcı gelebilir. Bilgisayar öncelikle, basit ve tekrarlanan işleri kesin bir doğrulukla gerçekleştiren bir araçtır. BASIC, program formunda verilen talimatları çevirir ve CPU tarafından ele alınabilecek lisana tercüme eder. Bilgisayar mantığı tarafından, binary notasyonunca "1" ve "0" diye gösterilen iki hal (evet veya hayır) anlaşılır. Boolean mantığında "doğru" ve "yanlış" vardır, "belki" veya "olabilir" gibi şeyler yoktur.

Bu kesin iki hal arasındaki anahtarlama işlemi "sayısal" terimin aslıdır ve bazan "anahtarlama" isnat edilir. Tabiatta birçok işlem tamamen durgun bir halden diğerine lineer bir ilerleme ile hareket eder. İdeal sayısal çevrede, bir halden diğerine anahtarlama yapılması zaman almaz fakat yarı iletken bilminin fiziği önemsiz bir gecikmenin olacağını söyler ki buna yayım gecikmesi denir. Bilgisayarın cevap vermeden önce bilgilerin işlenmesi sırasında biraz zaman harcamasının nedeni bu yayım gecikmelerinin toplanmasıdır.

Herhangi bir durumda bilgisayar ilk vazifenin neticesi üzerinde çalışmaya başlamadan önce bir işde sınırlı bir süre bekler, yani böyle yapay bir gecikme de vardır.

Sayısal çalışma siyah ve beyazdır, grinin değişik gölgeleri yoluyla geçiş halinin bir önemi yoktur. Ters olarak lineer veya analog ilerleme grinin gölgesidir.

Eğer en son cevap 0 veya 1 ise 'yaklaşık olarak doğru'' olan ihtimali yoktur. Bilgisayar, işleyebileceği sayıların boyutunun limitasyonundan dolayı sayısal veriyi ele alırken hata yapıyormuş gözükür, gereken aşırı büyük (boyut olarak) veri, mevcut yere uydurulmak için sıkıştırılabilir veya kesilirler. Örneğin yuvarlama hatalarına bir örnek verecek olursak: 999999999 sayısı, 1000000000 haline gelir.

Diğer bir deyişle sadece 0 ve 1 sayıları mevcutsa 1'den öteye nasıl sayarsınız?

Bit ve Bayt

Biz ondalık sisteme dayalı sayıları anlamaya alışmışızdır.Bu sistemde referans sayısı 10'dur. 0'dan 9'a kadar 10 hane vardır. (Bunu 1-10 olarak kullanmak genelde tercih edilir) Sayıların 0'dan 1'e kadar sıralanmasının olduğu sistem binary sistemidir ve bu sistemin çalıştığı birimlere BİT (Binary diğit) denir. Bit'lerle ondalık notasyon arasındaki ilişki kolayca anlaşılabilir:

Binary diğitlerin en büyük sayısını bildirmek için önüne sıfırlar koyarak sayıyı yapmak aslında alışılmış bir şeydir. Örneğin: Ondalık 7 sayısı, beş bit notasyonu kullanarak;

00111 binary olur.

Binary sistemde figürler, verilen 2'nin üssünün varolup (1), olmadığını (0) belirtici olarak kabul eder.

$2^0 = 1$
$2^1 = 2 = 2 = 2(2^0)$
$2^2 = 4 = 2x2 = 2(2^1)$
$2^3 = 8 = 2x2x2 = 2(2^2)$
$2^4 = 16 = 2x2x2x2 = 2(2^3)$

Bu durumda kolonlar şöyle gözükür:

2^4		2^3		2^2		2^1		2^0		
1		0		0		1		1		
(16	+	0	+	0	+	2	+	1)	=	19

Binary diğit bilgilerinden söz ederken daha kısa bir metod sağlamak için BAYT terimi kullanılır. Bayt 8 bit'lik bilgidir. Bir bayt'ta depolanabilecek en yüksek binary sayı 111111111, yani ondalık olarak 255'dir. Bu 256 gerçek değişikliği ima eder ki buna 00000000'da dahildir ve bilgisayar içim tam olarak geçerli bir veridir.

Bilgisayar verileri 8 bit çarpımlarıyla idare etmeye eğilimlidir. 256 çok büyük bir sayı değildir, bu sebeple hafızayı idare edebilecek geçerli bir yol bulabilmek için bellek adresleme metodunu önceden hazırlamak için iki bayt kullanılır ki bu bir matris şeklindedir ve bu matrisin elemanlarının yerleştirilebilmesi için dikey ve yatay adresleri vardır.

	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4										
5				1		1				
6										
7										
8										
9										

Bu matris 0'dan 9'a kadar numaraları kullanarak (10 × 10)'a kadar bilgi parçasının yerini bulabilir. 3,5 ve 5,5 pozisyonundaki parça 1'dir.

Böylece 256 × 256'lık bir binary matris, yatay ve dikey kolonlarına 8 bit adresler kullanılarak 65536 yeri ayrı ayrı ele alabilir. Böylece bizim '0' ve '1'imiz 65536 değişik elemandan bir tanesini tanımlayabilme kapasitesine ulaştı.

Binary için ikinci ifade yolu Kilobayt'tır, (Kbayt veya sadece K) ve bu 1024 bayt'tır. 1024, ondalık kullanımdaki kilo terimine en yakın binary çarpımıdır. Bu olay bilgisayarları tarif ederken neden 64K denildiğini izah eder ki 64K'lık bir bilgisayarın gerçek hafızası (belleği) $64 \times 1024 = 65536$ bayt'tır.

BASIC yorumcusu bütün bu gerekli değiştirmeleri sizin için yapar ve binary sistemi tam anlamadan da çok verimli bir programcı olmak mümkündür. Ama binary'nin anlamını bilmek, bilgisayar ilminde çalışmalarınız ilerledikçe kaçınılmaz olarak ortaya çıkan birçok sihirli veya anlamlı sayıyı anlamanıza yardımcı olacaktır.

Ancak...

Basit ve ince olsa da binary notasyon bir bakışta kolayca okunamadığı için uzun ve hataya yatkındır. Binary ile bağlantılı birkaç tane daha sayma sistemi vardır. Bunlardan biri mikrobilgisayarlarda kullanılan HEX (hexadecimal'in kısa yazılışı) denilen sayı sistemidir. Bu sistemde sayı 16 tabanına göredir ve tek karakterle gösterilir:

Ondalık

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

HEX

0 1 2 3 4 5 6 7 8 9 A B C D E F

Her sistemi bir bayt'ın 8 bitini 4 bitlik 2 bloğa ayırabilir. 15 sayısı 4 bitlik bir 1111 binary'dir. Birinci blok "15"'in tam ünite sayısını, ikincisi "kalanı" belirtir ve burada binary ile hex'in inceliği ortaya çıkar.

Binary notasyonunu tanıtan tabloyu yeniden ele alalım:

ONDALIK	BİNARY	HEXADECIMAL
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10

Bir 8-bit sayı 11010110(&D6 hex)kendi içinde bölünebilir ve nibbles diye bilinen iki 4-bit sayı olarak ele alınır. Bu kitapda hex tabanlı sayılar "&" sembolüyle tanatılacaktır. Örneğin: &D6 ve bu sayı tabanı, assembly dil programlama tekniğini kullanan programcıların çok kullandığı bir sayı tabanıdır. Bir assembly dil programı, çoğu programcının doğrudan makina kodu programlarına en yakın geldiği programlamadır, çünkü assembly dil programı programcıya, gerçek makina kodu sayılarını belirlemek için basit "mnemonics" harfler kullanmasına izin verir.

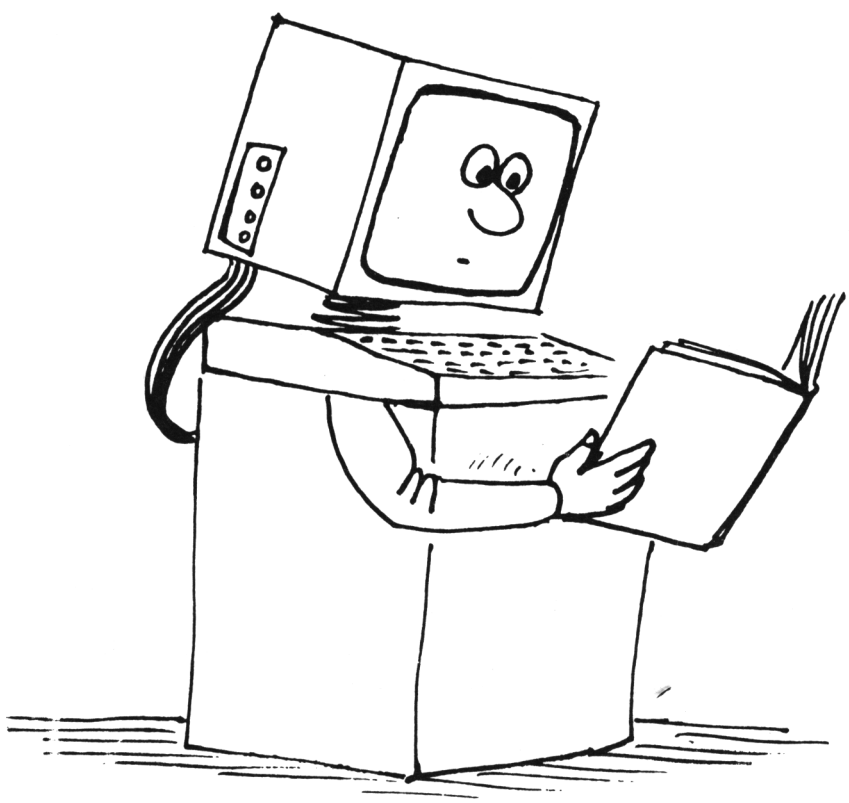
Hex kullanılırken, son sayıda 16 adedini elde etmek için önce ilk hanenin değeri üzerinde çalışmanız gerekir. Sonra toplam ondalık eşdeğerini elde etmek için ise hex notasyonunun ikinci yarısının gösterdiği kalanı eklemelisiniz. Burada D6'nın 13+6 veya 136 olacağı gibi büyük bir yanılgıya teşvik vardır fakat sayı $(13 \times 16) + (6) = 214$'dür.

Burada tek fark 10 ile çarpmanın 16 ile çarpmadan daha kolay olmasıdır ama çok pratik yapılırsa 16 ile de çarpma kolay gelecektir.

Buraya kadar aklınız çok karışmadan geldinizse, bilgisayarın temel prensiplerini kavrama yolunda ilerliyorsunuz demektir. Belkide bu kadar telaş neden diye düşünüyorsunuz ve haklı da olabilirsiniz.

Bir bilgisayar çok basit kavram ve fikirleri idare eden bir cihazdır, sadece bu işleri büyük bir sürat ile (bir saniyede milyonlarca kere) yapar ve girilen tüm verileri hatırlayabilecek muazzam bir kapasitesi vardır ve sonuca varıncaya kadar binlerce toplamanın binlerce ara sonuçlarını saklar.

Eğer bilgisayarınızla ilgili teoriyi sürdürmek istiyorsanız, bilgisayarcılık hakkında binlerce kitap bulabilirsiniz. Bu kitaplardan bazıları aklınızı daha da karıştırır, fakat birkaç tanesi sayı sistemleriyle, bilgisayarınızın onları işlemesindeki temel ilişkiyi size basit bir yolla anlatacaktır.



Bölüm 2: CPC 6128 Hakkında Daha Fazla Bilgi....

Bu bölüm 6128'in makina özel çehresi hakkında bilgi verecektir. Konuyla ilgili ön bilgilerin "TEMEL KURS"da ve "AMSTRAD CPC 6128 BASIC KOMUTLARININ KOMPLE LİSTESİ" adlı derslerde bulabilirsiniz.

Bu bölümün içeriği:
* Karakter seti
* ASCII
* Değişkenler
* Mantık
* Kullanıcı tarafından tanımlanabilen karakterler
* Yazı formatlanması
* Pencereler
* Kesintiler
* Veri
* Ses
* Grafikler
* Ekstra Belleği Kullanan Grafikler

Karakter Biti....

6128'in tuş takımından bir yazı yazdığınızda tanınan harflerin, numaraların vs.'nin ekranda görüneceğini kabu etmemelisiniz. Daha önce bilgisayarınız daktilo olmadığını tartışmıştık. Gerçekten meydana gelen, elektriksel anahtarlar kombinasyonuna basmanız sonucudur. Anahtarlara bastığınızda üretilen elektriksel sinyaller ekranda noktalardan meydana gelen şekillere, cihazın içindeki devreler tarafından çevrilirler. Biz bu nokta şekillerini; harf, sayı veya 6128'in karakter setindeki bir karekter olarak tanırız.

Gördüğünüz gibi bazı karakterler tuş takımındaki tuşlara basmakla doğrudan çıkmazlar fakat sadece PRINT CHR$ (sayı) komutunu kullanarak görünmeleri sağlanabilir. Bunun sebebi bilgisayarda saklanan her eleman, veri kısımlarında "bayt" olarak depolanır ve daha önce bahsettiğimiz gibi bir bayt 256 değişik değer komsinasyonuna sahiptir. Bilgisayar sakladığı her karakter için en azından bir bütün bayt kullandığı için biz de standart daktilolarda 96 karakter olduğu gibi 256 mümkün kombinasyonun hepsini kullanabiliriz.

Standart karakterler "altset" olarak bilinirler. Bu karakterler bilgisayar dünyasında "ASCII" görüntü sistemi diye isimlendirilir ve "American Standart Code for Information Interchange"in baş harflerinden oluşmuştur. Esas amacı bir bilgisayara gönderilen mesajın anlaşılır olmasını sağlamaktır. "Referansınız İçin" adlı dersde ASCII kod sıralamasını, 6128'de bulunan ek karakterler ve bunların sayısal kodları listesini bulabilirsiniz.

Oraya Nasıl Ulaşırız

Artık siz karakter setini ekranda görüntüleyen program biliyor olmalısınız

```
10 FOR n = 32 to 255
20 PRINT CHR$(n);
30 NEXT
```

Şimdi bu küçük programın mahiyetini inceleyelim. Önce dikkat edeceğimiz ilk nokta bilgisayara PRINT "abcdefghijklmn....vs" komutu değilde PRINT CHR$(n) komutunun verilmesidir. n sadece değişkeni belirten bir harftir. Bir değişken, programda verilen talimatlara göre değişen bilgisayar bilgi parçasıdır (n) harfinin seçimi rastgeledir, herhangi bir harfte de (harfler) olabilir.)

Değişkenin Ne Olduğunu Nasıl Söyleyebilirsiniz.

5 sayılı bir sabittir ve 4 ile 6 arasında gözükür. Böylece bir değişken değildir. Aynı şekilde n, harfide alfabede bir harf olduğundan bir sabittir.

Peki bilgisayar bu ayırımı nasıl yapabildi? Eğer n harfi alfabetik karakter olarak söylenseydi, biz n harfini tırnak işareti içinde yazardık. örneğin "n", ve bilgisayar buna syntax error mesajını verirdi, çünkü FOR "n" 32 to 255 komutunu anlayamaz.

Biz n'i tırnak işaretsiz yazarak bilgisayara n'in bir değişken olduğunu söyledik. BASIC'de FOR bildirisinin tanımı ondan sonra bir değişken gelmesini gerektirir, dolayısıyla bilgisayar FOR'un arkasından ne gelirse bunu değişken kabul eder.

Biz aynı zamanda bilgisayara n = 32 to 255 olduğunu da söyledik. Yani değişkenin sıralamasınıda açıkladık. Gerçekten de 32'de başlayıp 255'de biten bu seridir.

Bu değişkeni de açıkladıktan sonra onunla ne yapması gerektiğini de söylemeyiliz. Satır 20 bunu şu şekilde yapar.

```
2Ø PRINT CHR$(n);
```

Bu n'in son değeri ne ise bilgisayara bu değere karşılık hangi karakterin, pafızasında olduğunu bakmasını ve ekrana bu karakteri yazmasını söyler.

Satır sonundaki noktalı virgül bilgisayarın satır atlamasını söylememek içindir. (Aksi taktirde her yeni karakteri yeni satırın ilk kolonuna yazar.)

30'uncu satır bilgisayara n'in ilk değeriyle işlemi yaptıktan sonra (32) FOR'un bulunduğu yere dönmesini ve aynı şeyi n'in bir sonraki değeri ile yapmasını söyler.

Bu çevrim (Loop) olarak bilinir ve bilgisiyar programcılığının ve çalışmasının en temel ve hayati kavramlarından biridir. Bu uzun tekrarlanan dizileri elle yazmanızı önleyerek programınızda çabucak kullanmanızı sağlar.

Bu FOR NEXT çevrimi açıklanan limit değere (255) ulaşınca operasyon durur ve bilgisayar 30. satırdan sonraki satıra bakar (fakat burada yok) ve sona erer ve direkt moda dönerek ready (hazır) yazarak komut bekler.

Eğer siz RUN yazarsanız programı tekrar çalıştırabilirsiniz. Bunlar, artık bilgisayarın belleğinde emniyette olarak, siz silinmesini söyleyince veya elektrik kesilinceye (kapatıncaya) kadar saklar.

Bu program, bilgisayar konusunda bir temel noktayı ortaya koyuyor. Örneğin bilgisayarın her yaptığı şey sayılarla bağlantılıdır. Bilgisayar alfabeyi ve bütün karakter sesini, karakter için referans numarasını kullanarak görüntülenir. A tuşuna bastığınız zaman, bilgisayara A yazmasını söylemeyin fakat bilgisayara A harfini görüntülemek için gerekli sayısal bilgiyi içeren bellek kısmına bakmasını söyleyin. Bu verinin gerçek yeni tuş takımından bir tuşa basmakla harekete geçirilen bir sayısal kod ile tarif edilir.

Her karakterin bağlantılı bir numarası (sayısı) vardır ve bunlar "Referansınız İçin" adlı dersin 3. bölümünde listelenmiştir.

Aynı şekilde görüntülenen karakterler harflerin ekrana yazılmasından başka bir şey yapmaz bunların hepsi sayılardan meydana gelmiştir.

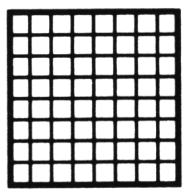
Örneğin A'nın ASCII kodu 97'dir. Bilgisayar 97'yi de anlamak ve bu sayı ondalık sistemden bilgisayarın ilişki kurabileceği bir koda değiştirilmesi gerekir ki buna genellikle Makine kodur denir. Bu özellik altında yatan prensipler bu dersin önceki bölümünde incelenmiştir.

Önceleri günlük hayatımızda kullandığımız ondalık sayı notasyonunu bilgisayarın 16'lık (hexadecimal) sistemine çevirmek işi yokuşa sürmek gibi görünecektir. 10 temeline dayanan sayıları düşünmek o kadar tabiidir ki başka türlüsü çatal ve bıçağı ters tutmaya çalışmak gibi olacaktır.

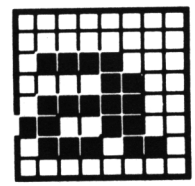
Hex notasyonunu anlamak için belli bir zihinsel ustalığı kazanmış olmak gerekir ama bir kez bunu yaptıktan sonra bilgisayar konusunda bir sürü şey yerine oturacak ve sayısal sistemin zarif yapısı açıkça görülecektir.

Eğer binary ve hexadecimal sayı sistemlerinden hala emin değilseniz bu dersin 1. bölümünü okumanızı tavsiye ederiz.

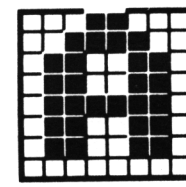
Bilgisayar, A tuşuna basılmasını anlayacağı sayı tipine çevirdiğinde hafızasında belirtilen bölüme bakar ve sonuç karakteri tanımlayan bir başka sayı serisidir. Yani ekranda görüntülenen karakter, hafızada sayısal matris olarak depolanmış bir blok data (veri) oluşmuştur.



A BLANK CHARACTER MATRIX (GRID)



LOWER CASE a



UPPER CASE A

Matris elemanları noktalardan oluşmuş kolonlar ve satırlardır. Karakter gereken dizideki noktaların açık veya kapalı hale çevrilmesiyle görüntülenir. Her noktayı bilgisayarın hafızasında olan veriler saptar. 6128'in görüntü tablosunda her karakter matrisinin veya hücresinin 8 sıra (satır) ve 8 sütunu (kolonu) vardır ve 255'lik karakter setinde istediğiniz karakteri bulamazsınız bu dersin ilerki bölümlerinde bahsedilen SYMBOL tuş sözcüğünü kullanarak kendi karakterinizi tekrar takımlayabilirsiniz.

Kullanıcı tarafından tanımlanan karakterler, rastgele bir düzende ayarlanmış 0'dan 64'e kadar noktadan oluşmuştur. Böylece bu matrisin bütün kombinasyonunu kullanan tam bir karakter seti bir çok değişik karakterler içerebilir. Buna bir de karakter blokları yapmayı eklerseniz kullanıcı tarafından tanımlanan grafik ve karakter sayısı sadece sizin yaratıcılığınızın ve zamanınızla sınırlanmış olur.

Mantık

Bilgisayar ve hesap makinesi arasındaki en önemli fark bilgisayarın IF THEN gibi mantıksal operasyonları yapabilmesidir. Bunu yapabilmek için mantık operatörü (işlemcisi) bit modelleri gibi uygulanan değerlerin üstünde ilerler.

Mantık işlemcisinin arguman olarak bilinen iki girdisi vardır.

argüman [< mantıksal işlemci > < argüman >]

burada arguman :
arguman olmaz
sayısal ifade
bağlantılı ifade
mantıksal ifade

Mantıksal ifadedeki her iki arguman tam sayı olmak zorundadır ve tam sayı olarak bulunmayan arguman olursa ERROR 6 (hata 6) ile sonuçlanır.

```
PRINT 1Ø AND 1ØØØ
```

Sonuç yine 8'dir. Çünkü 1Ø ve 1ØØØ sayıları iki tabanına çevrilir.
1Ø1Ø
111Ø1ØØØ

AND işlemi her iki biti tek tek kontrol eder ve üstteki sıradaki bit ile alt sıradaki bit 1 olduğunda yanıt 1'dir.

```
ØØØØØØ1ØØØ
```

Yukarıda görüldüğü gibi sonuç 8'dir. Mantıksal işlemci AND iki şart ardarda mevcut olduğunu saptamak için kullanılır. Şimdi size bir uygulama örneği:

```
1Ø INPUT "Bugün ayın kaçı"; gün
2Ø INPUT "Hangi aydayız"; ay
3Ø IF gün = 1 AND AY= 1 THEN GOTO 5Ø
4Ø CLS:GOTO 1Ø
5Ø PRINT "Yeni yılınız kutlu olsun"
```

OR ile bitler üstünde çalışılır. Her iki argümanın biti 0 olmadığı takdirde sonuç 1'dir.

```
PRINT 1ØØØ OR 1Ø
  1ØØ2
```

Bit gösterimi:

```
1Ø1Ø
1111Ø1ØØØ
```

Sonuç:

```
1111Ø1Ø1Ø
```

Ve aşağıdaki program örneğinde:

```
1Ø CLS
2Ø INPUT "Hangi aydaydız"; ay
3Ø IF ay = 12 OR ay = 1 OR ay = 2 THEN 5Ø
4Ø GOTO 1Ø
5Ø PRINT "Havalar soğuk olmalı"
```

NOT işlemcisi argüman içindeki her biti ters çevirir. (0, 1'dir ya da 1, 0'dır)

```
1Ø CLS
2Ø INPUT "Hangi aydayız"; ay
3Ø IF NOT (ay = 6 OR ay = 7 OR ay = 8) THEN 5Ø
4Ø GOTO 1Ø
5Ø PRINT "Bodrum'a gidilecek zaman değil"
```

Ele alınacak diğer bir özellikde istediğiniz kadar sayıda mantıksal şartı ekleyebilmenizdir. (en fazla satır uzunluğu kadar)

```
10 INPUT "Hangi aydayız ";gün
20 INPUT "Kaçıncı aydayız ";ay
30 IF NOT (ay = 12 OR ay = 1) AND gün = 29 THEN 50
40 CLS:GOTO 10
50 PRINT "Bu ay OCAK veya ARALIK değil fakat ŞUBAT 29 çekebilir"
```

Bağlantılı ifadenin sonucu ya -1 ya da 0'dır. -1 için bit tanıtımı bütün tamsayı bitleri = 1'dir; 0 için bütün tamsayı bitleri = 0'dır. Bu tür iki argüman da mantıksal işlem sonucu Doğru için -1, Yanlış için 0'dır. Bunu yukarıdaki programa 60. satırı ilave ederek deneyiniz:

```
60 PRINT NOT (ay = 12 OR ay = 1)
70 PRINT (ay = 12 OR ay = 1)
```

Program çalıştırıldığında gün için 29, ay için de 2 girersek 50. satırdaki cevap meydana gelir ve 60. ve 70. satırlardaki mantık ifadelerinden dönen esas değerler altalta görüntülenecektir.

Son olarak XOR'u görelim. Bu işlemde her iki argüman farklı olursa doğru (1) sonucu verilir.

Aşağıdakiler doğruluk tablosu diye bilinen bu özellikleri listeler.

Argüman A	1010
Argüman B	0110
AND işlemi	0010
OR işlemi	1110
XOR işlemi	1100

Kullanıcı Tarafından Tanımlanan Karakterler

Binary sayılarının ilk uygulamalarından biri SYMBOL komutunu kullanarak karakter tasarlamaktır. Eğer karakter 8×8 matrisi üzerine çiziliyorsa sekiz satırın herbiri binary sayıya çevrilir ve 1 konarak pixel son ink rengine ayarlanır. 0 konulursa pixel son sayfa rengine ayarlanır. O konularda pixel sayfa rengine ayarlanarak görünmez yapılır. Bu sekiz sayı SYMBOL komutuna geçmiş olur. Bir ev karakteri tanımlamak istersek:

```
   *      = 00001000 = &08 =                  8          = 8
  ****    = 00111100 = &3C =          32+16+8+4          = 60
 *    *   = 01000010 = &42 =      64                 +2  = 66
* *  * *  = 10100101 = &A5 = 128    +32       +4     +1  = 165
*      *  = 10000001 = &81 = 128                     +1  = 129
* ** * *  = 10110101 = &B5 = 128    +32+16    +4     +1  = 181
* **   *  = 10110001 = &B1 = 128    +32+16           +1  = 177
******** = 11111111 = &FF = 128+64+32+16+8+4+2+1     = 255
```

```
SYMBOL 240,8,60,66,165,129,181,177,255 veya;

SYMBOL 240,08,&3C,&42,&A5,&81,&B5,&B1,&FF veya;

SYMBOL 240,    X00001000, X00111100, X01000010, &X10100101,
               X10000001, 10110101, 10110001, X11111111
```

Komutları yukarıdaki 3 şekilde verebiliriz. Kullanıcı tarafından tanımlanan karakterleri yazmak için ise şunu yazınız:

```
PRINT CHR$ (240)
```

Son olarak, karakterlerin bloklarını biraraya gruplandırmak için şöyle belirtebiliriz:

```
semi$ = CHR$(240) + CHR$(240)
PRINT semi$
```

veya;

```
terrace$ = STRING$(15,240)
PRINT terrace$
```

Yazı Baskısı....

PRINT, programlamaya başlarken öğreneceğiniz ilk komuttur. PRINT söyleneni yapan komutlardan biridir. Bu komut ilk göründüğünden daha fazla şey kapsar. Mesela; nereye yazılacak ve nasıl yazacak gibi...

Yazı Formatlaması

PRINT komutunun kullanılabileceği birkaç yol vardır. En basiti, yazılacak parçanın komutu takip etmesidir. Bu parça bir sayı, bir dizi veya bir değişken adı olabilir

```
PRINT 3
3
PRINT "merhaba"
merhaba

a=5
PRINT a
5

a$="amstrad"
PRINT a$
amstrad
```

Birkaç parça bir ayırıcı veya TAB veya SPC tarafından ayrılarak aynı PRINT cümlesinde yer alabilir. Ayırıcılar (;) veya (,) olabilir. (;) doğrudan yazının devamını sağlar. (,) ise bir sonraki alanda devamını sağlar. Başlangıç olanı (zone) genişliği 13'dür fakat ZONE komutu kullanılarak değiştirilebilir.

```
PRINT 3;-4;5
3 -4 5

PRINT "merhaba ";"amstrad"
merhaba amstrad

PRINT "merhaba","amstrad"
merhaba         amstrad
PRINT 3,-4,5
3           -4          5

ZONE 4
PRINT 3,-4,5
  3     -4    5
```

Negatif sayılar önündeki (—) işareti ile, pozitif sayılar ise önlerinde bir boşluk ile yazılırlar. Diziler, tırnak işareti içinde nasıl görüküyorlarsa ayen yazılırlar.

SPC, fonksiyonu sayısal ifadeyi parametre olarak alır ve ifade tarafından belirtilen sayıda boşluk bırakır. Eğer değeri negatif ise sıfır kabul edilir, eğer son kanal (pencere) genişliğinden büyükse kanal genişliği kabul edilir.

```
PRINT SPC (5) "merhaba"
     merhaba

x = 3
PRINT SPC (x*3) "merhaba"
          merhaba
```

TAB hemen hemen SPC'in aynısıdır, sadece istendiği kadar boşluk bırakır böylece yazılacak parça belirtilen kolonda gözükür. Parça listesi yazılmadan önce kanal yöneticisi (#) ilave edilmemişse, bütün yazılmış çıkışların göründüğü kanal pencere 0'dır. Diğer kanallar diğer pencereler için kullanılabilirler. Kanal 8 ve 9 özel durumlardır. 8. kanaldan yazılan herhangi birşey, eğer yazıcı bağlı ise yazıcıda gözükür. Kanal 9, çıkışları disk veya kaset dosyasına yöneltir. Bununla beraber WRITE komutu, bu amaçla, PRINT'in yerine kullanılmamalıdır.

```
PRINT "merhaba"
merhaba                    - pencere 0

PRINT # 0, "merhaba"
merhaba                    - yine pencere 0

PRINT # 4, "merhaba"
merhaba                    - pencere 4
                           (ekranın en üstünde)

PRINT # 8, "merhaba"
merhaba                    - yazıcıda
                           (eğer bağlanmışsa)
```

TAB ve SPC basit yazı formatları için uygundur, fakat daha fazla detaylı format belirlemek için PRINT USING komutu uygun bir format kalıbıyla birlikte kullanılabilir. Bir format kalıbı, formatın bir parçısının tipini belirten özel karakterler içeren karakter dizisinden ibarettir. Bu karakterlere "FORMAT ALAN BELİRTİCİLERİ" denir ve bu kitabın başlarında PRINT USING komutunun tarifinde detaylı olarak anlatılmıştır. Bununla beraber aşağıdaki örnekler bunların kullanılmasına biraz daha açıklık getirecektir. İlk önce dizilerin yazılması için elde olan formatlar şunlardır:

```
PRINT USING " \       \ ";"test dizisi"
test d
```

''!'' işareti, karakter dizisinin ilk karakterini yazmak için kullanılabilir.

```
PRINT USING "!";"test dizisi"
t
```

Fakat en kullanışlı dizi formatı " "dır. Bu gerekirse BASIC'in dizi katlama özelliğini önemsiz kılmak için kullanılır. Önceden belirlenmiş olarak, BASIC, herhangi bir diziyi eğer son satıra sığamayacak kadar uzunsa yeni bir satırın üzerine yazmaya başlar. PRINT USING " " bunu engellemek için kullanılabilir.

(BORDER Ø'ı, sayfanı kenarlarını görmek için kullanın)

```
MODE 1: LOCATE 39, 1:PRINT "çok uzun"

                                                    — satır 1
çok uzun                                            — satır 2
```

```
MODE 1: LOCATE 39,1:PRINT USING "&";"çok uzun"

                                                  ço — satı 1
k uzun                                              — satır 2
```

Rakamların basılabilmesi için çok sayıda form vardır. Bunlardan en basiti, PRINT USING " ##### "dir. Kalıpta gözüken her " # " için bir hane yazılır.

```
PRINT USING "  ###/###  ";123
  123
```

Ondalık noktanın yeri, "." kullanılarak ilave edilebilir.

```
PRINT USING " ####  . ##### ";12.45
  12.45ØØ
```

Ondalık noktadan önceki haneler üçlü gruplara, (binler, yüzler vs.) "," ondalık noktadan önce konularak ayrılabilir.

```
PRINT USING "  ######## ,.   #### ";123456.78
  123.456.78ØØ
```

Hareketli dolar ve pound işaretleri formata ilave edilebilir. Örneğin: Dolar veya pound işareti, sayı tüm formatı doldurmasa dahi sayının ilk hanesinden önce yazılmalıdır. Bu, "$$" ve "££"in kalıpta kullanılmasıyla harekete geçirilir.

```
PRINT USING "$$ ##";7
  $7

PRINT USING "$$ ## ";351
$351

PRINT USING "££  #### ,. ## ";1234.567
  £1,234.57
```

Burada sonucun yuvarlanmasına dikkat ediniz.

Sonucun önündeki boşluk, "**" hareketli asteriks kullanılarak doldurulabilir.

```
PRINT USING "******.#";12.22
****12.2
```

Bu, dolar veya pound işareti ile birleştirilebilir. (O zaman tek bir para işareti kullanılır) örneğin: "**$...." veya "***£...."

Kalıbın başlangıcındaki bir "+" işareti daima sonucun ilk hanesinden önce işaret konmasını belirtir. Kalıbın sonundaki "+", arkasında işaret koyar.

Bir "-" işareti sadece formatın sonunda yer alabilir ve eğer sayı negatif ise sonuna eksi işareti koyar.

```
PRINT USING "+ ## ";12
 +12

PRINT USING "+ ## ";-12
-12

PRINT USING "  ## +";12
12+

PRINT USING "  ## -",-12
12-

 PRINT USING "  ## -";12
 12
```

A " ↑ ↑ ↑ ↑ " şeklindeki formak, kalıba; sayıları üstel format olarak yazmasını sağlar.

```
PRINT USING "###.##↑↑↑↑";123.4
 12.35E+Ø1
```

Sayılar için yazı formatlarını kullanırken eğer sayı belirtilen kalıptan çok uzunsa % sembolü bu durumu belirtmek için sonuçtan önce yazılır ve sonuç, belirtilen kalıba uydurulmak için kısaltılmaz.

```
PRINT USING " ##### ";123456
%123456
```

Pencere....

6128'in BASIC'i, 8 metin penceresinin seçilebileceği kapsamlı bir metot hazırlamıştır. Metin ekranı sürücü komutlarından herhangi biri, bu pencerelerin birine yöneltilebilir. Pencere teşkil etmek için kullanılacak komut sadece WINDOW'dur. Bu komut 5 değer tarafından izlenir. 1.si isteğe bağlıdır ve hangi pencerenin tanımlanacağını belirtmekte kullanılır. Eğer kullanılmazsa pencere 0 kabul edilmiş demektir. Bu sayının önünde yer alan işareti kullanılan kanalı tanımlar. Sonraki 4 sayı pencerenin sol, sağ, üst ve alt sınırlarını belirtir. Bu değerler kolon ve satır numaralarıdır. Bundan dolayı sol ve sağ değerler 1 ile 80, üst ve alt değerler ise 1 ile 25 arasında olmalıdır.

Aşağıdaki örnek 4 numaralı pencereyi tanımlar ve 7. kolondan başlayan (sol) ve 31. kolona kadar (sağ) devam eden ve 6. satırdan başlayan (üst) ve 18. satıra (alt) kadar devam eden bir penceredir. Şimdi bilgisayarı, resetleyin ve şu satırı yazın:

```
WINDOW 4,7,31,6,18
```

Bu komuttan sonra hiçbir şey gözükmedi fakat şunları yazarsanız:

```
INK 3,9
PAPER # 4,3
CLS # 4
```

Bu komutlar ekranda büyük yeşil bir komut görünmesine neden olur ve 4 nolu penceredir. CLS ve PAPER komutları 8 pencereden herhangi biriyle, kanal numarası ilave edilerek kullanılabilir. Kanal yöneticisinin yazılmaması komutun pencere 0 ile çalışmasına sebep olur.

Aşağıdaki her komut, komutların ele alınacağı pencereyi tanıyan kanal direktörünü içerir.

```
CLS
COPYCHR$
INPUT
LINE INPUT
LIST
LOCATE
PAPER
PEN
POS
PRINT
TAG
TAGOFF
VPOS
WINDOW
WRITE
```

Ekrana koyduğunuz yeni yeşil pencere, önceki yazıların bazısı tarafından (0 numaralı pencereye yazılan) örtülebilir.

Yazı herhangi bir pencereye, PRINT cümlesine kanal direktörünün ilavesiyle yönlendirilir.

```
PRINT # 4, "merhaba"
```

Bu kelime yeşil dikdörtgenin üstünde gözükür ve aşağıdaki satırın yaptığının tersidir.

```
PRINT "merhaba"
```

Eğer bütün BASIC mesajlarının pencere 4'de gözükmesini istiyorsanız, daha önceden belirlenmiş pencere ile (Ø) WINDOW SWAP komutu ile değiş-tokuş edilebilir.

```
WINDOW SWAP Ø,4
```

"Ready" yeşil pencerede komutdan sonra yızılır. Şimdi şunları yazınız:

```
PRINT # 4, "merhaba"
```

"merhaba" kelimesi WINDOW SWAP komutunun hemen altında gözükür ki bu pencere 4'dür. Buradan da anlaşılıyor ki her pencerenin son yazı pozisyonu saklanır böylece WINDOW SWAP'dan sonra yazı yeni kanal 4'ün aşağısına yazılır. (Yukarıdan başlayacağı yerde.) Aşağıdakileri deneyin:

```
LOCATE 4,20,1
PRINT "bu pencere 0'dır."
PRINT # 4, "bu pencere 4'dür"
```

Pencere 0 mesajı PRINT satırından sonra gözükür, pencere 4 mesajı bütün ekranın en üst satırının ortasında gözükür.

WINDOW komutundan önce bütün 8 pencere halihazırda ki ekranı kaplar. Bu MODE komutundan sonra da doğrudur. Böylece pencereleri kullanıldıktan sonra kürsör küçük bir pencerede sona erer. Aşağıdakini yazın.

```
MODE 1
WINDOW 20,21,7,18
MO
DE
  1
```

MODE'un parçalara ayrılmasından dolayı telaş etmeyin, bu hala çalışır ve MODE ile 1 arasında boşluk bırakmayı unutmayın.

Şimdi siz pencerelerin çalışması hakkında biraz bilgi sahibi oldunuz. Aşağıdaki kısa programı yazın.

```
10 MODE 2
20 FOR n=0 TO 7
30 WINDOW n,n+1,n+6,n+1,n+6
40 PAPER N, n+4
50 CLS n
60 FOR C=1 TO 200:NEXT c
70 NEXT n
```

Bu bir öncekinin üstünde oluşan 8 pencere meydana getirir. Program çalışmasını bitirip "Ready" göründüğünde, tekrar (ENTER)'a basınız ve katlanan 0 nolu pencerenin ekrandaki renkli bloklara nasıl etki yaptığını gördünüz. Bununla beraber bu renkli blokları kotlanabilir, diğer pencerelerin konumları gerçekten haraket etmez. Aşağıdakini deneyiniz.

```
CLS # 4
```

Ve 4. pencerenin aynı pozisyonda olduğunu göreceksiniz. Yeni renklendirilmiş blok alttakini kapatır. Aşağıdakileri yazın ve farklılıkları izleyiniz.

```
LIST
LIST # 4
LIST # 3
```

WINDOW komutuyla ilgili başka şeyler bu bölümde final programında gösterilecektir ki pencere boyutlarını ters sırada (sol ve sağ) belirtirseniz önemli değildir. Bu, eğer ilk parametrenin değeri ikincisinden büyükse BASIC otomatik olarak boyutları doğru dizilimde sıraya sokar demektir. Bu alt ve üst pencere boyutları içinde uygulanır.

```
10 MODE 0
20 a = 1 + RND*19:b = 1 + RND*19
30 c = 1 + RND*24:d = 1 + RND*24
40 e = RND*15
50 WINDOW a,b,c,d
60 PAPER e:CLS
70 GOTO 20
```

Kesintiler....

AMSTRAD tipi bilgisayarın yazılımının en önemli yeniliği BASIC'den kesintileri idare edebilme yeteneğidir. Bu AMSTRAD'ın bir programın içinde birkaç tane ard arda fakat ayrı operasyonu gerçekleştirebilmesi anlamına gelir. Bu olanağa bazen, çok görevlilik denir ve AFTER ve EVERY komutlarının uygulanmasıyla kullanılır.

Bu faaliyet sesin, kuyruk ve randevü faaliyetleriyle de açıkça ortaya konulmuştur.

Zamanlamanın her yönü, bilgisayarın içindeki quartz kontrollü zamanlama sistemi olan zamanlama ve senkronizasyon olaylarını idare eder ki görüntünün taranması ve işlemcinin zamanlanması da dahildir. Donanımdaki bir fonksiyon zamanla bağlantılı olduğundan bunu quartz ana saate kadar izlemek mümkündür.

Yazılım tamamlayıcısı AFTER ve EVERY komutlarıdır ki kullanıcının AMSTRAD BASIC'e iyi bir yaklaşımını sağlar yani ne söylenirse yapar. Örneğin emirde zamanı yeniden ayarladıktan sonra program gösterilen alt-programa gönderilir ve burada tarif edilir.

6128, gerçek bir zaman saati donatmıştır. AFTER komutu BASIC programının ileride çağıracağı alt-programı ayarlamasına izin verir. Dört geciktirme zamanlayıcısı vardır ve her birinin bağlantılı alt-programı olabilir. Belirtilen zaman geçin-

ce programın son konumunda GOSUB verilmiş gibi alt-program otomatik olarak çağrılır. Alt programı RETURN komutunu kullanarak bittiği zaman esas program kaldığı yerden çalışmaya devam eder. EVERY komutu BASIC programının düzenli aralıklarla alt-programları çağırmasına imkan sağlar. Dört gecikme zamanlayıcısı vardır ve her birinin bağlantılı alt-programı olabilir. Zamanlayıcıların değişik interrupt öncelikleri vardır. 3. zamanlayıcının en yüksek önceliği ve 0. zamanlayıcının en düşük önceliği vardır. ("Referansınız İçin" adlı ders bakınız)

```
10 MODE 1:n=14:x=RND*400
20 AFTER x,3:GOSUB 80
30 EVERY 25,2 GOSUB 160
40 EVERY 10,1 GOSUB 170
50 PRINT "test your reflexes"
60 PRINT "press the space bar.";
70 IF flag=1 THEN END ELSE 70
80 z=REMAIN(2)
90 IF INKEY(47)=-1 THEN 110
100 SOUND 1,900:PRINT "cheat!":GOTO 150
110 SOUND 129,20:PRINT "NOW":t=TIME
120 IF INKEY(47)=-1 THEN 120
130 PRINT "you took";
140 PRINT (TIME-t)/300;"seconds"
150 CLEAR INPUT:flag=1:RETURN
160 SOUND 1,0,50:PRINT ".";:RETURN
170 n=n+1:IF n>26 THEN n=14
180 INK 1,n:RETURN
```

Veri Kullanımı....

Bir programda daima başlangıçta girilecek bir seri bilgiler gerekir. Bu, her seferinde kullanıcıya yazmasını söylemek yerine herhangi bir yolla bütün değerlerin girilmesi ile daha mantıklı olur. Bu işlem READ ve DATA komutları tarafından önceden hazırlanır. READ kelimesi INPUT'a çok benzer, şöyle ki değişkenlerin değerlerini tesbit ederken kullanılır. Bir farkı tuş takımından her hatırlatmada girmek yerine DATA cümlesinden, değerler okunur. Aşağıdaki iki örnek bunu göstermektedir.

```
10 INPUT "Virgüllerle ayrılmış 3 sayı girin";a,b,c
20 PRINT "Sayılar şunlardır ";a; "ve" ;b; "ve"; c
RUN

10 READ a,b,c
20 PRINT "sayılar şunlardır"; a; "ve"; b;
30 DATA    ,14,21
```

INPUT cümlesinde olduğu gibi değişik parçalar virgüllerle ayrılırlar.

Sayısal değerlere ilave olarak sabit karakter dizileri veya her ikisinin de karışımıda olabilir.

```
10 DIM a$(12)
20 FOR i=0 TO 12
30 READ a$(i)
40 NEXT
50 FOR i=0 TO 12
60 PRINT a$(i);"  ";
70 NEXT
80 DATA Bir, berber, bir, berbere, gel, beraber, berberistanda, bir, berber, dükkanı, açalım, demiş
RUN
```

DATA'nın dizi içerdiğine ve dizilerin çift tırnak (""") içine alınmadığına dikkat ediniz. Çift tırnak kullanmak dizileri ayırmak içindir ve DATA cümlesinde tercihe bağlıdır. Eğer dizinin kendisi virgül içeriyorsa o zaman çift tırnak işareti kullanmak faydalı olur. Şayet böyle bir durumda çift tırnak kullanılmazsa READ cümlesi DATA cümlesindeki virgülleri dizileri ayırmak için kullanılır.

```
10 READ a$
20 WHILE a$ < > "*"
30 PRINT a$
40 READ a$
50 WEND
60 DATA Eski, harap, temelinden meyilli ev rüzgarda yıkıldı
70 DATA "Uzun, ince, siyah adam yüksek sesle öksürdü."
80 DATA *
RUN
```

60. satırdaki dizi virgüller içeriyor ve her bir kısım ayrı ayrı okunup yazılacak. 70. s atır ise çift tırnakla ayrılmış ve hepsi birden okunup yazılacak.

Yukarıdaki örnek data (veri)nin birkaç satıra dağıtılabileceğini göstermektedir. READ numara sırasına göre satırlarda (60, 78, 80 vs) çalışır.Aynı zamanda DATA cümlesi bilgileri, kendinden önce veya sonra olmasına bakmadan okur.

Eğer program birden fazla READ cümlesi içeriyorsa, ikinci READ birincisinin durduğu yerden devam eder. Şöyleki:

```
10 DATA 123,456,789,321,654,2343
20 FOR i=1 TO 5
30 READ num
40 total=total+num
```

```
50 NEXT
60 READ total2
70 IF total = total2 THEN PRINT"veri tamam"
ELSE PRINT"data'da bir hata var"
RUN
```

10. satırda karışık veri gerektiriyorsa (karakter dizisi ve sayı) parçalar doğru okunacak kadar uzunlukta olmak kaydıyla karakter ve sayı dizilerini READ ve DATA cümlesinde birleştirebilirsiniz. örneğin: Eğer bir veri iki sayı dizisini takip eden bir karakter dizisi, ihtiva ediyorsa READ'ı iki sayısal değişken ve bir değişkeni takip etmelidir.

```
10 DIM a(5),b(5),s$(5)
20 FOR i = 1 TO 5
30 READ a(i),b(i),s$(i)
40 NEXT
50 DATA 1,7,fred,3,9,jim,2,2,eric,4,6,peter,9,4,alfonzo
60 FOR i = 1 TO 5
70 PRINT s$(i),":";a(i)*b(i)
80 NEXT
```

Alternatif olarak, değişik tip veri olarak ayırmak isterseniz.

```
10 DIM a(5),b(5),s$(5)
20 FOR = 1 TO 5
30 READ a(i),b(i)
40 NEXT
50 FOR i = 1 TO 5
60 READ s$(i)
70 NEXT
80 DATA 1,7,3,9,2,2,4,6,9,4
90 DATA fred,jim,eric,peter,kamil
100 FOR i = 1 TO 5
110 PRINT s$(i),":";a(i)*b(i)
120 NEXT
```

Eğer 20. satırdaki FOR döngüsü değiştirilip:

20 FOR i = 1 TO 4 yapılırsa, 60. satırın ilk iki küme teşebbüsü sonucu "9" ve "4" okunur. Bu değerler tabii ki geçerli dizidir. Fakat sonuç planlandığı gibi değildir. Programın uygun olarak çalışmasını zorlamak için bir metot da aşağıdaki komutları ilave etmektir.

```
15 RESTORE 80
45 RESTORE 90
```

RESTORE cümlesi veri (DATA)- okuma göstergesini belirtilen satıra götürür ve böylece bazı satırlara dayanarak veri bloklarının okunabilmesi için şartlı cümlede kullanılır. Örneğin: Çok aşamalı oyunlarda birkaç değişik ekran vardır ve her ekran için veri (data) bazı değişkenlere göre alınır. ("level" gibi). Aşağıdaki bu tür program için bir örnektir.

```
1000 REM section to draw the screen
1010 IF level = 1 THEN RESTORE 2010
1020 IF level = 2 THEN RESTORE 2510
1030 IF level = 3 THEN RESTORE 3010
1040 FOR y = 1 TO 25
1050 FOR x = 1 TO 40
1060 READ char
1070 LOCATE x,y:PRINT CHR$(char);
1080 NEXT x,y
:
2000 REM DATA for screen 1
2010 DATA 200,1H0,244,244,210,...etc.
:
2500 REM DATA for screen 2
2510DATA 100,103,245,243,242,...etc.
:
3000 REM DATA for screen 3
3010 DATA 190,191,192,193,194,...etc.
```

DATA READ ve RESTORE'nin kullanılmasıyla ilgili diğer bir örnek de melodi çalan bir programdır. Ton periodu değerleri DATA cümlesinden okunabilir ve RESTORE tekrar bölümlerini, müzik verisinin belirli bir kısmının başlangıcına götürerek, yapmakta kullanılır.

```
10 FOR i = 1 TO 3
20 RESTORE 100
30 READ note
40 WHILE note<>-1
50 SOUND 1, note, 35
60 READ note
70 WEND
80 NEXT
90 SOUND 1,142,100
100 DATA 95,95,142,127,119,106
```

```
110 DATA 95,95,119,95,95,119,95
120 DATA 95,142,119,142,179,119
130 DATA 142.142,106,119,127,-1
RUN
```

Müziğin Sesi....

6128'in ses ve zarf (envelop) komutları ilk bakışta sizi korkutabilir. Biraz pratikle siz birçok ses yapabilir ve hatta biligisayarınıza tüm bir medodiyi harmonikleriyle beraber çaldırabilirsiniz.

Önce SOUND komutunun ilk 4 kısmına bir göz atalım. Bunlar: kanal numarası, ton periodu, notanın süresi ve volum (ses şiddeti)'dir. Her sayının hangi alt ve üst sınır değerlerine sahip olduğunu merak edebilirsiniz.

Kanal numarası olan ilk kısmı şimdilik bırakalım çünkü oldukça karışıktır. İkinci sayı olan ton periyodu kısmı 0'dan 4095'e kadar olan herhangi bir sayı olabilir fakat bunlardan sadece birkaçı gerçekten tanımlanabilir müzik notalarını üretebilir. Bunlar "Referansınız İçin" adlı dersin 5. bölümünde listelenmiştir. Eğer ton periyodu 0 ise hiçbir ton çalınmaz. Bu özellik gürültüyü kullanırken (daha ileride açıklanacaktır) faydalı olacaktır.

SOUND komutunun 3. kısmı saniyenin yüzde biri birimlerde notanın kalış süresini verir. Değeri 1 ile 32767 arasındadır. Bununla beraber eğer değer 0 ise notanın uzunluğu zarf tarafından hesaplanır. Eğer değer negatif ise zarfın kaç defa çalınması gerektiğini belirtir. Yani -3 ses zarfını 3 defa tekrar et demektir. (Bu kısımda daha sonra anlatılacaktır.)

SOUND komutunun 4. kısmı ses şiddetidir. Buranın değeri 0 ile 15 arasında olabilir, belirtilmemiş ise 12 olarak kabul edilir. Basit ses de, ses şiddeti, nota çalındığı sürece sabit kalır. Ses şiddeti zarfı, bunu değiştirmek için kullanılırsa SOUND komutunun ses şiddeti kısmı notanın başlangıç değeri olarak alınır.

Şimdi komutun kanal numarası kısmını ele alalım. Burayı tam olarak anlamak için binary sayıları biraz bilmek gerekir. Bir ses, eğer bilgisayar stereo amplifikatöre bağlanmışsa üç kanaldan biri üzerinden çalınabilir. Notanın hangi kanal veya kanalların üzerinden çalınacağını seçmek için şu sayılar kullanılır:

1 kanal A
2 kanal B
4 kanal C

Bir kanaldan daha fazla kanalda çalmak istiyorsanız gerekli kanal numaralarını ekleyiniz. Örneğin A ve C'de çalmak için 1 + 4 = 5'i kullanın

```
SOUND 5,284
```

Belki siz niye C kanalı 3 yerine 4 olarak verildi diye merak ediyorsunuz. Bunun sebebi, binary sayı formuna uysun diye her sayının 2'nin üstü olarak teşkil edilmesidir. (1 = 2 ↑ Ø, 2 = 2 ↑ 1, 4 = 2 ↑ 2)

Ondalık 5 sayısının eşdeğeri 1*4 + Ø*2 + 1*1 veya binary olarak 1Ø1'dir. Bu binary sayının her kolonunu C, B, A diye isimlendirirsek:

```
C B A
1 0 1
```

Diğer bir deyişle C çalışıyor, B çalışmıyor, A çalışıyor. Eğer nota kanal A ve B'den çalınıyorsa:

```
C B A
0 1 1
```

Binary 011, 0*4 + 1*4 + 1*1 = 3 ile aynıdır. Böylece SOUND komutu aşağıdaki gibi olur:

```
SOUND 3,142
```

Eğer bunu anlamadıysanız merak etmeyin. Kanal kombinasyonlarının, her birinin sayısını eklemekle elde edildiğini bilmeniz yeterlidir.

Ne yazık ki kanal numarasında kullanılan birkaç değer daha vardır. 8, 16 ve 32 sayıları sesin diğer kanallarla (sırasıyla A, B ve C) randevusunu belirtmek için kullanılır. Siz randevu sözüyle ne demek istendiğini merak edebilirsiniz. Şimdiye kadar ürettiğiniz sesler belirtilen kanaldan doğrudan doğruya gittiler: Şunu deneyiniz:

```
SOUND 1,142,2ØØØ
SOUND 1,9,2ØØ
```

Eğer çok yavaş bir yazıcı değilseniz, bu komutlardan 2.'sini, 1.'sini bitirmeden yazabileceğinize dikkat edeceksiniz. Bunun sebebi, ses sisteminin her kanal için 5 ses komutun bir "kuyruk" içinde tutabilmesidir. Eğer bir sesin kanal A'da çalmasını ve sonra, A ve B'de iki sesin aynı anda çalmasını istiyorsak, A'daki ikinci nota çalmaya hazır olmadan B'deki sesin başlamaması gerektiğini, yani bir kanalın diğerini beklemesini bilgisayara bildirmenin bir yolunu bulmalıyız. İşte bu olay randevu olarak adlandırılır. Gerçekleştirilmesinin 2 yolu vardır:

```
SOUND 1,200,1000
SOUND 3,90,200
```

Burada ikinci nota A ve B'ye yöneltilmiştir. Bu nedenle A'daki bitmeden başlayamaz. Bu metodu (bir nota kombinasyonunu, birleşimini, çalacağı bütün kanallar serbest oluncaya kadar bekletmek) kullanılırken şu konuda sınırlıyız: aynı ses, kanalların herbirine yönelecektir. (örnekte 90 ile 200 A ve B'nin her ikisine de gitmiştir.) Bir diğer metot da aşağıdaki yolu kullanmaktır.

```
SOUND.1,200,2000
SOUND 1+16,90,200
SOUND 2+8,140,400
```

Burada A'daki ikinci nota B'deki sesle "randevu" laştırılmıştır. Gene değerler, binary sistemde anlamı olan tek basamaklı sayılarla ifade edilebilir:

$8 = 2 \uparrow 3$, $16 = 2 \uparrow 4$ ve $32 = 2 \uparrow 5$

Böylece kanal numarası, sütunların aşağıdaki şekilde adlandırıldığı, binary sistemde bir sayı haline getirilebilir:

Randevu C	Randevu B	Randevu A	Çal C	Çal B	Çal A
32 topla	16 topla	8 topla	4 topla	2 topla	1 topla

Bir notayı C'de çalmak ve A ile randevulaştırmak için şunu kullanacaksınız:

Ø Ø 1 1 Ø Ø

Bu binary sistemde 1100 sayısıdır ve $8 + 4 = 12$'dir. Bu sebeple, 12 kanal numarası bilgisayara bir notayı kanal C'de çalmasını ve kanal A'da kendisi ile randevulaştırılmış notayı beklemesini bildirecektir.

Eğer kanal numarasına 64 ($2 \uparrow 6$) katılırsa notanın tutulması gerektiğini gösterir. Gerçekten, RELEASE komutu kullanılmadan nota çalınmayacaktır. Son olarak kanal numarasına 128 ($2 \uparrow R$) katılırsa belirtilen kanalın kuyruğu silinecektir. Bundan dolayı, belirli bir kanalda uzun süre devam edecek bir ses başlatmışsanız, durdurmanız çabuk bir yolu kanalı silmektir.

```
SOUND 1,248,3000          (5 dakika boyunca çalacaktır)
SOUND 1+128,Ø             (bu komut ile kısa zamanda durur)
```

Doğrudan kullanma halinde, uzun sesleri durdurmanın daha çabuk bir yolu (DEL) anahtarına, bir satırın başlangıcında basmaktır. Kısa ikaz 'beep'i tüm ses kanallarını temizler. Artık seçeceğiniz 3 kanaldan herhangi birisine ses gönderebiliriz. Basit SOUND komutunun çıkardığı oldukça melodisiz 'beep' sesinden biraz daha iyisini üretmek herhalde güzel olacaktır. Bunu yapmanın yolu bir zarf ile (çaldığı kısa süre içinde notanın nasıl açılacağını ve kısılacağını gösteren bir modelle) sesi çalmaktır. Bir müzik aletinin çıkardığı nota, ses başlangıçta ses şiddetinin aniden arttığı bir başlama tarzına sahiptir. Ses sonra bir müddet uzatıldığı daha aşağı seviyelere sonra yavaş yavaş sıfıra iner. Notalara SOUND komutuyla elde edilen bu tür zarf vermek mümkündür. ENV komutu bu amaçla kullanılır.

```
ENV 1,5,3,4,5,-3,8
SOUND 1,14,20,0,1
```

ENV komutu birlikte kullanıldığı SOUND komutundan evvel gelir. Bir zarfı SOUND komutunda kullanmak için, numarası SOUND komutunun beşinci bölümününkü gibi alınmalıdır. Bu örnekte 1 sayısı zarfın numarasıdır. ENV komutundaki ilk sayı, tarif ettiği zarfın numarasıdır. ENV komutunun eğitici yönü, notanın ne kadar süreceği, ne kadar ses şiddetinde olacağı hakkında bilgiler içermesidir. Bu yüzden SOUND komutunun süreklilik ve ses şiddet kısımları 0 yapılmıştır. Yukarıda anlatılan zarf, sesin 5 adımda artmasına neden olur. Her adım ses şiddetini 3 ile arttırır ve her adım saniyenin 4 yüzde biri uzunluğundadır. Sonra ses şiddeti 5 adımda azaltılacaktır her adım saniyenin 8 yüzde biri sürecektir. Başka bir deyişle, ilk sayı zarfın ifade edildiği numarayı belirtir. Bu sayıyı 3'er sayıdan oluşan 2 grup takip eder. Bu grupların her birinde ilk sayı ses şiddetinin kaç adımla artıp azalacağını gösterir. 2. sayı bu adımların herbirinde ses şiddetinin ne kadar artıp azalacağını belirtir. 3. sayı ses şiddetinin her adımının ne kadar süreceğini belirtir.

Her bölümün süreceği toplam süre ilk değerin (adım sayısı) 3. değer ile (durma süresi) çarpımına eşittir. Ses şiddetindeki toplam artma veya azalma, adım sayısının adım büyüklüğü ile çarpımıdır. Çok bölümlü zarfın toplam uzunluğu ise bölümlerin herbirinin uzunluklarının taplamına eşittir.

Başlangıçda ses şiddeti her zaman 0 olmak zorunda değildir. (SOUND komutu ile yapıldığı gibi.) Yukarıdaki örnek yükselen ve sonra azalan bir işaret üretmiştir. Aşağıdaki ise önce azalan sonra artan bir örnektir:

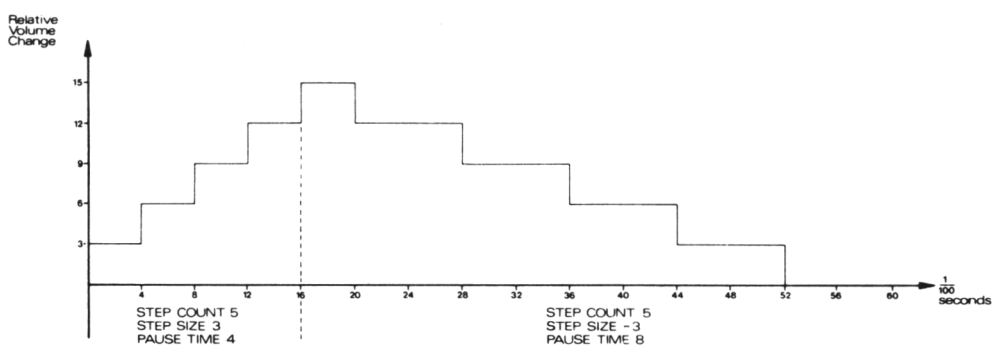
```
ENV 2,5,-2,1,20,0,1,10,1,1
SOUND 1,248,0,15,2
```

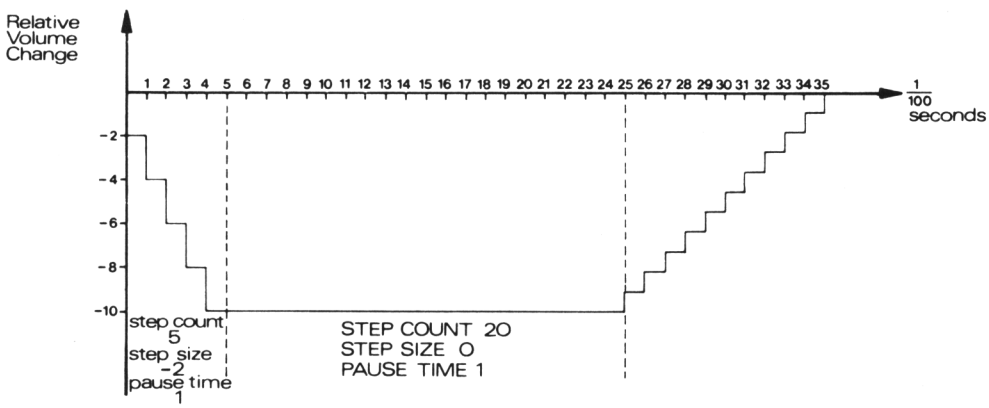
Bu zarfa 2 numarası verilmiştir ve 3 bölümü vardır. Birincide ses şiddeti (-2)'lik 5 adımda azaltılmıştır. Yani her adımda 2 ölçü azalacaktır. Her adımın uzunluğu

saniyenin yüzde 1'idir. 2. bölüm 20 adım içerir. Ama ses şiddetindeki her adım için 0 azalma veya artma vardır. (sabit ses şiddeti.) Yine her adımın uzunluğu saniyenin yüzde 1'idir. Son olarak, 3. bölüm 10 adımdan oluşmuştur. Her biri ses şiddetini 1 arttırır. Buradada adımlar saniyenin yüzde 1'i uzunluğundadır.

SOUND komutu başlangıç ses şiddeti olarak 15 değerine sahiptir. Böylece ilk bölümden sonra ses şiddeti 5'e düşecek ve bu durum saniyenin 20 yüzde birlik müddeti boyunca sabit tutulacaktır. Sonra zarfın son bölümünde 15'e yükseltilecektir.

Zarfların şekillerini zihnimizde canlandırmak belki biraz zor gelebilir. Fakat böylece onları grafik kağıdına çizebilir ve ENV komutu için değerleri hesaplayabiliriz. Aşağıda şimdiye kadar anlatılan iki zarfın şekilleri gösterilmiştir:

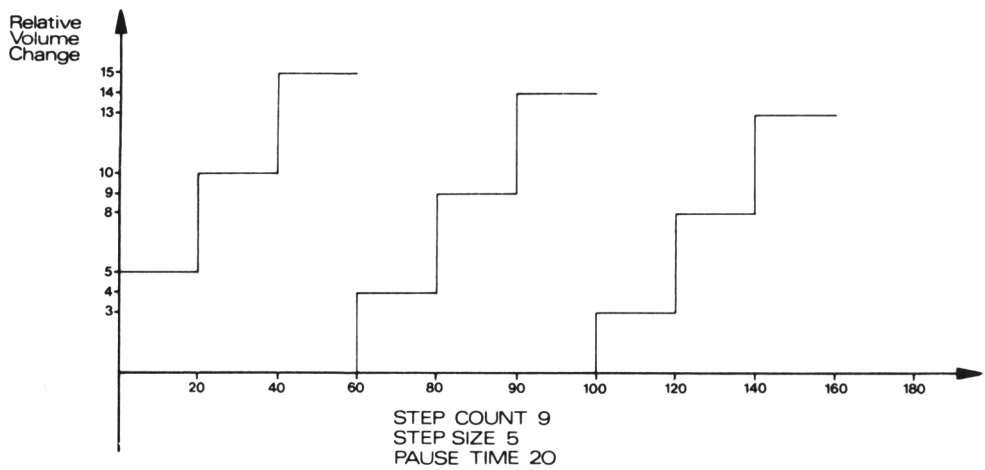




Bir zarftaki maksimum bölüm sayısı 5'tir ve her bölüm 3 değer alabilir. Böylece ENV komutu en fazla 16 bölüme sahip olabilir. (Birinci, yani 1) zarftan (1'den 15'e) hangisinin oluşturulduğunu gösteren dahil.) Eğer aşağı veya yukarı doğru adımlar ses şiddetinin 15'den yukarı çıkmasına veya 0'dan aşağı inmesine neden olursa 15'in üstündeki adım 0 ve 0'ın altındaki adım 15 alınır.

```
ENV 3,9,5,2Ø
SOUND 1,142,Ø,Ø,3
```

Bu basit zarf 9 adım üretecektir. Her adım ses şiddetini 5 ile arttırır ve her adım saniyenin 20 yüzde birlik süresi boyunca devam eder. Böylece ilk 3 adım sonunda ses şiddeti 15 olacaktır. Bu nedenle bir sonra ki adım, değeri 4. sonra 9 yapacak ve bu şekilde devam edecektir. Yan sayfadaki diyagram bu durumu göstermektedir:



Adım sayısını gösteren değerler 0'dan 127'ye kadar olabilir. Adım büyüklüğü -128'den + 127'ye kadar olan değerleri alabilir. (Negatif değerler ses şiddetini azaltır.) İki adım arasında durma süresi 0 ile 255 arasında olabilir.

Artık kendine özgü bir şekil verecek bir ses şiddeti zarfı yapabiliriz. Aynı zamanda vibrato isimli müzik aletinde olduğu gibi, notaların bir ana ton etrafında "titrediği" tipte şeyler içeren kendine has bir ton modeli tarif etmek isteyebiliriz.

Bu, ses şiddeti zarflarının tarif edildiği yola çok yakın bir yolla gerçekleştirilebilir. Ton zarfları ENT komutu kullanılarak tarif edilir. Örneğin:

```
ENT 1,5,1,1,5,-1,1
SOUND 1,142,10,15,,1
```

Bir ton zarfı SOUND komutunun içine, ton zarfının numarası SOUND komutunun 6. kısmına yazılarak, çağrılır. Burada da ENT komutu SOUND komutundan önce gelmelidir.

ENT ile ilgili ilk örnek şunları göstermektedir: Ton zarfının numarası 1'dir ve 5 adım vardır. Her adım ton değeri 1 ile artacaktır ve her adım saniyenin yüzde 1'idir. Zarfın diğer bölümü de 5 adıma sahiptir, ton değeri -1 ile azalır ve her adım saniyenin yüzde 1'idir. Böylece toplam uzunluk 5 + 5 = 10 olur. Bu uzunluğu SOUND komutunda belirtildiğine dikkat ediniz. Çünkü bir ton zarfı notanın çalacağı süreyi belirlemez. (Ses şiddeti zarfının aksine.) Eğer SOUND komutunda kullanılan süre ton zarfının süresinden kısaysa, ton zarfının son kısmı yok olacaktır. Eğer büyükse, notanın son kısmı sabit bir tonla çalacaktır. Bu, ses şiddeti zarfının nota uzunluğunu belirlemek için kullanılmasında da uygulanır. (SOUND komutunun ses şiddeti zarfı kısmında bir sayının yokluğuna dikkat ediniz. Bunun sebebi bu ses için bir ses şiddeti zarfı oluşturmamamızdır.) Ton zarflarının çoğunluğu, muhtemelen, beklenen nota uzunluğundan kısa olacaktır. Bu nedenle, bir ton zarfı müzik çaldığı sürece tekrarlamak üzere yapılabilir. Pozitif değerlisi SOUND komutunda kullanılması şart olan bir negatif zarf numarası tayin edilerek, bu sağlanabilir:

```
ENT -5,4,1,1,4,-1,1
SOUND 1,142,100,12,,5
```

Bu, müzik üzerinde bir vibrato etkisi yapacaktır. Ton zarflarını tanımlarken 1.ton periyodu etrafında simetrik değişmeler sağlanabiliyorsa genellikle iyidir. Böylece nota tekrarlandığında, başlangıçtaki frekanstan (en alt ve en üst noktaları) uzaklarda kesilmeyecektir.

```
ENT -6,3,1,1,
SOUND 1,142,90,12,,6
```

Bu tonun frekansının oldukça çarpıcı olarak azalacağına dikkat ediniz. Zarfın her tekrarında, ton süresi 3 ile artacak ve 30 kere (90/3) olacaktır. Bu tür etki 'kuş şakıması' türünde sesler ve sirenler için yararlıdır.

```
ENT -7,20,1,1,20,-1,1
SOUND 1,100,400,12,,7

 END -8,60,-1,1,60,1,1
 SOund 1,100,480,12,,8
```

Mümkün olan en fazla ton zarfı sayısı 15'tir. (1'den 15'e kadar) Negatif değer alması zarfın tekrarlanması gerektiğini gösterir. Adım sayısı (her 3'lük gruptaki ilk değer) 0 ile 239 arasında olabilir. Ses şiddeti zarfında olduğu gibi adım büyüklüğü -128 ile 127 arasındadır. Durma süresi de 0 ile 255 arasında olabilir. Yine, ENV komutunda olduğu gibi, ENT komutu da herbiri 3 değerli en fazla beş bölüme sahiptir.

Bir SOUND komutuna eklenebilecek son kısım 7. değerdir. Bu değer ses içinde kapsanacak gürültü seviyesidir. Sesin içinde bu tür gürültülere müsaade ederken akla takılabilecek nokta, sadece bir tek gürültü kanalı olması gerçeğidir. Böylece her müteakip gürültü tekrarında gürültü değeri bir öncekine eklenecektir. Gürültüler tona değişik bir ses vermesi için eklenebilir veya tamamen tek başlarına kullanılabilirler. SOUND komutunun ton süresi (2. kısmı) 0 yapılarak, sadece gürültülerin kalması sağlanabilir. Bu işlem, vurma tipi sesler üretmekte kullanışlıdır. Örneği deneyiniz:

```
ENT -3,2,1,1,2,-1,1
ENV 9,15,1,1,15,-1,1
FOR a=1 TO 10: SOUND 1,4000,0,0,9,3,15: NEXT
```

Dikkat ederseniz bu örnek zarfların ve gürültülerin her iki tipini de birleştirmiştir. SOUND komutunun durma ve ses şiddeti kısımları, ses şiddeti zarfında istendiği gibi, O'da ayarlanmıştır. Artık SOUND ENV ve ENT komutlarını tam anlamıyla kullanabiliriz. Konuyla ilgili diğer komut ve fonksiyonlara şimdi göz atabiliriz. Hatırlayacaksınız, SOUND komutunun kanal numarasını anlatırken bu sayıya 64 ekleyerek sesi, serbest bırakılıncaya kadar çalmadan bir dizide bekletebiliyorduk. Bir sesi serbest bırakmanın yolu RELEASE komutunu kullanmaktır. RELEASE komutunu binary sistemde değeri olan bir sayı takip eder. Bu sayının her rakamı serbest bırakılabilecek 3 kanaldan birini temsil eder. Sadece şu aşağıdakilerini anlamışsanız, konuyu tamamen kavramak gereksizdir.

4 kanal C demektir
2 kanal B demektir
1 kanal A demektir

Ve bir kanal kombinasyonu, her kanal için olan değerlerin toplanmasıyla serbest bırakılabilir. Yani 3 kanalın tümünde tutulmuş sesleri serbest bırakmak için aşağıdaki komut kullanılmalıdır:

```
RELEASE 7
```

Şu örneği deneyiniz:

```
SOUND 1+64,90
SOUND 2+64,140
SOUND 4+64,215
RELEASE 3:FOR t=1 TO 1000: NEXT: RELEASE 4
```

Kanal A ve B'deki sesleri, çalması için serbest bırakan ilk RELEASE komutuna kadar SOUND komutlarından beklediğiniz sesler üretilmeyecektir. Bir müddet gecikmeden sonra kanal C'deki ses serbest kalacaktır.

Birden çok sesin randevulaştırıldığı bir başka metot daha vardır. Tutulduğunu belirten (64 eklenmiş) numaralı kuyruğa bir ses daha eklendiğinde yalnız tutulan ses değil bütün müteakip sesler de aynı kuyruğa yollanacaktır. Tutulmuş kuyruğa 4 fazla ses daha gönderilirse, o zaman alet, kuyruk serbest bırakılıncaya kadar duracaktır. Bu iş, belirli bir zaman aralığı sonrası çağrılan bir alt-program kullanılarak sağlanabilir. (AFTER veya EVERY kullanarak.) Ses sistemini kullanırken bu metot iyi bir yol değildir. Çünkü ses kuyruklarının tamamen dolması halinde olduğu gibi programın içerdiği komutlar zaman zaman durabilir. Bu durum birçok uzun sesin hızlı bir sıra ile kuyruğa eklenmesi halinde de geçerlidir.

```
10 FOR a=1 TO 8
20 SOUND 1,100*a,200
30 NEXT
40 PRINT "merhaba"
```

'merhaba' kelimesinin hemen gözükmeyeceğine dikkat ediniz. Bunun nedeni kuyrukta serbest bir boşluk oluncaya kadar program akışının devam edememesidir.

AFTER ve EVERY komutlarında ve ON BREAK GOSUB'ta kullanıldığı gibi, BASIC ara verilebilir bir mekanizmaya sahiptir. Bu size gerekli kuyrukta bir serbet boşluk oluştuğu zaman çağrılabilen alt-programla çalan bir ses tayin etme imkanı sağlar. örneği deneyiniz:

```
10 a=0
20 ON SQ(1) GOSUB 1000
30 PRINT a;
40 GOTO 30
1000 a=a+10
1010 SOUND 1,a,200
1020 IF a 200 THEN ON SQ(1) GOSUB 1000
1030 RETURN
run
```

SOUND komutu sadece kanal A'nın kuyruğu (1 numara) serbest bir boşluğa sahip olduğu zaman çağrılabilir. Bu durum 20. satırdaki komutla meydana çıkarılmıştır. Bu komut tayin edilmiş kuyrukta serbest bir boşluk oluştuğunda ses alt-programını çağıran ara verilebilir mekanizmayı başlangıç değerine getirir. 1020. satırdaki alt program ile ON SQ GOSUB bir kere kullanıldıktan sonra tekrar başlangıç durumuna getirilmiştir. Bu örnekte (a)'nın değeri 200'den küçük olduğunda ses alt-programı kendini tekrar başlangıç durumuna getirmektedir.

Ekranda nesnelerin yerini değiştiren, toplama yapan ve benzer işlemler yapan tamamlanmış bir programda her notayı sadece kuyrukta bir serbest boşluk olduğu

zaman çağıran bir alt-program sayesinde arka planda devamlı çalan bir müzik elde edilebilir. Bu sayede program serbest bir boşluk gözükmesini beklerken durmaması sağlanır. Bu müzik için nota değerleri DATA ifadelerinden okunmuşsa ses alt-programının kendini devamlı başlangıç haline getirmesi durduralabilir.

O SQ ( ) GOSUB komutundaki parantezin içindeki sayı, serbest bir boşluk için denenecek kanal kuyruğuna bağlı olarak 1,2 veya 4 olabilir.

Ses kanallarından herhangi birinin akış durumunu okumak için bir programla birlikte kullanılabilen bir SQ ( ) fonksiyonu daha vardır. Parantezin içindeki sayı yine 1,2 veya 4 olabilir ve bilginin hangi kanala geri gönderileceğini belirler. Fonksiyon binnary sayı sistemine çevrilebilen tek basamaklı bir sayı gönderir. Bu sayıya BİT denir. Bunu anlamak için, bu sistemdeki sayıları biraz bilmek gerekmektedir. Geri gönderilen bu sayıların değerleri aşağıdaki anlamdadırlar:

BİT	ONDALIK DEĞERİ	ANLAMI
0,1,2	1,2,4	Kuyruktaki serbest boşlukların sayısı
3	8	Kuyruğun başındaki nota, A ile randevulaşacaktır
4	16	Kuyruğun başındaki nota, B ile randevulaşacaktır
5	32	Kuyruğun başındaki nota, C ile randevulaşacaktır
6	64	Kuyruktaki en yüksek nota tutulmuş bit setine sahiptir
7	128	Bir nota devamlı çalmaktadır.

Aşağıdaki basit örneği deneyiniz:

```
10 SOUND 2,200
20 X = SQ(2)
30 PRINT BIN$(X)
run
```

Örnek, binary sistemdeki 10000100 sayısını yazacaktır. İçindeki 7 'bit'i, SQ fonksiyonu kullanıldığında kanalın devamlı çalacağını belirtir. Son 3 basamak 100, ondalık düzene çevrildiğinde 4 değerini verecektir. Yani kuyrukta 4 boş giriş olduğunu belirtecektir. Bu fonksiyon bir program içinde verilen bir noktada kuyruğun durumunu incelemek için kullanılabilir. ON SQ ( ) GOSUB komutunda komutunda ise belirli olmayan bir noktada kuyruğun durumu incelenecektir.

Şimdiye kadar tüm örnekler sesin bir veya iki notası ile ilgiliydi. Bağımsız notalardan oluşan bir bütün grup oluşturmak, örneğin bir müzik içinde, gerekli notaları bir DATA ifadesi ile yükleme sonucunda elde edilebilir. Böylece bu notalar DATA komutundan, SOUND komutu ile okunabilir.

```
10 FOR octave = -1 TO 2
20 FOR x = 1 TO 7: REM notes per octave
30 READ note
40 SOUND 1, note/2 octave
50 NEXT
60 RESTORE
70 NEXT
80 DATA 426,379,358,319,234,253,239
run
```

Bu bölümdeki en son örnekte, beste ve ritim A ve B kanallarında sırayla randevulaşarak çalmaktadır. Bu örnek DATA'nın notayı, oktavı, uzunluğu ve randevu bilgilerini içerecek şekilde oluşturulmasının yollarından birini açıklamaktadır:

```
10 REM line 190 gives treble clef tune
20 REM line 200 gives bass clef tune
30 DIM scale%(12):FOR x%=1 TO 12:READ scale%(x%):NEXT
40 ch1%=1:READ ch1$:ch2%=1:READ ch2$
50 CLS
60 Spd%=12
70 scale$=" a-b b c+c d-e e f+f g+g"
80 ENV 1,2,5,2,8,-1,10,10,0,15
90 ENV 2,2,7,2,12,-1,10,10,0,15
100 ENT -1,1,1,1,2,-1,1,1,1,1
110 DEF FNM$(s$,s)=MID$(s$,s,1)
120 ch1%=1:GOSUB 200
130 ch2%=1:GOSUB 380
140 IF ch1%+ch2%>0 THEN 140
150 END
160 DATA &777,&70c,&6a7,&647,&5ed,&598
170 DATA &547,&4fc,&4b4,&470,&431,&3f4
180 DATA 4cr4f4f1f1g1A1-B2C2f4g2g1A1-B6A2Cr1f1g1f1g1a1-
    b1A1-b2C2g2A2g2f1g1a2g2f6e2c2e2c2g2e2c1-B1A2g2f4e4d
    8c4f3f1c2d4-b2fr2-B2A2g2f6e2gr4C4-B1a1f1-b1g2c2-b4a
    4g4fr6A2A2-B4-B2Ar2-B2A2g2f6e2g4C4-B1A1f1-B1g2C2-B4
    A4g8f.
```

```
190 DATA r4f4f8f4e4c4fr8f4e2f2e4d2e2d8c8c6e2f4g4g8e4f3f
    1c4dr8g4cr4e4c6f2d4c4c8fr8-e4dr8g8c4e4c6f2d4c4c8f.
200 REM send sound to channel A
210 p1$=FNM$(ch1$,ch1%)
220 IF p1$<>"r" THEN r1%=0:GOTO 240
230 r1%=16:ch1%=ch1%+1:p1$=FNM$(ch1$,ch1%)

240 IF p1$="." THEN ch1%=0:RETURN ELSE l1%=VAL(p1$)
250 ch1%=ch1%+1
260 n1$=FNM$(ch1$,ch1%)
270 ch1%=ch1%+1
280 IF n1$="+" OR n1$="-" THEN 350
290 n1$=" "+n1$
300 nd1%=(1+INSTR(scale$,LOWER$(n1$)))/2
310 IF ASC(RIGHT$(n1$,1))>96 THEN o1%=8 ELSE o1%=16
320 SOUND 1+r1%,scale%(nd1%)/o1%,Spd%*l1%,0,1,1
330 ON SQ(1) GOSUB 200
340 RETURN
350 n1$=n1$+FNM$(ch1$,ch1%)
360 ch1%=ch1%+1
370 GOTO 300
380 REM send sound to channel B
390 p2$=FNM$(ch2$,ch2%)
400 IF p2$<>"r" THEN r2%=0:GOTO 420
410 r2%=8:ch2%=ch2%+1:p2$=FNm$(ch2$,ch2%)
420 IF p2$="." THEN ch2%=0:RETURN ELSE l2%=VAL(p2$)
430 ch2%=ch2%+1
440 n2$=FNM$(ch2$,ch2%)
450 ch2%=ch2%+1
460 IF n2$="+" OR n2$="-" THEN 530
470 n2$=" "+n2$
480 nd2%=(1+INSTR(scale$,LOWER$(n2$)))/2
490 IF ASC(RIGHT$(n2$,1))>96 THEN o2%=4 ELSE o2%=8
500 SOUND 2+r2%,scale%(nd2%)/o2%,Spd%*l2%,0,2
510 ON SQ(2) GOSUB 380
520 RETURN
530 n2$=n2$+FNM$(ch2$,ch2%)
540 ch2%=ch2%+1
550 GOTO 480
run
```

Grafiklerle Konuşmak....

Bu bölüm kullanışlı grafik imkanlarını açıklamaktadır. İlk örnek önemli özellikleri sırayla açıklayacak şekilde hazırlanmıştır.

Başlarken, MODE'ları ve yol boyunca çakan renkleri kullanarak (20. satır), ekranı bir metin penceresine (40. satır) ve bir grafik penceresine (30. satır) çevireceğiz:

```
10 REM mask and tag in window
20 MODE 1:INK 2,10,4:INK 3,4,10
30 ORIGIN 440,100,440,640,100,300
40 WINDOW 1,26,1,25
50 CLG 2
```

Bu programı çalıştırdığınızda, ekranın sağ tarafının ortalarında kare şeklinde, parıldayan bir renk göreceksiniz. Bu kare, 50. satırdaki 2 numaralı renkle renklendirilmiştir. Koordinatların orijini karenin sol alt köşesine kaymıştır. Mode komutu grafik kursörünü koordinaz eksenlerinin orijinine yerleştirilmiştir. (X = 0, Y = 0) Böylece 60. satır ile karede diyagonal bir çizgi çizebiliriz:

```
60 DRAW 200,200,3
```

Sonucu görmek için programı çalıştırıp şu satırı ekleyiniz:

```
80 MOVE 0,2:FILL 3
```

80. satır grafik kursörünü karenin iki yarısından bir tanesinin içine koyar ve INK 3 ile doldurur. Doldurmanın sınırı, grafik penceresinin kenarıdır (burada aynı zamanda karenin kenarı). Grafik kalemiyle (3) çizilen herhangi birşey veya mürekkeple çizilmiş herhangi birşey doldurulabilir (yine 3).

Artık son programı çalıştırınız.

'doldurma' köşeleri hakkındaki noktayı aydınlatmak için aşağıdaki 70. satırı ekleyin. Dolduran miktarı, onu karenin yarısı ile sınırlayan diyagonal çizgi ile aynı renkte olduğunu göreceksiniz.

```
70 GRAPHICS PEN 1
run
```

80. satırı 1 nolu renk ile doldurmak (FILL) üzere düzenleyin. Sonra bu son noktayı görmek için çalıştırıp tekrar eski haline getirin (FILL 3).

Programa aşağadaki satırları ekleyip çalıştırdığınızda bir kutunun çizildiğini göreceksiniz:

```
100 MOVE 20,20
110 DRAW 180,20
120 DRAW 180,180
130 DRAW 20,180
140 DRAW 20,20
run
```

70. satır nedeniyle kutu 1 nolu renkle, renklendirilecektir. 70. satırı kullanmazsak, o zaman bir (,1) eklemeliyiz. Bu 100. satırdaki MOVE komutunun 3. parametresi veya 110. satırdaki DRAW komutunun 3. parametresi olacaktır.

Noktaları Birleştirin....

Çizgileri kesintisiz olmayabilir, nokta nokta da olabilir. MASK komutu noktaların boyutunu belirlememizi sağlar. Model, her 8 pikselde tekrar edecektir. Her müteakip satır, son satırın bittiği yerden noktalamaya devam edecektir. Yeni bir MASK komutu (ihtimalen geçerli olan ile aynı parametrede) noktalama mantığını 8 piksellik modelin başına getirecektir.

Gerçekten noktalama modeli, tek bayt'lık bir binary sayıdır. Bit seti mürekkebin nereye konulacağını gösterir. Örneğimizde, binary bir sabit kullanacağız (&×'ile çağırılır). Bu, her 8 piksellik grubun ortasında 4 pikselin çizilmesini, diğer iki tarafının çizilmemesini belirtmektedir. Bunun sonucunda 4 pikselin açık, 4'ünün kapalı olduğu kesik kesik bir çizgi oluşacaktır. Elde etmek için şunu ekleyiniz:

```
90 MASK & X00111100
run
```

Ama bir dakika bekleyiniz!. Bu program, beklediğimiz gibi köşeler etrafında düzgün devamlı bir noktalama vermez. Bunun sebebi her köşe noktasının gerçekten, biri, bir satırın son pikselinde, ikincisi bir sonraki satırın ilk pikselinde olmak üzere, iki defa işaretlenmesidir.

Aşağıdaki yol, iyi bir yöntem değildir:

```
115 MOVE 180,22
125 MOVE 178,180
135 MOVE 20,178
run
```

Program istenilen sonucu verecektir. Ama daha kolay bir yol vardır: Bilgisayara her satırın ilk noktasını işaretlememesini söyleyen MAKS komutunun sonuna ikinci bir parametre eklemek. 90. satırı aşağıdaki gibi düzenleyin:

```
90 MASK & X 00111100,0
```

ve yeni eklediğiniz satırları siliniz:

```
115
125
135
```

Şimdi programı çalıştırınız. Bir kere daha noktalanmış kutu simetrik'tir. Eğer 2. MASK parametresi (,1) ise, komutun ilk piksel de dahil olmak üzere bütün satırları çizmek üzere uyarlanacağına dikkat ediniz.

Şimdi çizginin tireleri arasına göz gezdiriniz. Sol üst üçgende olmayan ama sağ alt üçgende olan birşey vardır. Bu, 50. satırdaki CLG 2 komutuyla 2 nolu renge ayarlanan grafik kağıdıdır. Sol üst üçgende görünmez, çünkü arkadaki fonun rengindedir. 50. satırı şu şekilde değiştirin:

```
50 CLG 2:GRAPHICS PAPER 0
```

Programı tekrar çalıştırın. Paper şimdi kutunun her yerinde açıkça görülecektir. Grafik kağıdını görülebilir yapmak mümkündür, biz bunu 'saydamlık' diye isimlendiriyoruz. Daha açıkcası, varolan resmin üzerine çizilmiş noktalı bir çizgi noktalar arasındaki boşlukları koruyacaktır. GRAPHICS PEN komutuna (,1) parametresi ekliyerek grafik çizimi saydam yapılabilir (tekrar saydam olmayan hale (,0) parametresi ile çevrilebilir.) 70. satırı şöyle değiştiriniz:

```
70 GRAPHICS PAPER 1,1
run
```

ve sonucu izleyiniz.

Çizgi çizmek (nokta işaretlemek) gibi, grafik kursörünün pozisyonundaki normal metin karakterlerini yazmak da mümkündür. Bunun faydalı yönleri büyük bir doğrulukla (1 pikselde, 8 pikselde olduğundan daha fazla) karakterleri yerleştirebilmemiz ve grafik renk mod'larından istediğimizle karakterleri çizebilmemizdir. (İleri konulara bakınız.)

Grafik kursörünün bulunduğu yerlere karakterleri yazmak için, grafik kursörünü karakterin çizilmesini istediğiniz yerin sol üst köşesine getiriniz. Sonra normal yazıcı komutlarla takip edilen TAG komutunu kullanınız. Grafik kursörü otomatik olarak, her karakter çizildikten sonra 8 piksel sağa kayacaktır. Şunları ekleyiniz:

```
160 MOVE 64,108
170 TAG
180 PRINT "OSMAN"
190 TAGOFF
run
```

(BASIC ile verilen herhangi bir mesaj, TAG/TAGOFF düğmesinin durumuna bakmaksızın metin ekranına gönderilecektir. TAG atamasını, peşisıra gitmenin bittiği yerde iptal etmek iyi bir uygulamadır.)

İsimden sonra gelenler nelerdir? Bunlar geri getirme vasıtası sembolü CHR$ (13) ve satır besleme CHR$ (10) sembolleridir. Grafik çizimi programları, ilk 32 ASCII karakterinin hepsini yazabilir, uyarlamalara çevirir. (Eğer metin ekranına gönderildiklerinde, önlerinde CHR$(1) varsa.) Bunun sebebi ilk 32 kontrol işaretinin hemen hepsinin sadece metin ekranınca anlaşılabilir olmasıdır. Aynı özellik nedeniyle, örneğin, eğer grafik penceresinin sağ tarafına yığılım yaparsak, hiçbir katlanma oluşmayacaktır.

Geri getirme vasıtası ve satır besleme sembolleri, PRINT ifadesini bir noktalı virgül ile bitirme tekniği ile gizli tutulacaktır.

```
180 PRINT "OSMAN"
run
```

TAG kullanılarak grafik ekranına gönderilen metin, çizgi çizme gibi, aynı GRAPHICS PEN komutlarının etkisindedir. Bu nedenle devamlı olarak, isim GRAPHICS PEN 1'in içinde yazılır ve saydamdır. Komut:

```
150 GRAPHICS PEN 1,0
run
```

şeffaf olmayan kağıda çıkaracaktır.

```
150 GRAPHICS PEN 0,1
run
```

iken ise, 0 nolu renk ile saydam olarak yazacaktır. Şimdi 150. satırı silin ve tekrar çalıştırın. Renk numarası 1 + saydam mod (daha önce 70. satırda konulmuştu) grafik kalemine tekrar yerleştirilecektir.

Saydam Karakterler

Karakterleri metin ekranına saydam olarak yazmakta mümkündür. İçerilen kontrol birimlerinden bir tanesi kullanılır. Şunları ekleyiniz.

```
200 PRINT 2,CHR$(22);CHR$(1)
210 LOCATE 2,32,14:PRINT 2,"*******"
220 LOCATE 2,32,14:PRINT 2,"-----"
230 LOCATE 2,CHR$(22);CHR$(0)
run
```

200. satır akıntı 2'yi saydam mode'a ayarlar. Yıldız işaretlerini "bastırmak" için satır altı tireleri"nin oluştuğuna dikkat ediniz. Böylece karşılık karakterler, hatta çok renkli bile, üretilebilmenin mümkün olduğu anlaşılır. 230. satır saydam mode'i bitirir. Akıntı 2'yi tekrar saydam olmayan mode'a ayarlar.

Renk Modları

Ekranın üzerindeki renk ile istenilen boyamayı birleştiren bir grafik renk modu kullanarak boyamak mümkündür. İşaretlenen grafik renkli (kalem ve kağıt) pixel için, eski rengin mantıklı bir birleşim oluşturarak her pixel için son renk üretilir. İçerilen mantıklı birleşimler XOR, AND ve OR'dur. Renk modu DRAW/DRAWR, PLOT/PLOTR ve MOVE/MOVER komutlarının 4. parametresi olarak ayarlanabildiği gibi, bir CHR$(23) yani CHR$(modes) kontrol birimi dizisi yazarak (PRINT)da elde edilir. Her örnekte, 1 değeri XOR'u işaretlemeye, 2 değeri AND'ı işaretlemeye, 3 değeri OR'u işaretlemeye ayarlar. 0 değeri grafikrenginin olduğu gibi kullanıldığı "Hakim olan" moda ayarlar. Bir sonraki örnek XOR birleşimlerinin kullanımını göstermektedir. OR genellikle kaplumbağa grafikleri olarak adlandırılan grafiklerde kullanılır. Bunun sebebi, aynı modeli iki defa çizerek orijinal şekli koruyabilme özelliğine sahip olmasıdır. Bu nedenle kare çizimi programı iki defa icra edilmiştir. (110 ve 130. satırlarda) etiketlenmiş (TAG) yazmada iki defa icra edilmiştir. (170 ve 190. satırlarda) FRAME komutları, etkinin daha belli olması için yeterli bekleme sağlanır. 90. satırda komutların ilk parametresiz kullanımına dikkat ediniz. Bu komutların tamamında geçerlidir ve basit olarak, ilk parametrenin son ayarlamasına değişmeden bırakır. 220. satırdaki MOVE komutunun 3. parametresi (,1) GRAPHICS PEN'i 1'e ayarlar. (60.satırdaki ayarlanan 3'ü geçersiz yaparak.) XOR modu, 230. satırdaki DRAWR komutunun 4. parametresi olarak ayarlanmıştır. Bir kere daha kullanılmayan parametreye dikkat ediniz. İlk noktayı işaretleme etkisini hatırlatan bir şey, satır 90'daki MASK komutunu kaldırma sonucu görülebilir. Karenin köşeleri silinir, çünkü iki defa çizilmişlerdir. (Bir çizginin sonunda ve diğer çizginin başında) ve bu yüzden XOR sonucunda olmuşlardır.

```
10 REM XOR ink modes
20 MODE 1:INK 2,10:INK 3,4
30 ORIGIN 440,100,440,640,100,300
40 WINDOW 1,26,1,25
50 CLG 2:GRAPHICS PAPER 0
60 DRAW 200,200,3
70 MOVE 2,0:FILL 3
80 ORIGIN 400,0,440,640,0,400
90 GRAPHICS PEN ,1:MASK ,0
100 FOR Y=60 TO 318 STEP 2
110 GOSUB 220
120 FRAME:FRAME
130 GOSUB 220
140 NEXT
150 TAG
160 FOR y=60 TO 318 STEP 2
170 MOVE 96,Y:PRINT CHR$(244);
180 FRAME:FRAME
190 MOVE 96,y:PRINT CHR$(244);
200 NEXT
210 END
220 MOVE 90,y,1
230 DRAWR 20,0,,1
240 DRAWR 0,20
250 DRAWR -20,0
260 DRAWR 0,-20
270 RETURN
run
```

Canlılık

Çeşitli mürekkepli renkleri devreye sokmakla bir canlılık etkisi üretmek olasıdır. Ekran hafızasının içerikleri değişmemesine rağmen, bir hareket varmış gibi görünür. Böyle bir örnek sisteminizin diskler paketinin 4. kısmındaki 'Welcome' programında gösterilmiştir. (Bu etkiyi görmek için RUN "disc" yazınız.) Eğer canlı modellerin üst üste gelmesi isteniyorsa, bu örneğin basit palet açılımı yetersiz olacaktır. Bir sonraki örnek, ekrana 1'den 4'e kadar numaralı yazmak için çeşitli mürekkepleri OR'lama işlemini kullanmaktadır. (Şekil, sol alt köşeden yazılmış karakteri iyice inceleyerek kararlaştırmıştır ve ne bulursa büyük blok grafiklerle gösterecektir.) Bu örnekle bir kontrol karakter dizi ile, OR modunun açılmasıyla, rakamlar sırasıyla 1,2,4 ve 8 mürekkepleri kullanarak yapacaklardır.

100'den ilerdeki satırlar, gösterildiği bir anda, bir blok grafik numarasıyla sonuçlanan bir matematiksel formüle bağlı olarak paleti çevirmektedir. Mürekkeple her

biri sırasıyla incelenerek ve istediğimiz ikilik unsuru içerip içermedikleri anlaşılarak hazırlanırlar. Örneğin 3 sayısı 4 nolu mürekkeple çizilmiştir. Bu nedenle 3 sayısını göstermek için, ikilik bir 4 içeren tüm mürekkeplerden görünür bir renk tayin etmeliyiz. Bu mürekkepler

4(0100),5(0101),6(0110),7(0111,12(1100),13(1101),15(1111)

Bu pratik uygulamada, canlandırma sırasında her adımda değiştirilmeleri gereken mürekkepler hesaplanmalıdır ve 180'den 200'e kadar olan satırlar daha başarılı bir program bölümüyle değiştirilmelidir.

```
10 REM latch animation
20 ON BREAK GOSUB 220
30 FOR i=1 TO 15:INK i,26:NEXT
40 m(1)=1:m(2)=2:m(3)=4:m(4)=8
50 MODE 0:PRINT CHR$(23);CHR$(3);:TAG
60 FOR P=1 TO 4
70 GRAPHICS PEN m(p),1
80 LOCATE #1,1,25:PRINT#1,CHR$(48+p);
90 FOR x=0 TO 7
100 FOR y= 0 TO 14 STEP 2
110 IF TEST(x*4,y)=0 THEN 140
120 MOVE (x+6)*32,(y+6)*16:PRINT CHR$(143);
130 MOVE (x+6)*32,(y+7)*16:PRINT CHR$(143);
140 NEXT y,x,p
150 LOCATE #1,1,25:PRINT#1," ";
160 FOR p=1 TO 4
170 FOR i= 1 TO 25:FRAME:NEXT
180 FOR i=0 TO 15
190 IF (i AND m(p))=0 THEN INK i,0 ELSE INK i,26
200 NEXT i,p
210 GOTO 160
220 INK 1,26
run
```

Hareket Eden Renkli Düzlemler

Yukarıdaki örnekte, grafikleri 1,2,4 ve 8 mürekkepleri ile yazılarak canlılık etkisinin renk değiştirme ile nasıl sağlandığını gördük. Eğer aynı mürekkepler kullanılırsa, fakat renkler değişik bir yolla hazırlansa da tamamen değişik bir etki ortaya çıkacaktı. Bu etki 'Renk Düzlemleri' olarak adlandırılır. Aşağıdaki örnekte daha geniş gösterilmiştir.

```
10 REM mountains
20 DEFINT a-z
30 INK 0,1:INK 1,26
40 INK 2,6:INK 3,6
50 FOR i=4 TO 7:INK i,9:NEXT
60 FOR i=8 TO 15:INK i,20:NEXT
70 MODE 0:DEG:ORIGIN 0,150:CLG:MOVE 0,150
80 FOR x=16 TO 640 STEP 16
90 DRAW x,COS(x)*150+RND*100,4
100 NEXT
110 MOVE 0,0:FILL 4
120 cx=175:GOSUB 320
130 cx=525:GOSUB 320
140 SYMBOL 252,0,0,&C,&1F,&30,&7F,&FF
150 SYMBOL 253,0,6,&E,&F2,2,&F2,&FE
160 SYMBOL 254,0,&60,&70,&7F,&7F,&7F,&7F
170 SYMBOL 255,0,0,0,&F8,&EC,&FE,&FF
180 pr$=CHR$(254)+CHR$(255)
190 pl$=CHR$(252)+CHR$(253)
200 TAG:t!=TIME
210 FOR x=-32 TO 640 STEP 4
220 x2=((608-x)*2)MOD 640:hl=RND*10:hr=50*SIN(x)
230 GRAPHICS PEN 8,1:MOVE x,100+hr,,3:PRINT pr$;
240 GRAPHICS PEN 2,1:MOVE x2,115+hl,,3:PRINT pl$;
250 IF (TEST(x2-2,115+hl-12) AND 8)=8 THEN 380
260 IF TIME-t!<30 THEN 260
270 FRAME:t!=TIME
280 GRAPHICS PEN 7,1:MOVE x,100+hr,,2:PRINT pr$;
290 GRAPHICS PEN 13,1:MOVE x2,115+hl,,2:PRINT pl$;
300 NEXT
310 GOTO 210
320 MOVE cx,100
330 FOR x=0 TO 360 STEP 10
340 DRAW cx+SIN(x)*50+10*RND,100+COS(x)*25+10*RND,1
350 NEXT
360 DRAW cx,100:MOVE cx,90:FILL 1
370 RETURN
380 ENT -1,1,1,1
390 SOUND 1,25,400,15,,1,15
400 FOR y=100+hr TO -132 STEP -2
410 GRAPHICS PEN 7,1:MOVE x,y,,2:PRINT pr$;
420 GRAPHICS PEN 8,1:MOVE x,y-2,,3:PRINT pr$;
430 NEXT
440 GOTO 70
run
```

Nasıl çalışacağını anlamak için bir kere daha mürekkep (INK) numarasını ikilik olarak kafanızda canlandırın. En yüksek INK numarası (15) ile başlama sonucu, '8' bit içeren bütün INK'ler (15'den 8'e) koyu maviye dönüşecektir. Sonra '4' bit içeren '7'den 4'e) bütün INK'ler yeşile ayarlanacaktır. 2 ve 3 INK'leri 2 bit'e sahiptir ve kırmızıya ayarlanacaktır. Son olarak INK 1 parlak beyaza ayarlanırken INK Ø mavi olarak kalacaktır.
Ekranda grafikler yerleşme OR'lanmışlardır. 230 ve 240 satırlara bakınız. Belirli bir pixel'deki, ekranda görülen renk, o noktadaki bileşkenin en belirgin bit'ince kararlaştırılacaktır. Bu nedenle, 'daha belirgin' bir düzlemdeki bir tasvir her zaman, daha az belirgin' bir düzlemdeki bir tasviri karartacaktır. Arka plan gene aynı kalacaktır ve 'daha belirgin' tasvir kaldırıldığında gene gözükecektir. Bir tasviri kaldırmanın yolu 7,11,13 veya 14 INK numaralarıyla AND mürekkep modunu kullanarak onu işaretlemektir. Böylece sırasıyla 8,4,2 ve 1'in orijinal INK'leri (mürekkepleri) kaldıracaktır. 280 ve 290 satırlara bakınız.

Fazla Hafıza Kullanan Grafikler

Bu bölümü tamamlamak için 6123'de RAM'ın fazla 64K'sını kullanan, açıklayıcı bir "GRAFİK EKRAN DÜZENLEYİCİ" program sunuyoruz.

```
10 'SCREEN DESIGNER by DAVID RADISIC
20 ' copyright (c) AMSOFT 1985
30 '
40 ' Programı yazıp çalıştırmadan evvel RUN "BANKMAN" yazınız!
50 '************************************************
60 '
70 ON ERROR GOTO 2740
80 DEFINT a-x
90 MODE 1:ch=127:cmnd=1:pn(Ø)=Ø:pn(1)=26:pn(2)=15:pn(3)
   =6:pn(4)=Ø:pn=1:norx=1:menu=1:zzz=HIMEM
100 DIM command$(22)
110 norx$(Ø)="Normal":norx$(1)="XOR    ":norx$(2)="Trans
    p":norx$(3)="XOR    "
120 RESTORE:READ cmnds$(1),cmnds$(2):cmnd$=CHR$(16)+CHR
    $(&7F)+cmnds$(1)+cmnds$(2)
130 READ cmno:FOR i=1 TO cmno:READ command$(i):NEXT
140 READ st$:IF st$<>"**" THEN cmnd$(cmnd)=st$:cmnd=cmn
    d+1:GOTO 140
150 WINDOW #Ø,1,40,1,3:PAPER #Ø,Ø:PEN #Ø,1:CLS #Ø
160 WINDOW #1,1,40,4,4:PAPER #1,3:PEN #1,1:CLS #1
170 ORIGIN Ø,Ø,Ø,640,Ø,334
180 x=320:y=200:MOVE x,y
190 BORDER pn(4):FOR i=Ø TO 3:INK i,pn(i):NEXT
200 MASK 255,Ø:PAPER Ø:PEN 1:PAPER #1,3:PEN #1,1:GRAPHI
    CS PEN pn,norx
210 IF flag<>5 THEN 280
```

```
220 IF pn<2 THEN pnt$=CHR$(240):px=(pn+1)*13 ELSE IF pn
    <4 THEN pnt$=CHR$(241):px=(pn-1)*13 ELSE pnt$=CHR$(
    243):px=37
230 LOCATE px,2:PRINT pnt$;
240 LOCATE 1,1:PRINT USING"   PEN 0 : ##   PEN 1 : ##";
    pn(0);pn(1);
250 LOCATE 29,2:PRINT USING"Border : ##";pn(4)
260 LOCATE 1,3:PRINT USING"   PEN 2 : ##   PEN 3 : ##";
    pn(2);pn(3);
270 LOCATE px,2:PRINT " ";
280 LOCATE #1,1,1:PRINT#1,USING"X :####  Y :####     ";
    x;y;:PRINT #1,"Plot mode : ";norx$(norx+(undraw*2))
    ;" ";
290 IF flag=0 THEN GOSUB 2260
300 '
310 GOSUB 970
320 '
330 IF flag>0 THEN 390
340 IF i$="" THEN 390
350 cmnd=INSTR(cmnd$,i$):IF cmnd=0 THEN 390
360 IF cmnd=1 THEN CLG:x=320:y=200:GOTO 390
370 IF cmnd=2 THEN RUN 70
380 ON cmnd-2 GOSUB 1240,1410,1520,1640,1840,1860,1950,
    2020,2090,2120,2170,2200,2660,2660,2660,2660,2390,2
    330,2200
390 IF tx=0 AND ty=0 THEN 200
400 IF flag>0 THEN 440
410 GOSUB 630
420 GOSUB 680:FRAME:GOSUB 680
430 GOTO 200
440 MOVE tempx,tempy,pn,1
450 ON flag GOSUB 470,490,550,640
460 GOTO 200
470 PLOT x,y:GOSUB 630:PLOT x,y
480 RETURN
490 DRAW tempx+x,tempy:DRAW tempx+x,tempy+y
500 DRAW tempx,tempy+y:DRAW tempx,tempy
510 GOSUB 630
520 DRAW tempx+x,tempy:DRAW tempx+x,tempy+y
530 DRAW tempx,tempy+y:DRAW tempx,tempy
540 RETURN
550 MOVE tempx,tempy:DRAWR x,y
560 IF triside=0 THEN 580
570 DRAW tempxx,tempyy:DRAW tempx,tempy
```

```
580 GOSUB 630
590 MOVE tempx,tempy:DRAW tempx+x,tempy+y
600 IF triside=0 THEN RETURN
610 DRAW tempxx,tempyy:DRAW tempx,tempy
620 RETURN
630 x=x+tx:y=y+ty:RETURN
640 MOVE tempx,tempy:DRAW x,y
650 GOSUB 630
660 MOVE tempx,tempy:DRAW x,y
670 RETURN
680 ' draw and undraw cursor
690 IF flag=5 THEN RETURN
700 MASK 255,1
710 IF flag>1 THEN xx=tempx+x:yy=tempy+y ELSE xx=x:yy=y
720 IF flag=4 THEN xx=x:yy=y
730 IF flag=1 THEN xx=x:yy=y
740 IF undraw=1 THEN 820
750 GOSUB 790
760 MASK 255,0
770 IF i$=" " THEN GOSUB 2150:i$=""
780 RETURN
790 MOVE xx-4,yy,pn,1:DRAW xx+4,yy
800 MOVE xx,yy-4:DRAW xx,yy+4
810 MOVE xx,yy,,xorn:RETURN
820 nx=1:GOSUB 1220
830 FRAME:GOSUB 1220
840 IF i$=" " THEN nx=norx:GRAPHICS PEN pn,1:GOSUB 1220
850 i$=""
860 IF flag<>6 THEN 760
870 IF moved=0 AND j$<>"" AND (j$<CHR$(240) OR j$>CHR$(
    247)) THEN ch=ASC(j$):moved=1
880 IF moved=0 THEN RETURN
890 LOCATE 5,2
900 FOR i=ch-5 TO ch+5
910 PEN ABS(i<>ch)+1
920 ch$=CHR$(1)+CHR$(ABS(i+256)MOD 256)
930 IF ch=i THEN PRINT" "ch$" "; ELSE PRINT ch$;
940 NEXT
950 PEN 1:PRINT"   = "ch"  ";
960 GOTO 760
970 ty=0:tx=0:GOSUB 680:FRAME:GOSUB 680
980 IF INKEY(0)<>-1 OR INKEY(72)<>-1 THEN ty=16
990 IF INKEY(2)<>-1 OR INKEY(73)<>-1 THEN ty=-16
1000 IF INKEY(8)<>-1 OR INKEY(74)<>-1 THEN tx=-16
```

```
1010 IF INKEY(1)<>-1 OR INKEY(75)<>-1 THEN tx=16
1020 IF INKEY(21)<>-1 OR INKEY(76)<>-1 THEN tx=tx/8:ty=
     ty/8
1030 IF tx=0 AND ty=0 THEN moved=0 ELSE moved=1
1040 j$=INKEY$:i$=UPPER$(j$)
1050 IF (i$=" " OR i$=CHR$(13)) AND flag>0 THEN 1090
1060 IF flag=5 THEN 1120
1070 IF flag=6 THEN 1170
1080 RETURN
1090 ON flag GOSUB 1240,1410,1640,1860,1950,2020
1100 i$=""
1110 RETURN
1120 IF moved=0 THEN RETURN
1130 IF tx>2 THEN pn=(pn+1) MOD 5 ELSE IF tx<-2 THEN pn
     =ABS((pn<1))*5-1+pn
1140 IF ty>2 THEN pn(pn)=(pn(pn)+1) MOD 27 ELSE IF ty<-
     2 THEN pn(pn)=ABS((pn(pn)<1))*27-1+pn(pn)
1150 GRAPHICS PEN pn:PEN #1,pn
1160 tx=0:ty=0:BORDER pn(pn):RETURN
1170 IF tx<0 THEN ch=ABS(ch+255) MOD 256
1180 IF ty<0 THEN ch=ABS(ch+246) MOD 256
1190 IF tx>0 THEN ch=(ch+1) MOD 256
1200 IF ty>0 THEN ch=(ch+10) MOD 256
1210 tx=0:ty=0:RETURN
1220 TAG:MOVE xx-8,yy+6,pn,nx:PRINT CHR$(ch);:TAGOFF
1230 RETURN
1240 ' C
1250 IF flag=1 THEN 1290
1260 ro=1:GOSUB 2240
1270 tempx=x:tempy=y:flag=1
1280 RETURN
1290 IF tempx=x AND tempy=y THEN 1390
1300 PLOT x,y,,1
1310 tix=MAX(x,tempx)-MIN(tempx,x):tiy=MAX(y,tempy)-MIN
     (tempy,y)
1320 ti=SQR((tix↑2)+(tiy↑2))
1330 ORIGIN tempx,tempy
1340 PLOT 0,0,pn,0:MOVE 0,-ti
1350 FOR z=0 TO PI*2+0.01 STEP PI/(ti/2)
1360 DRAW SIN(z+PI)*ti,COS(z+PI)*ti,pn,norx
1370 NEXT z
1380 ORIGIN 0,0
1390 x=tempx:y=tempy:tempx=0:tempy=0:flag=0
1400 RETURN
```

```
1410 ' B
1420 IF flag=2 THEN 1470
1430 ro=2:GOSUB 2240
1440 tempx=x:tempy=y:flag=2
1450 x=0:y=0
1460 RETURN
1470 IF norx=1 THEN 1500
1480 MOVE tempx,tempy:DRAW tempx+x,tempy,,norx
1490 DRAW tempx+x,tempy+y:DRAW tempx,tempy+y:DRAW tempx
     ,tempy
1500 x=tempx:y=tempy:flag=0
1510 RETURN
1520 ' F
1530 ro=3:GOSUB 2240
1540 GOSUB 1620:IF i$=" " THEN 1600
1550 edgecol=VAL(i$)
1560 ro=4:GOSUB 2240
1570 GOSUB 1620:IF i$=" " THEN 1600
1580 filler=VAL(i$)
1590 MOVE x,y,edgecol:FILL filler
1600 flag=0:i$=""
1610 RETURN
1620 i$=INKEY$:IF (i$<"0" OR i$>"3") AND i$<>" " THEN 1
     620
1630 RETURN
1640 ' T
1650 IF flag=3 THEN 1700
1660 flag=3:ro=5:GOSUB 2240
1670 tempx=x:tempy=y
1680 x=0:y=0
1690 RETURN
1700 IF triside<>0 THEN 1770
1710 ro=6:GOSUB 2240
1720 MOVE 0,0,pn,1:GOSUB 590
1730 tempxx=tempx+x:tempyy=tempy+y:x=x/2:y=20
1740 triside=1
1750 GOSUB 550:GOSUB 590
1760 RETURN
1770 IF norx=1 THEN 1800
1780 MOVE tempxx,tempyy,,norx:DRAW tempx,tempy
1790 DRAW tempx+x,tempy+y:DRAW tempxx,tempyy
1800 tempxx=0:tempyy=0
1810 x=tempx:y=tempy:triside=0
1820 tempx=0:tempy=0:flag=0
```

```
1830 RETURN
1840 ' @
1850 norx=1:undraw=undraw XOR 1:RETURN
1860 ' L
1870 IF flag=4 THEN 1910
1880 ro=7:GOSUB 2240
1890 tempx=x:tempy=y:flag=4
1900 RETURN
1910 IF norx=1 THEN 1930
1920 MOVE tempx,tempy,,norx:DRAW x,y
1930 x=tempx:y=tempy:flag=0
1940 RETURN
1950 ' I
1960 IF flag=5 THEN flag=0:CLS:INK 3,tmpcol:INK pn,col:
     GOTO 1990
1970 CLS:flag=5:BORDER pn(pn)
1980 RETURN
1990 FOR i=0 TO 3:INK i,pn(i):NEXT:BORDER pn(4)
2000 IF pn=4 THEN pn=1
2010 CLS:RETURN
2020 ' A
2030 IF flag=6 THEN 2070
2040 tempx=0:tempy=0:CLS
2050 undraw=1:flag=6:norx=1:moved=1
2060 RETURN
2070 flag=0
2080 RETURN
2090 ' N
2100 norx=0
2110 RETURN
2120 ' E
2130 GRAPHICS PEN pn,0:TAG:MOVE xx-8,yy+6,,0:PRINT " ";
     :TAGOFF
2140 RETURN
2150 '<SPACE>
2160 PLOT x,y,pn,norx:RETURN
2170 ' X
2180 norx=1
2190 RETURN
2200 ' M
2210 menu=menu MOD 2+1
2220 GOSUB 2260:RETURN
2230 i$=UPPER$(INKEY$):IF i$="" OR INSTR(ser$,i$)=0 THE
     N 2230 ELSE RETURN
```

```
2240 CLS:undraw=0:PRINT cmnd$(ro);:LOCATE 1,3:PRINT"<SP
     ACE> ";:IF ro=3 OR ro=4 THEN PRINT"To exit"
2250 RETURN
2260 CLS:flag=-1
2270 FOR i=1 TO LEN(cmnds$(menu))
2280 ps=i+ABS(menu=2)*LEN(cmnds$(1))
2290 PEN 1:PRINT"<"MID$(cmnds$(menu),i,1)">"MID$(comman
     d$(ps),2,4)" ";
2300 NEXT
2310 PRINT"<CLR>   <DEL>   <SPACE>";
2320 RETURN
2330 ' S
2340 GOSUB 2460:IF filename$="" THEN 2370
2350 GOSUB 2550
2360 SAVE filename$,b,&C000,&4000
2370 GOSUB 2260
2380 RETURN
2390 ' R
2400 GOSUB 2460:IF filename$="" THEN 2440
2410 GOSUB 2730
2420 LOAD filename$,&C000
2430 GOSUB 2570
2440 GOSUB 2260
2450 RETURN
2460 CLS:LOCATE 10,3:PRINT"<RETURN> to Abort!";
2470 LOCATE 1,1:PRINT"Enter Filename :";
2480 INPUT "",filename$:IF filename$="" THEN RETURN
2490 n=INSTR(filename$,"."):IF n=0 THEN 2520
2500 IF n=1 THEN 2460
2510 filename$=LEFT$(filename$,n-1)
2520 filename$=LEFT$(filename$,8)+".scn"
2530 CLS
2540 RETURN
2550 FOR i=0 TO 4:POKE &C000+i,pn(i):NEXT
2560 RETURN
2570 FOR i=0 TO 4:pn(i)=PEEK(&C000+i) MOD 27:NEXT
2580 cn=0:FOR i=0 TO 2:IF pn(i)=pn(i+1) THEN cn=cn+1
2590 NEXT:IF cn=3 THEN 2630
2600 FOR i=0 TO 3:INK i,pn(i):NEXT
2610 BORDER pn(4):pn=1:GRAPHICS PEN pn
2620 RETURN
2630 pn(0)=0:pn(1)=26:pn(2)=15:pn(3)=6:pn(4)=0
2640 GOTO 2600
2650 ' 1, 2, 3, & 4
```

```
2660 CLS:PRINT"Do you wish to <S>tore":PRINT TAB(16)"<R
     >etrieve":PRINT TAB(13)"or <E>xchange  the screen
     ?"
2670 ser$="SRE"+CHR$(13):GOSUB 2230:IF i$=CHR$(13) THEN
      2260
2680 bnk2=(cmnd-13):bnk1=1
2690 IF i$="S" THEN CLS:GOSUB 2550:|SCREENCOPY,bnk2,bnk
     1
2700 IF i$="R" THEN GOSUB 2730:|SCREENCOPY,bnk1,bnk2:GO
     SUB 2570
2710 IF i$="E" THEN CLS:GOSUB 2730:GOSUB 2550:|SCREENSW
     AP,bnk2,bnk1:GOSUB 2570
2720 GOSUB 2260:RETURN
2730 FOR i=0 TO 3:INK i,0:NEXT:BORDER 0:RETURN
2740 CLS:GOSUB 2600:RESUME 2260
2750 DATA "CBFT@LIANEXM","1234RSM"
2760 DATA 19,Circle,"Box  ","Fill ",Triangle,Alternate,
     "Line ","Inks ",ASCII,Normal,Erase,"Xor  ","Menu "
     ,"1st  ","2nd  ","3rd  ","4th  ",Restore,"Save ","
     Menu "
2770 DATA Circle,Box,Edge colour,Filler colour,Triangle
      1,Triangle 2,Line,**
```

Bu programda kullanılan iki RSX komutu SCREENCOPY ve SCREENSWAP, sisteminizin diskler paketinin 1. kısmındaki "Bank Manager" yardımcı ünitesince hazırlanmıştır. Komutlar, uygun hafıza blokları arasındaki ve hafızanın iki 64K'lık kümesi arasındaki çeşitli görüntülerin kopyasının çıkartılmasını ve değiştirilmesini kolaylaştırır.
Bu nedenle 'Ekran Düzenleyici Program'ı çalıştarmadan önce "Bank Manager" yardımcı ünitesini çalıştırmalısınız. Bunu yapmak için de, sisteminizin diskler paketinin 1. kısmını yerleştiriniz ve şunları yazınız.

```
run "bankman"
```

Şimdi Ekran düzenleyici Programı çalıştırabilirsiniz.

Sonra Ne Olacak?

Önce Bank Manager yardımcı ünitesini, sonra Ekran Düzenleyici Programı çalıştırdıktan sonra, ekranın ortasındaki grafik kürsörü ile tercih haklarınızı ekranda göreceksiniz.

Sonra seçiminize uygun harfe basınız (Örneğin çember için C'ye)
Deneme için,

C

yazınız. Sonra kursör için yukarı düğmesine, parıldayan grafik kürsörü ekranın merkezinden 1 inch kadar yukarı gidine kadar basınız.

Son olarak çember fonksiyonunu yürürlüğe sokmak için "SPACE" tuşuna basınız. Sonra menüyle dönebilirsiniz.

M yazma (Menü için) size ekranı koruyabileceğiniz veya yeniden düzenleyebileceğiniz veya 1.2.3 ve 4 ekranların içeriklerini (hafızanın 2. 64K'lık bölümünde) ustalıkla kullanabileceğiniz alternatif bir menü getirecektir.

Ekran fonksiyonları kullanmak için, hafıza numarasını (1,2,3 veya 4) yazınız. Daha ileri bir menü ekrana gelecektir.
Bunlardan şu özellikleri seçebiliriz.

S Ekranı saklamak için
R Ekranı düzeltmek için
E Ekranları değiştirmek için
(RETURN) İşlemi bitmeden durdurmak için

Örneğin eğer ekrandaki görüntüyü 2 numaralı hafızada saklamak istiyorsanız, önce 2 sonra S yazınız.

Başka bir menüye dönerken (bir düzeltme (R) saklama (S) veya yapılan herhangi bir işlemden sonra), M yazarsanız tekrar orijinal menüye dönebilirsiniz.

Ekran Düzenleyici bir dizi kolaylık sağlayacaktır. Örneğin kutular, çemberler, üçgenler, çizgiler, noktaları dizme, doldurma ve karakter çizim.

Bir ekran görüntüsü tamamlandığında, alternatif menüden Saklama ünitesini kullanarak ileri gösterme/basma için disklerde saklanabilir. Düzeltme (R) kullanarak da istenildiği anda tekrar yüklenebilir.

Böylece bu el kitabındaki son bölümü de tamamladık. Bu rehberde sunduğumuz programları kullanır, inceler ve deneyler yaparsanız, AMSTRAD BASIC ve 6128'in kendisi hakkında yeterli bilgi sahibi olabilirsiniz.

İleri Detaylar

BASIC ve program ayrıntılarıyla 6128 hakkında ileri detaylar, CP/M işlemleri hakkında ve Dr.LOGO dili hakkındaki bilgiler için çok AMSOFT yayınından veya bağımsız yayınlardan elde edilebilir.

Son olarak size CP/M son kullanıcı Lisans anlaşması terimleri sözlüğü, bazı oyun programları listesi ve el kitabımızın geniş bir indexini içeren 4 ilave bölüm sunuyoruz.

Bu el kitabını öğretici ve ilginç bulduğunuzu ümit eder, CPC 6128'i satın aldığınız için teşekkür ederiz.

EK 1
Digital Research ve AMSTRAD

Son Kullanıcı Program Ruhsatı Anlaşması

KULLANICININ DİKKATİNE-LÜTFEN BU NOTU DİKKATLE OKUYUNUZ. BU RUHSAT ANLAŞMASINI OKUMADAN DİSKET PAKETİNİ AÇMAYINIZ.

DİSKET PAKETİNİ AÇMANIZ BU ANLAŞMADAKİ KOŞUL VE KURALLARI TANIYARAK ANLAŞMAYI BENİMSEYİP KABUL ETMENİZ ANLAMINA GELİR.

1. Açıklamalar:

1- DRI. DIGITAL RESEARCH (CALIFORNIA) INC, P.O.Box 579, Pasific Growe California 93950 anlamına gelir ve yayın hakkı ya da programın yetkili ruhsat sahibidir.

2- Makina, programı kulandığınız tek mikrobilgisayar anlamına gelir. Çoklu MİB (CPU) sistemleri bu ruhsata ek başka ruhsatlar olmaksızın kullanılamazlar.

3- Program, program seti ile pakette bulunan ilgili materyallar ve dökümanlar ile DRI tarafından -sizin kullanım şekliniz ve gideceğiniz düzenlemeler gözetilmeksizin- hazırlanan ekler ve arttırımlar anlamına gelir.

4- AMSTRAD ise AMSTRAD CONSUMER ELECTRONICS PLC., Brentwood House 169 Kings Road, brentwood, Essex CM14 4 EF U.K. anlamına gelir.

İstediğiniz sonuçları elde etmede kullanacağınız programın seçimi, kullanımı ve elde ettiğiniz sonuçların tüm sorumluluğu size aittir.

2. Ruhsat

Aşağıda belirtilenleri yapabilirsiniz:

1- Programı tek bir makinada kullanınız.

2- Programı geri saklama ya da dizinleme amacı ile;tek bir makinada kullanırken yedeğini bulundurma veya dizinleme amacıyla başka bir makinaya okunabilir ve yazılabilir biçimde kopya edebilirsiniz. Ancak bu amaçlar içinde olsa üç (3) kopyadan fazlasını yapamazsınız. (Zaten bu tür programlar kopya sayısını sınırlayan bir mekanizma içerirler. Bu tür programlar "copy protected" (kopya önlemeli) şeklinde bir ibre taşırlar.) Dökümanların ve diğer yazılı-basılı materyallerin kopyalarının çıkarılması yasaktır. Programın parça parça veya çevirisiz halinin kopyası da yasaktır.

3- Tek makinanızda kullanmak amacıyla programda bir değişiklik yapabilir ve/veya başka bir programla birleştirebilirsiniz. (Ancak bu yeni oluşan programdan başka bir programa yapılan herhangi bir eklendi bu anlaşmanın koşul ve kurallarının konusu olacaktır.)

4- Programı ve ruhsatı, DRI'ya diğer partinin ad ve adresini bildirdiğiniz ve diğer parti izleyen koşullara uyuyorsa o partiye aktarabilirsiniz: a) Bu anlaşma'nın koşul ve kurallarına uymalıdır, b) Kayıt kartının bir kopyasını imzalayıp DRI'ya yollamalısınız ve c(Cari transfer ücretini yollamalısınız. Eğer programı aktarırsanız, bu durumda aktarılmamış kopyaları yok etmeniz ya da tüm kopyaları da -özgün program dahil- yazılı veya makina tarafından okunabilir formda aktarmanız gerekir. Burada sözü geçen aktarma diğer programa eklenen bölümleri veya değiştirilen programı kapsar.

Değişiklik yapılan yada bölüm eklentileri yapılan her kopyada yayın hakkı ibaresini yeniden yaratmak ve bulundurmak zorunludur.

HER DİSKET BİR SERİ NUMARASINA SAHİPTİR VE ÜÇÜNCÜ BİR PARTİYİ OLUŞTURACAK BİÇİMDE KULLANILAMAZ, KOPYA EDİLEMEZ, DEĞİŞTİRİLEMEZ, AKTARILAMAZ VE BENZERİ İŞLEMLERE KONU OLAMAZ.

PROGRAMA AİT KOPYALAR, DEĞİŞTİRMELER VE EKLENTİLERDEN HERHANGİ BİRİNİ BAŞKA BİR PROGRAMA AKTARDIĞINIZDA RUHSATINIZ OTOMATİKMAN İPTAL EDİLECEKTİR.

3. Kural

Ruhsat, iptal edilinceye kadar geçerlidir. İptal etmek için yapabileceğiniz; programın bütün kopyalarının sonradan yapılmış değişiklik ve eklentilerin tümüyle birlikte, var olan bütün formlarını istediğiniz zaman imha etmektir. Ruhsatınız bu anlaşmada geçen herhangi bir maddenin ihlali sonucunda da iptal edilir. Böyle bir durumda da programı; bütün kopyaları, eklenti ve değişikliklerle birlikte imha etmeyi kabul etmiş sayılırsınız.

4. Sınırlı Garanti

PROGRAM "OLDUĞU GİBİ" SAĞLANMIŞTIR. PAZARLAMA GÜVENCESİ VEYA BELLİ BİR AMACIN SAĞLANMASI İÇİN DE OLSA VEYA BÖYLE GÖRÜNSE, HATTA BUNUN DOĞRULUĞU KANITLANSA BİLE NE AMSTRAD NE DE DRI HİÇBİR BİÇİMDE GARANTİ VERMEZLER. PROGRAMIN KALİTE VE PERFORMANSIYLA İLGİLİ DOĞABİLECEK TÜM TEHLİKELERİN SORUMLULUĞU SİZE AİTTİR. PROGRAMIN HERHANGİ BİR EKSİĞİ BULUNDUĞUNDA GEREKEN SERVİS, TAMİR VEYA DÜZELTME İŞLEMİNİN HARCAMALARI SİZE (ve AMSTRAD yaDA DRI'ya) AİTTİR.

Ne AMSTRAD nede DRI programın içerdiği fonksiyonların sizin isteklerinizi tümüyle karşılayacağına yada programın hatısız ve kesintisiz çalışacağına garanti vermezler.

Bununla birlikte AMSTRAD, programınızın yer aldığı diskte malzeme veya işçilik nedeniyle doğan arızaları;makbuzunuzda belirttiğiniz teslim tarihini izleyen 90 gün içinde normalin dışında kullanmamış olmanız kaydıyla; gidermeyi garanti eder.

5. Çözüm ve İyileştirme Yolları

AMSTRAD'ın tüm sorumluluğuna ek olarak sizin sorumluluğunuz, bu "Sınırlı Garanti"nin gereklerini karşılamayan disketinizi, makbuzunun bir kopyasıyla birlikte AMSOFT'a yollayarak değiştirilmesini sağlamaktır.

AMSTRAD YADA DRI DOĞABİLECEK ZARARLARA İLİŞKİN ÖNERİLERDE BULUNMUŞ DA OLSA, BAŞKA BİR PARTİ DE BENZERİ İSTEKLERDE BULUNMUŞ OLSA; NE AMSTRAD NE DE DRI KAR VE TASARRUFLARIMIZIN KAYBINDAN YADA BAŞKA YOLLARDAN DOLAYLI VEYA DOLAYSIZ DOĞABİLECEK ZARARLARA İLİŞKİN OLAYLARDA HİÇBİR SORUMLULUK KABUL ETMEZ.

6. Kayıt Kartı

DRI zaman zaman programlarında bazı yenilemeler yapmak isteyebilir. Eğer geçerli biçimde imzalanmış bir kayıt kartınız DRI'nın ana bürosundaki dosyanızda ya da bu konuda yetkili bir temsilcimizde var ise bu değişikliklerden yararlanabilirsiniz. DRI program yenilemeleri yapmaya ve size de benzeri yenileme hizmetleri sunmaya zorunlu değildir. y

7. Genel

Hiçbir şekilde alt lisans, alt ruhsat düzenleyemez ve veremezsiniz, yada sözleşmede geçtiği gibi programı ve/veya ruhsatı başkasına devredemez, aktaramazsınız. Yoksa bu alt ruhsatlar ve aktarım/devredim yüzünden boşu boşuna yükümlülük altına girmiş olursunuz.

Bu anlaşma İngiliz yasalarına uygun olarak hazırlanmıştır ve İngiliz yasalarının güvencesi altındadır.

Bu anlaşmayla ilgili herhangi bir sorununuz olduğu zaman DRI'nın aşağıdaki adresine yazabilirsiniz:

"Digital Research Inc., P.O.Box 579, Pacific Grove, California 93950, USA"

BU ANLAŞMA İLAN, İSTEK BİLDİRİMİ VEYA BAŞKA YOLLARLA HİÇ KİMSE TARAFINDAN DEĞİŞTİRİLEMEZ, YALNIZCA SİZLE AMSTRAD VEYA DRI'NIN YETKİLİ TEMSİLCİLERİ ARASINDA YAPILABİLECEK YAZILI BİR ANLAŞMAYLA DEĞİŞTİRİLEBİLİR.

BU ANLAŞMAYI OKUDUĞUNUZU, ANLADIĞINIZI VE İÇERDİĞİ KOŞUL VE KURALLARA UYACAĞINIZI ONAYLAMIŞ BULUNUYORSUNUZ. AYNI ZAMANDA BU ANLAŞMANIN DAHA ÖNCE AMSTRAD VE/VEYA DRI İLE YAPMIŞ OLDUĞUNUZ YAZILI VEYA SÖZLÜ TEKLİF VEYA ÖNANLAŞMALARIN VE BU ANLAŞMANIN İÇERDİĞİ KONULARDA DAHA ÖNCE AMSTRAD VE DRI İLE YAPILMIŞ BİLDİRİMLERİN YERİNE GEÇEN VE TÜM BUNLARI DEVRE DIŞI BIRAKAN SON ANLAŞMA AOLDUĞUNU KABUL ETMİŞ VE ONAYLAMIŞ BULUNUYORSUNUZ.

BU ANLAŞMA KAZANILMIŞ HAKLARINIZI ETKİLEMEYECEKTİR.

Ek 2
Terimler Sözlüğü

6128 kullanıcıları için bilgisayar dünyasında çok kullanılan bazı terimlerin açıklamaları

Accumulator (akümülatör)

Mikro bilgisayarların kalbi olan mikro işlemcide, işlem gören bilginin geçici olarak saklandığı bölüm. Makina dili ile programlamada çok sık kullanılır-BASIC kullanıcıları varlığından haberdar olmazlar.

Acoustic coupler (Akustik bağlaç)

Akustik modem olarak da bilinir. Bilgisayarın normal telefon hattı aracılığı ile haberleşmesini sağlayan elektronik aksamdır. Bu yolla bir bilgisayar diğer bilgisayarlarla veya halka açık danışma sistemleri (Prestel gibi) ile iletişim sağlayarak program ve bilgi alış verişi yapabilir.

Address (Adres)
Bilgisayarların belleğindeki bir yeri simgeleyen sayı. Adres kullanılarak hafızadaki bir bölgede ne yazıldığı okunabilir; eğer bu bölge RAM'da ise aynı zamanda buraya yeni bir değer verilebilir.

Adventure game (Macera oyunu)

Kimilerine göre çok zevkli kimilerine göre ise akıcı olan bir bilgisayar oyunu tipi. Metin özelliği gösteren bu tür oyunda, oyuncu bir labirenti çözmeye veya zor bir durumda kurtulmaya çalışır.

Algorithm (Algoritma)

Bir problemin çözümünü sonlu sayıda adımda matematiksel ve mantıksal işlem şeklinde tarif etme yöntemi.

Alphanumeric (Alfa sayısal)

Bir harf veya rakam ve bir grafik karakteri arasındaki farkı gösteren sıfat.

ALU

Aritmetik lojik ünitenin kısa adı. Aritmetik mantık birimi Mikro işlemcide aritmetik ve mantıksal işlemlerin yapıldığı birim. Makina kodu ile programlama dışında direkt olarak kullanılamaz.

Ambiguous File Name (Belirsiz dosya adı)

Bir veya daha çok wild card karakteri içeren dosya adı. Ortak özellikteki bir veya daha çok dosyayı aynı anda ifade etmek için kullanılır.

AMSDOS

Amstrad disk işletim sisteminin kısa adı. Locomotive BASIC'in disk dosyalarına erişebilmesini sağlayan program.

AMSOFT

AMSTRAD'ın özel bilgisayar destek bölümü. Özellikle 6128 ve kullanımını, yazılım, donanım, yayın gibi hizmetlerle destekler.

A/D
Analogue

Başlangıç ve varış noktaları arasındaki değişimin ani olması yerine adım adım gerçekleştiği durum. Bilgisayarlar sayısal aletlerdir, doğal olayların çoğu ise örneksel tabanlıdır. Bu yüzden bilgisayar örneksel bir kaynaktan alınan verileri işlemeden önce bir örneksel sayısal (A/D) çevirim yapmalıdır.

Animation (Canlandırma)

Karton filimler canlandırmanın en yaygın örnekleridir. Bilgisayar canlandırması, hareketli grafikler ile gerçek hareket izlenimi vermeye çalışılır.

Applications program

Özel amaçlı program

Arcade Game (Hareketli oyun)

Hareketli bilgisayar oyun türü. Oyuncunun kontrolündeki figürler tehlikelerden korunmak ve oyunun amacını gerçekleştirmek için çaba gösterirler. Bu tür oyunlar, genellikle refleksleri geliştirirler fakat bilgisayar öğrenenlere eğitim konusunda yardımcı olmazlar.

Architecture (Yapı)

Bir mikro bilgisayarın veri kaynağı, çevre birimleri ve işlemcisi arasındaki ilişki planı

Argument

Bağımsız bir değişken. Örneğin, $x + y = z$ ifadesinde, x, y ve z terimleri birer A R G U M E N T dir.

Array (Dizi)

İçerisinde yatay ve dikey koordinatları ile adreslenerek bilgi saklanan 2 boyutlu matris.

Artificial intelligence AI (Yapay zeka)

Programın geçmişteki deneyimlerden öğrenmesini sağlayan programlama tekniği.

ASCII

Bilgisayarlarda kullanılan 128 adet harf, sayı ve özel işaretlerden oluşan karakter setini belirleyen standart

Assembler

Assembly dilindeki komutları, makina dili komutlarına çeviren bilgisayar programı.

Backup (Yedek)

Yardımcı bellekteki bilgilerin, herhangi bir aksaklığa karşı korunmak amacıyla çıkarılmış kopyası. Bu işlem bir disk veya dosyanın kopyasını yapmayı ifade eder.

Bar code (Çubuk kodu)

Düşük güçlü lazer ile okumak gibi teknikler kullanılarak bilgisayar tarafından okunması mümkün olacak şekilde basılmış kodlar.

Base (Taban)

Herhangi bir sayı sisteminde kullanılabilecek değeri.

BASIC

Beginner's All-purpose Symbolic Instruction Code kelimelerinin kısa adı. Tüm ev bilgisayarlarında kullanılan bir programlama dilidir. BASIC özellikle öğrenim ve kullanım kolaylığı ön planda tutularak hazırlanmıştır. Bu dilde yazılan bir program, yazılımın her safhasında çalıştırılarak denenebilir. Buna karşı derleyici kullanan dillerde yazılan programlar tamamı yazılıp çalıştırılmadan denenemezler.

Baud

Saniye başına iletilen bit sayısı. Bilgisayarlar ve çevre birimleri arasındaki iletim hızı birimi.

BCD

Binary Code Decimal kelimelerinin kısa yazılışı. İkili kodlanmış onlu yazım. Her basamağın dört ikili rakam ile ifade edilmesi yolu ile kodlanmış desimal sayı.

BDOS

Basic Disk Operating System kelimelerinin kısa yazılışı Ana disk işletim sistemi. CP/M işletim sisteminin, kullanıcı programım CP/M fonksiyonlarını kullanabilmesini sağlayan bölüm

Benchmark (Denek izi)

Bilgisayar donanım ve yazılım elemanlarını diğerleri ile kıyaslamak üzere yapılan standart testler ve deneyler. Örneğin 99.999'un karekökünün karesinin hesaplanması.

Binary (İkili)

Tabanı iki olan, sadece 1 ve 0 rakamlarının kullanıldığı sayı sistemi.

Binary number (İkili sistem)

İkili sistemle ifade edilen sayı. 6128 programlamasında x ön eki ile gösterilir. Örneğin: X0101 = 5 (ondalık)

BIOS

Basic Input/Output System kelimelerinin kısa yazılışı. CP/M'in kullanıldığı bilgisayarın özelliklerine göre yazılmış kısmı. Ekran, klavye, disk ve diğer çevre birimleri aracılığıyla yapılan giriş ve çıkışları gerçekleştirir.

Bit (Bit)

Binary Digit'in kısa yazılışı. İkili sayı sisteminde kullanılan 0 ve 1 rakamlarından biri.

Bit Significant (Anlamlı bit)

Bir bayt'ın içerdiği bilgi kendisini oluşturan sekiz bitin her birinin durumlarına bakılarak okunur. Bayt'ın sayısal değeri anlamsız olabilir.

Boolean algebra (Boolean cebiri)

Sadece doğru ve yanlış değerini alabilen mantıksal önermeler. Genellikle 1 ve 0 ile gösterilirler.

Boot (ön yükleyici)

Bir işletim sisteminin hafızaya yüklenmesi işlemi. BASIC'den CP/M'e geçerken önce küçük bir önyükleyici program diskden yüklenir, daha sonra bu program işletim sisteminin kalanını yükler.

Booting or Bootstrapping (ön yükleme)

Programlar ve işletim sistemleri kendi kendilerini yükleyemezler, genellikle ROM'da yazılı küçük bir programla hafızanın belli bir yerine alınırlar.

Buffer (Ara bellek)

Bilgi akışındaki hız farklılıklarının etkisini önlemek üzere bilginin geçici olarak yüklendiği birim.

Bug (Hata)

Bilgisayar sisteminin kendisinde meydana gelen çeşitli aksaklıklar. Örneğin anlamsız bir verinin anlamsız sonuçlar doğurması (dört tuşa basılınca ekranın renk değiştirmesi gibi) veya programım belli bir yerinde tüm hafızanın silinmesi.

Built-in commands (Mevcut komutlar)

Bir işletim sisteminin parçası olan komutlar. Disk'den yüklenen komutlara göre daha hızlıdırlar.

Bus (Hat)

Bilgi bitlerini bir yerden diğerine taşıma işlemini yerine getiren sinyal yolları. 6128'de üzerinde EXPANSION yazan bağlantı yeri.

Byte (Bayt)

Sekiz bitten oluşan ve CPU tarafından hafızaya yazılıp okunması mümkün olan en küçük hafazı birimi.

CAD

Computer Aided Design kelimelerinin kısa yazılışı. Bilgisayar yardımı ile dizayn işlemi. Bilgisayarın hesaplama ve grafik kabiliyetleri kullanılarak bir elektronik çizim masası görevi üstlenmesi.

CAE

Computer Aided Education kelimelerinin kısa yazılışı. Bilgisayar yardımı ile eğitim işlemi. Bilgisayarın eğtimde yardımcı olarak kullanılması (CAE), bilgisayar aracılığı ile talimat verme (CAI) ve öğretme (CAL) olmak üzere iki şekildedir.

Cartridge (Kartuş)

Bilgisayarda özel olarak hazırlanmış bir sokete takılan ve yazılım içeren entegre devre. Kartuş üzerindeki yazılımın yüklenmesi ve çalışması hızlı ve kolaydır, fakat fiatı disk üzerinde olanlara göre oldukça fazladır.

Cassette

Bilinen kayıt bandından başka olarak, üzerine ROM yazılım kaydedilen çeşitli araçları ifade eder.

CCP

Console Command Processor kelimelerinin kısa yazılışı. CP/M'in, klavyeden yapılan girişleri yorumlayan ve işleme koyan modülü.

Character (Karakter)

Harfler, rakamlar ve grafik sembolleri dahil olmak üzere bilgisayar tarafından balirtilebilen ve gösterilebilen tüm semboller.

Character set (Karakter tablosu)

Bir bilgisayar yada yazıcıda mevcut olan tüm semboller. Bir yazıcı, karakter setinde mevcut olmayan bir karakteri bağlı olduğu bilgisayarınkinde bulunsa bile masamaz.

Character string (Karakter dizisi)

Tek bir ünite halinde saklanabilen ve işlem gören karakterler dizisi. Örneğin bir kelime veya kelimeler topluluğu.

Chip (Çip)

Yaygın olarak her türlü yekpare elektronik entegre devreyi ifade etmek için kullanılır. Gerçekte ise üzerine özel yollarla devre işlenmiş ufak silikon parçasıdır.

Clock (Saat)

Bir bilgisayarın işlemesindeki zaman uyumunu sağlamak için kullanılan zamanlama sistemi. Saat ve tarihi tutanına gerçek zaman saati denir.

Code (Kod)

Literatürün dışında, programcılar tarafından genellikle makina kodunun kısaltılması olarak kullanılır.

Cold start (Her şeyi silerek yeniden başlama)

Bir işletim sisteminin yüklenmesi ve çalıştırılması işlemi. CP/M'de bu lCPM komutu ile sağlanır.

Command (Komut)

Bir programa yapacağı işlemi belirten sözcük.

Compiler (Derleyici)

BASIC gibi yüksek seviyeli bir dilde yazılmış programı makina diline çeviren ve böylece daha hızlı çalışmasını sağlayan program.

Computer generations (Bilgisayar jenerasyonları)

Bilgisayar teknolojisindeki bazı önemli gelişmeler dönüm noktası olarak kabul edilirler ve bunların aralarındaki dönemler birer bilgisayar nesili olarak adlandırılırlar.

Computer literacy (Bilgisayar literatürü)

Bilgisayarları anlama anlamında bir deyim.

Console mode (Klavye modu)

CP/M'in direkt modu. Ekranda A> belirdiği ve sistemin bir komut beklediği hal.

Corruption (Bozulma)

Bir disk dosyasının veya hafızanın istenmeyen bir şekilde tahrip olması veya değişmesi.

CP/M

Control Program for Microcomputers kelimelerinin kısa yazılışı. Digital Research tarafından hazırlanmış, disk kullanan bir işletim sistemi. Değişik bilgisayarlar için yazılmış programların kullanılmasını sağlayan standart bir arabirimdir.

CPU

Central Processing Unit kelimelerinin kısa yazılışı. Emirleri derleyen ve yerine getirilmesini sağlayan bilgisayarın en önemli parçalarından biri. Bir mikro bilgisavarda CPU, mikroişlemcinin kendisidir.

Cursor (Kursör)

Ekranın üzerinde kontrolun nerede olduğunu gösteren işaret.

Cursor control keys (Kursör kontrol tuşları)

Kursörü ekran üzerinde hareket ettiren ve genellikle hareketli oyunlara kumanda için kullanılan, üzerlerinde yönü gösteren oklar olan tuşlar.

Daisy-wheel printer (Bir çeşit yazıcı)

Yüksek kaliteli veya daktilo kalitesinde yazı yazabilen yazıcı. Yazı, harflerin mürekkebe veya şeride karşı bastırılması ile yazılır.

Database (Veri tabanı)

Kendi içinde kayıtlara ve alanlara ayrılarak, yapısal bir biçimde organize edilmiş olan veri dosyaları.

Data capture (Bilgi algılanması)

Ana bilgisayara herhangi bir şekilde bağlı olan tüm dış kaynaklardan gelen verilerin toplanmasını ifade eden terim.

Debugging (Hata bulma)

Bir programdaki hataların çeşitli metotlar kullanılarak düzeltilmesi işlemi.

Decimal notation (Onluk gösterim)

Denary diye de bilinen ve günlük yaşamda kullanılan 10 tabanlı sayı sistemi.

Default (ön değer)

Herhangi bir tercih yapılmadığında, kabul edilecek normal değer. Örneğin CP/M başlatıldığında Drive:A normal sürücü olarak kabul edilir.

Delimiter (Ayırıcı)

Separatör'e bakınız.

Diagnostic (Diyognostik)

Bilgisayar tarafından, programdaki bir hatayı belirlemek için otomatik olarak üretilen mesaj.

Digital (Sayısal)

Bilginin sürekli olmayan, ayrık sinyallerle gösterilmesine verilen ad.

Digitiser (Sayısallaştırıcı)

Örneksel bir formdan, sayısal bir forma dönüşümü sağlayan cihaz.

Directory (Rehber)

Disk üzerindeki dosyaların adları, uzunlukları gibi bilgileri tutan yer.

Disc (or disk)

Bilgi saklamak için kullanılan üzeri manyetik oksit ile kaplı yassı ve yuvarlak plastik parçası. Disk bir koruyucu içerisine yerleştilir ve okuyucu kafa için bir açıklık bırakılır. 3'''lik diskde bu açıklık, disk sürücüden çıkınca otomatik olarak kapanan bir metal kapakla kapatılır.

Disk drive

Diskden veri okumak ve yazmak için kullanılan alet.

Documentation (Belgeleme)

Bilgisayar veya programların beraberinde verilen ve nasıl kullanıldıklarını açıklayan el kitapları.

DOS

Disk Operating System kelimelerinin kısa yazılışı. Bir disk sürücüsünü tüm işlemlerini kontrol eden program.

Dot matrix

Izgara şeklindeki düzenlenmiş ve istenenler seçildiğinde bir karakteri gösteren noktalar topluluğu.

Double sided (Çift yüzlü)

İki tarafında bilgi saklanabilen disk. Çift taraflı bir disk sürücüsü disk çevirmeye gerek kalmadan iki yüzüde ulaşabilir.

Download

Bir bilgisayarda diğerine bilgi alış verişi işlemi.

Dr. LOGO

LOGO programlama dilinin, Digita Research tarafından yazılmış şekli.

Dumb terminal

Sadece veri giriş ve çıkışı için kullanılan, herhangi bir veri işleme özelliği olmayan bilgisayar terminali.

Edit

Veride program veya bir metinde değişiklik veya düzeltme yapma işlemi.

Editor

Genellikle bilgisayarın ROM'unda bulunan ve metin düzeltme işlemini mümkün kılan bir program.

EPROM

Eraseable Programmable Read Only Memory kelimelerinin kısa yazılışı. Bu PROM'a benzer fakat yonganın içerdiği bilginin ultraviyole ışık ile silinebilmesi bakımından farklıdır. EEPROM'da elektronik olarak silinebilmesi dışında aynı özelliklere sahiptir.

Expression

Veriler kullanılarak istenen hesabı yapmak için program içerisine yerleştirilen basit veya karmaşık bir formül. İfade, genellikle ne tip veri kullanılacağını belirtir. Dr. LOGO'da bir ifade, yapılabilecek işin ismi ve gerekli olan verilerden oluşur.

Fifth generation computers

Yapay zekanın geliştirilmesi ile kendi kendini programlama özelliğine sahip olmaları beklenilen büyük bilgisayarlar.

File

Kaset veya disk üzerine saklanmış veri toplulukları.

File name

Bir dosyanın adı. Dr. LOGO'da bir dosya adı 8 karakter uzunluğunda olabilir. CP/m ve AMSDOS'da ise buna ek olarak bir noktayı takiben üç karakter daha yazılabilir.

Firmware

ROM'un ihtiva ettiği yazılım. Yazılım ve donanım arasındaki bir geçiştir.

Fixed-point number

Ondalık kısmı belli ve sabit bir pozisyonda olmak üzere saklanan ve işlem gören sayı.

Floating-point number

Ondalık kısmı gerektiği gibi yer değiştirebilecek şekilde saklanan ve işlem gören işlem sayı. Bu yöntem büyük sayılarda daha kullanışlıdır.

Floppy disc

Diske bakınız.

Flowchart

Program akışının diyagramlarla gösteriliş biçimi.

Forth

Yüksek hızlı bir programlama dili. Hız ve karmaşıklık bakımından yüksek seviyeli diller ile makina kodu arasında yer alabilir. Yeni başlayanlar için tavsiye edilmez.

Function key

Kendi görevine ilaveten veya yerine değişik bir fonksiyon getirmek için programlanmış tuş.

Gate

Belli girdiler için çıktılar üreten sayısal devre. Basit mantıksal fonksiyonlar kendilerine karşılık gelen geçitlerle gerçekleştirilir. (OR, AND, XOR gibi)

Graphics

Bilgisayarın ekranında, karakterlerin dışında olarak, çizgi, daire vb. gibi grafiklerin gösterilmesi işlemi.

Graphics character

Bir görüntüyü yaratmak için özel olarak dizayn edilmiş şekil.

Graphics cursor

Metin kursörüne benzer fakat grafik ekranını belirler. 6128'in farkedilmeyen fakat çizim yapmak için zorunlu olan bir unsurudur. Karakter setinin bir parçası olan ve metin kursörünün bulunduğu yere yazılan grafik karakterleri ile karıştırmamak gerekir.

Graphics mode

Eski bilgisayarların karakter yazmak veya grafik çizmek için ayarlanması gerekiyordu. Modern kişisel bilgisayarlar ise bu iki işlemi de aynı anda yapabilirler.

Graphics tablet

Verilen bir resim veya çizimi sayısal hale çevirip bilgisayara veren alet.

Handshaking

Bir bilgisayar ile diğer bir bilgisayar veya çevre bimi arasındaki bilgi alış verişini düzenleyen elektronik sinyaller dizisi.

Hard Copy

Bir program, metin veya grafiğin kağıda basılmış şekli. Ekranda gösterim ise 'soft copy' olarak bilinir.

Hardware

Bilgisayarı oluşturan fiziksel birimlerin tümüne verilen genel ad.

Hexadecimal (or HEX)

Onaltılı rakam kullanılan sayı sistemi. 6128 programlamasında veya ön eki ile gösterilirler. Örneğin, FF = 255 (ondalık)

Hex file

Bir komut veya makina kodu dosyasının ASCII ile temsili

High-level language

Bilgisayarları programlamak için makine dili yerine kullanılan ve insan dillerine daha yakın olan programlama dilleridir. Makine kodu merkezli dillere göre daha yavaş çalışırlar fakat anlaşılmaları daha kolaydır.

IEEE-48

Mikro bilgisayarları diğer birimlere bağlamak için kullanılan standart arabirimlerden biri. Centronics paralel ara birimine benzer fakat tamamen aynı değildir.

Information tecnhology

Bilgi işleme ve haberleşme alanında kullanılan her türlü elektronik birimler: kelime işleme, veri iletimi vb.

Initialise

Sistem ya da programın başlangıç durumuna getirilmesi: değişkenlerin belirlenmesi, dizilerin açılması gibi.

Input

Her türlü veri ya da bilgilerin, klavye, disk, ara birim veya değişik kaynaklardan bilgisayar sistemine girilmesi.

Instruction

Bilgisayarın belli bir işlemi yapması için verilen emir. Bir program bir dizi komuttan meydana gelir.

Instruction set

Mikro işlemcinin gerçekleştirebileceği komutlar grubu yüksek seviyeli tüm komutlar, işleme konmadan önce CPU'nun anlayabileceği komutlara dönüştürülmelidir. Bazen tek bir yüksek seviyeli komut birçok sayıda CPU komutu ile ifade edilebilir.

Integer

Ondalıksız sayılar tam sayılar

Integer number

Ondalık kısmı olmayan sayı. Tamsayı ve ondalık kısımdan oluşan reel sayıların tersine sadece tamsayıdan oluşurlar.

Integrated circuit

Birleştirilmiş ve bir silikon parçası üzerine yerleştirilmiş elektronik devreler grubu (Bkz. Chip)

İntelligent terminal

Veri giriş çıkışı yanında, ana sistemle bağlantısı kesildiğinde işlem yapabilen terminal

Interactive

Bilgisayar sisteminin ya da programın kullanıcı ile soru cevaplı bir şekilde çalışması. Hareketli bir oyunda bir uzay gemisinin kontrolu veya bir eğitim programında soruların cevaplanması gibi. Kullanıcının yaptıkları programın çalışma zamanlamasını etkiler.

Interface

Bilgisayarın diğer araçlar veya insan ile haberleşmesini sağlayan yol. 6128'in ara brimi klavye (giriş) ve ekran (çıkış) ve de diğer donanım için bağlantı soketlerinden oluşur.

Interpreter

Sistem yazılarının, yüksek seviyeli bir dilde yazılmış olan komutları, CPU'nun anlayabileceği seviyeye indiren bölümü. Örneğin: klavyeden girilen BASIC'i bilgisayarı kendi diline çevirir.

I/O

Giriş/Çıkış

Iteration

Bilgisayar her işi, CPU tarafından yerine getirebilecek parçalara bölerek yapar. Bunun içinde bir koşul gerçekleşene kadar birçok basit eleman arasında gidip gelmesi gerekir.

Joystick

Genellikle kursör tuşların vazifesini gören bir alet. Bilgisayar oyunlarının daha hızlı ve kolay oynanmasını sağlar.

K

Kilo ön ekinin kısaltılmış. Bilgisayarda 'kilobyte'ı temsil eder. Gerçekte 1024 (2 üzeri 10) bayttır.

Keyboard

Üzerindeki tuşlara basıldığında bilgisayarın kullandığı karakteri üreten aygıt.

Keyword

Bilgisayar programında veya dilinde belli bir komut veya fonksiyonu ifade eden kelime.

Least significant bit

İkili bir sayıda en sağındaki bit (LSB).

Light Pen

Özel bir kalem kullanan giriş yöntemi

Line number

BASIC ve bazı diğer dillerde program satırları, satır numaraları ile dizilirler.

Lisp

LIST Processor Language kelimelerinin kısa yazılışı. Yüksek seviyeli bir bilgisayar dili.

Logic

Bilgisayarın tüm işleyişini oluşturan elemanter mantıksal işlem ve fonksiyonları yerine getiren elektronik birimler.

Logical device

Bir cihazın tanımı kendi fiziksel formundan başka olabilir. Örneğin :CP/M Mantık cihazı LST bir Centronics çıkış birimide olabilir ya da bir monitördür.

LOGO

İsmi Yunanca logos kelimesinden alınan bir programlama dili. Bilgisayar programlamanın esaslarını öğretmek amacı ile geliştirilmiştir.

Loop

Belirlenen koşul sağlanıncaya kadar program konularının tekrar edilmesi.

Low-level language

Assembly dili gibi, her komutun makina kodu komutlarına karşılık olduğu program dili.

LSI

Large Scale Integration kısa adı. Daha küçük alanlara daha çok fonksiyon yerleştirilecek entegre devrelerin geliştirilmesi.

Machine Code

Mikro işlemci tarafından direkt olarak anlaşılan programlama dili.

Machine redable

Herhangi bir işlem gerekmeden bilgisayara girilebilecek veri ortamı.

Man-machine interface

Bilgisayar ve kullanıcı arasında iletişimi sağlayan klavye, ekran, ses gibi araçlar.

Matrix

Ekranda veya 'dot matrix' yazıcının yazma kafasında karakteri oluşturan noktalar grubu. Matematikçe ve bilgisayar biriminde dizileri belirtmek için de kullanılır.

Memory

Bilgisayarda bilginin sistemli ve istenilen bilgiye kolayca erişebilecek şekilde saklandığı yer. Hem okunması hem de yazılması mümkün olan RAM ve sadece okunabilen ROM şekillerinden oluşur. Disk ve bant da CPU tarafından ulaşılması mümkün oldukları için bir çeşit hafıza olarak kabul edilirler.

Memory map

Bilgisayar hafızasında hangi adreslerde hangi fonksiyonların bulunduğunu gösterir plan.

Menü

Programın kullanışındaki tercihlerin bulunduğu liste.

Microprocessor

Bilgisayarın beynini oluşturan ve tüm merkezi işlemlerin yapıldığı entegre devre.

Modem

Modüler DEModulatör'ün kısa adı. Bilgisayarın giriş ve çıkışlarının telefon hattı veya optik lif gibi ortamlarla yapılmasını sağlayan aygıt. (Bkz. Acoustic coupler)

Monitör

Bir bilgisayar terminalinin ekranı. Aynı zamanda bilgisayarın bazı temel işlevlerine ulaşmayı mümkün kılan makina kodu program.

Mouse

Her yönde haraket edebilen ve bir zemin üzerinde yürütülerek ekranda kursörü hareket ettirmeye yarayan aygıt. Program kullanımını kolaylaştırmak için geliştirilmiştir.

MSB

Most significant Bit'in kısa adı. İkili bir sayının en soldaki bit'i

Network

İki veya daha çok sayıda bilgisayarın çeşitli yollarla bağlanarak birlikte çalışır halde olması.

Nibble

Bir kayıtın yarısı. Dört bitlik bir ifade. F6 ifadesindeki F ve 6 onaltılı rakamlarının herbiri bir nibbledir

Node

LOGO dilinde bir hafıza birimidir. Bir nod genellikle dört bitten oluşur.

Noise

6128'in her türlü ses özellikleri.

Numeric keypad

Klavyede, nümerik veri girmek için kullanılan tuşlar grubunun bulunduğu kısım. 6128'de bu tuşlar tanımlanabilir fonksiyon tuşları olarak da kullanılırlar.

OCR

Optical Character Recognition kelimelerinin kısa yazılış. Yazılı karakterlerin optik olarak okunup, bilgisayarca anlaşılabilir veri haline getirilmesi.

Octal

Her rakamın 3 bit ile ifade edildiği 8 tabanlı sayı sistemi.

Off line

Ekran veya yazıcı gibi bir bilgisayar aksamının ana işlem birimi ile bağlantılı olmaması hali.

On line

Off line'ın tersi.

Operating system

Bilgisayarın işlemlerini kontrol eden yazılım.

Operatör

Sayılar ile yapılacak işlemleri belirleyen ifadeler. (+,-,*,/ gibi)

Output

Bir bilgisayardan hesaplamalar sonunda çıkan her türlü sonuç.

Owerwrite

Hafızada bir bölgenin, üzerine yeni veri yazılarak silinmesi.

Paddle

Oyun çubuğunun bir başka adı.

Page zero

CP/M ortamında, hafızanın sistem değişkenlerini saklayan ve 0000 ile 0100 adresleri arasında kalan kısım.

Paperware

Hardcopy'nin diğer bir anlatımı. Bazen geliştirilmesi tamamlanmadan piyasaya sürülen bilgisayarlarda, paperware exercise olarak tabir edilirler.

Parallel interface

61298'in yazıcı ara birimi, paralel bir yazıcıyı besler. Bu bilgisayardan verilen her veri satırının yazıcı tarafından direkt olarak alınması demektir. Paralel ara birim ile veri transferi, seri ara birim ile yapılandan daha hızlıdır. Çünkü seri ara birim önce her bir bayt'ı transfer için uygun hale getirmeli ve aynı zamanda senkronizasyonu da sağlamalıdır.

Pascal

Yüksek düzeyli, çalışmadan önce derlenmesi gereken ve bu yüzden çok hızlı olan bir programlama dili. Genellikle BASIC'den sonra öğrenilen ilk dildir.

PEEK

Bilgisayarın belleğinde istenilen bir yeri okumak için kullanılan BASIC fonksiyonudur.

Peripheral

Bilgisayarın yeteneklerini geliştirmek amacı ile bağlanan her türlü birim. Yazıcı, modem, oyun çubuğu, kaset birimi gibi.

Physical device

Fiziksel devresi olan gerçek bir cihaz. Bu tip cihazlar mantıksal cihazlarla da tanımlanabilirler.

Pixel

Ekran üzerinde, bilgisayar tarafından kontrol edilebilen en küçük alan.

Plotter

Yazıcı kafa yerine, kalem kullanarak gerçek anlamda çizebilen yazıcı tipi. Teknik ve grafik çizimler için kullanılır.

POKE

Bellekte belirtilen bir yere istenilen bir değeri vermek için kullanılan komut.

Port

Ara birim üzerinde veri giriş çıkışı için özel olarak adreslenebilir nokta.

Portability

Genel anlamı dışında bir yazılımın değişik bilgisayarlarda çalışabilmesini ifade eder. Bu genellikle ortak bir işletim sistemi (CP/M gibi) sayesinde mümkün olur.

Primitives

Dr. LOGO'yu meydana getiren işlem veya komutlar.

Printer

Metin yazmak için kullanılan her türlü hardcopy metodu.

Procedure

Belli bir işlemin nasıl yapılacağını belirten ifadeler veya program komutları serisi.

Program

Bilgisayarın istenilen işi yapması için gerekli olan komutlar kombinasyonu. Basit bir makina kodu rutininden kelime işlemci gibi kompleks bir programa kadar her türlü yazılımı kapsar.

Programming language

Bir programın yazılabilmesi için gerekli olan ve katı kurallara sahip ortam.

PROM

Programmable Read Only Memory kelimelerinin kısa yazılışı. Üzerine veri yazıldıktan sonra bir daha silinmesi mümkün olmayan entegre hafıza devresi. (EPROM'a bakınız)

Prompt

Kullanıcıya, bir girdi beklendiğini gösteren kısa mesaj veya karakter. Örneğin CP/M'de , Dr. LOGO'da ? karakteri.

PSU

Power Supply Unit kelimelerinin kısa yazılışı. Normal şebeke ceryanını bilgisayar ve çevre birimleri için gerekli voltaja ayarlayan elektronik devre.

QWERTY keyboard

İngiliz veya Amerikan daktilo klavyesini ifade eden terim.

RAM

Ramdom Access Memory kelimelerinin kısa yazılışı. Programın normal seyri sırasında, bilgisayarın dahili donanımı kullanılarak hem okunması ve hem de yazılması mümkün olan bellek.

Random access

Belleğe veya disk üzerine istenilen şekilde bilgi yazıp okuyabilme kabiliyeti.

Random number

Bilgisayar programı tarafından rastgele olarak üretilen ve ne tekrarı ne de tahmin edilmesi mümkün olan sayı. 6128 bir sahte rastgele sayı serisi üretebilir.

Raster

Ekranın yatay çizgiler halinde taranması yöntemi ile görüntü oluşturulan bir yazma sistemi (Raster scan)

Read only R/O

Bir disk, dosya veya sürücünün veri yazma veya değiştirme fonksiyonların/önlemek için kullanılan sıfat.

Read write R/W

Bir disk, dosya veya sürücünün veri okuma ve yazmasının serbest olduğunu ifade eden sıfat.

Real number

Tamsayı ve ondalık kısımdan mevcut sayı.

Real time

Gerçek zaman içinde programın çalışması.

Record (Kayıt)

Bir dosyadaki baytlar grubu. CP/M, 128 baytlık kayıtlar kullanır.

Recursion (Tekrarlayan)

Her seferinde sonucun bir öncekine bağlı olduğu program adımları tekrarı serisi.

Refresh (Taneleme)

Bilgiyi hafıza ya da ekranda son hale getirme.

Register (Kayıt)

CPV'da geçici saklama için kullanılan hafıza bölümü.

Remark (İşaret)

Programın çalışmasında herhangi bir etkisi olmayan, sadece programcıya hatırlatmalarda bulunmak veya program hakkında bilgi vermek için programın içerisine yerleştirilen ifadeler.

Reserved word (Ayırılmış kelime)

Daha önceden verilmiş bir anlamı olan ve başka bir iş için kullanılması mümkün olmayan kelime. Örneğin BASIC'te NEW sözcüğü bir değişken adı olarak kullanılmaz, başka bir amaç için ayrılmıştır.

Reselotuon (Çözünüm)

Bir görüntü elemanının bitip diğerinin başladığı yeri tayin etme kabiliyeti. Aynı zamanda bilgisayarın büyük sayılarla aritmetik işlemler yapabilme kabiliyetinide ifade için kullanılır.

Reserve Polish notation

(RPN) operatörlerinin (+ ,-,*,/) ait oldukları değerlerden sonra yazıldığı aritmetik ifade metodu.

RF Modülatör

Bilgisayarın görüntü sinyallerini standart TV tarafından kullanılabilecek hale getiren araç.

ROM

Read Only Memory kelimesinin kısa yazılışı. Bir kere yazıldıktan sonra silinmesi veya değiştirilmesi mümkün olmayan, sadece okunabilen, genellikle yarı iletken hafıza sistemi.

Routine (Rutin)

Bir iş gören program bölümüdür. Bir ana programın içerisinde, veya değişik programlar ile kullanılan ayrı bir modül halinde bulunan alt program. Örneğin sistem saatinden, gerçek bir saat görüntüsü oluşturan program bir rutin sayılabilir.

RS232C

Seri, veri alış verişi için bir standarttır. RS232C kullanan veri hattının iki ucundaki cihazın da RS232C standartlarında olması gerekir. Bunu her yerde standart olan Centronics paralel ara birimi ile karşılaştırınız.

Screen editor (Ekran editörü)

Ekrandaki metin veya program üzerinde kursörle istenilen yere gidilerek değişiklik yapılmasını sağlayan editör.

Scrolling (Kaydırma)

Ekran dolup, yeni yazılacak satır için yer açmak amacı ile ekrandaki görüntünün yukarıya kayması.

Sector (Dilim)

Disk üzerinde bir veri bloğu. AMSTRAD disk sistemi 512 bayt'lık dilimler kullanır.

Separatör (Ayırıcı)

Sınırlayıcı ile aynı işi görür. Ayrılmış kelimeler ile programın diğer bölümleri veya veriler arasındaki sınırı işaretler.

Serial interface (Seri ara birim)

Bu terim RS232 ara birimine çok yakın olduğu halde bu alanda başka standartlar da vardır.

Simulation (Taklit)

Bilgisayar kullanarak gerçek olayların taklit edilmesi. Araba veya uçak kullanmak gibi.

Single sided (Tek taraflı)

Sadece bir yüzünde veri saklanabilen disk.

Soft key

(UDK tanımlanabilir karakterlere bakınız.)

Software (Yazılım)

Programlar disk, kaset, ROM veya diğer esaslı olabilirler.

Software engineering (Yazılım mühendisliği)

Sistemli ve planlı olarak, bilgisayar programlamayı ifade eden terim.

Sound generatör (Ses üreteci)

Bilgisayarın ses üreten kısmı. (Yazılım veya donanım olabilir)

Speech recognition (Konuşma algılaması)

Konuşulan kelimelerin makina tarafından anlaşılabilir veriler haline çevrilmesi.

Speech synthesis (Konuşma sentezi)

Yazılım ve donanım kullanılarak insan sesinin ve konuşmanın taklit edilmesi.

Spreadsheet (Çalışma sayfası)

Satır ve sütunlar halinde sayılar girilmesini ve bunların aritmetik işlemlerinin yapılmasını sağlayan program. Bir verinin değiştirilmesi, ilgili tüm hesapların tekrar edilmesi ve yeni durumun üretilmesine sebep olur.

Sprite (Hareketli ekran)

Özel bir donanım veya yazılım tarafından oluşturulan, ekranda serbestçe hareket eden ve gelişigüzel görünüp kaybolan bir karakter.

Stack (Küme)

Hafızada bilgi istifi için kullanılan bölge. Fakat sadece son girilen bilgi geri çağrılabilir.

Statement (İfade)

Bir bilgisayar programında emir veya emirler serisi.

Stream (Akım)

Bilgisayardan çıktı almak için kullanılan yollar. (Ekran, yazıcı veya disk gibi.)

String (Dizi)

Bir karakterler dizisinde oluşan ve sayısal bir değişken gibi işlem görmesi mümkün olmayan veri tipi. Tamamen sayısal olabilir fakat gerekli komut ile çevirim yapılmadıkça bir sayı gibi kullanılamaz.

Structured programming ( )

Yukarıdan aşağıya kolay anlaşılabilir adımlar halinde akan programlar yazmada kullanılan, mantık ve tasarlama gerektiren programlama tekniği.

Sub-routine (Alt-rutin)

(Rutin'e bakınız.)

Syntax error (Yazılım hatası)

Eğer programın kuralları, anahtar kelimeler veya değişkenlerin yanlış kullanımları ile ihlal edilirse, BASIC kullanıcıya bu mesajı verir.

System tracks (Sistem izleri)

Disk üzerine CP/M için ayrılmış olan izler.

Terminal (Terminal)

Bir klavye giriş birimi ile VDU ekranı veya teletip daktilo çıkış biriminden oluşan sistem.

TPA

Transient Program Area kelimelerinin kısa yazılışı. Belekte Ç0100'den sonra gelen ve CP/M kullanıcı programlarının çalıştığı ve veri sakladığı alan.

Track (İz)

Disk üzerinde müşterek merkezli dairelerdir. Her iz belli bir sayıda dilimi tutar. Formatlama sırasında iz ve dilimler disk üzerinde belli yerlere yazılır.

Transient program (Geçici program)

İsmi klavye ile yazılarak TPA'ya yüklenen ve çalışan CP/M kullanım programı. (PIP gibi)

Truncated (Kesik)

Bir kısmının ayrılması sonucunda kısaltılmış sayı veya dizi. İstenerek bir sayının yuvarlanması veya istenmeyerek bir dizinin yazılacak yere sığması için kısaltılması şeklinde olabilir.

Truth table (Doğruluk tablosu)

Bir mantıksal ifadenin sonucu ya 'doğru' ya da 'yanlış'tır. Bilgisayar bunları 1 veya 0 olarak yorumlar. Doğruluk tablosu bir mantıksal izlerin mümkün olan hallerinin listesidir.

Turnkey

Sistem yüklendiğinde otomatik olarak çalışan program anlamında bir söz.

Turtle

Dr. Logo grafik ekranı üzerindeki ok ve şeklindeki grafik kursörü.

Turtle graphics

'Turtle'ın ekran üzerinde hareketi sonucunda oluşan grafik. Turtle hareket ederken arkasına iz bırakır.

Turtle step

'Turtle'nin hareket edebileceği en kısa mesafe. Genellikle bir pixel.

UDK

User defined keys kelimesinin kıza yazılışı. 6128'de 32'ye kadar tuş kullanıcı tarafından herhangi bir amaç için tanımlanabilir.

Unsigned number (İşaretsiz sayı)

Önünde pozitif veya negatif olduğunu belirten bir işaret bulunmayan sayı.

Utility (Kullanım)

Veri sıralama veya dosya kopyalama gibi sık tekrarlanan işlemleri yapmak için kullanılan her türlü program.

Utility program (Kullanım programı)

Kullanıcının belli işlemleri yapmasını sağlayan disk programı (Transient program).

Variable (Değişken)

Bir isim ile ifade edilen ve değeri programın çalışması sırasında değişebilen sayı veya dizi.

Warm start

CP/M'de (CTRL)C'ye basılması ile sağlanır. Disk alt sistemini sıfırlar ve kontrolu yeni bir komut almak üzere CP/M'e geri verir.

Wildcard character

* ve ? karakterlerinin herhangi biri. Dr. Logo sadece ?'i kabul eder. * herhangi bir sayıda ?'i gösterir. Belirsiz dosya adlarını ifadede kullanılır. Dosya adındaki bir ? her türlü karakteri kapsar.

Write protection (Yazma koruması)

Bir disk veya disk dosyası üzerine veri yazılmasını önler. Yazma korumalı bir disk veya dosya sadece okunmak içindir.

XYZY

Serüven oyunlarında zor durumdan kurtulmaya yarayan sihirli bir sözcük.

EK 3
Sizin İçin Bazı Programlar

Bustout

Bir oyuncu bilgisayara karşı. Klayve ve Joystick.

```
10 'BUSTOUT by ALEXANDER MARTIN
20 'copyright (c) AMSOFT 1984
30 '
40 MODE 1:BORDER 1:INK 0,1:INK 1,26:INK 2,24:INK 3,6
50 SPEED KEY 15,2
60 ENV 1,1,18,0,11,0,10
70 ENT 1,10,2,2
80 ENV 3,1,0,16,5,-3,2
90 ENV 2,5,3,3,1,-21,22,9,-3,2
100 ENT -2,10,2,2,5,-7,1,2,11,3,2,-4,8
110 '
120 '
130 MOVE 30,32:DRAWR 0,400,1:MOVE 610,32:DRAWR 0,400,1
140 PEN 3:LOCATE 3,1:PRINT STRING$(36,143)
150 PEN 2:LOCATE 3,2:PRINT STRING$(36,143)
160 PEN 1:FOR r=5 TO 6:LOCATE 3,r:PRINT STRING$(36,143)
    :NEXT r
170 bx=9
180 lives=5:score=0
190 PEN 1:GOSUB 680:CLEAR INPUT
200 IF INKEY$<>CHR$(32) AND JOY(0)<16 THEN 200
210 LOCATE 11,23:PRINT SPACE$(20):LOCATE 1,24:PRINT SPA
    CE$(40);
220 GOSUB 690:GOSUB 660:GOTO 280
230 '
240 '
250 LOCATE bx,24:PRINT"  ";STRING$(4,131);"  ":RETURN
260 '
270 '
280 xa=1:ya=1:IF INT(RND*2)=1 THEN xa=-xa
290 PEN 1:GOSUB 250
300 ORIGIN 0,400
310 x=bx+4:y=11:x1=x:y1=y
320 '
330 '
340 x1=x+xa:y1=y+ya
```

```
350 IF x1=3 OR x1=38 THEN xa=-xa
360 GOSUB 540
370 IF y1=24 AND x1>bx+1 AND x1<bx+6 THEN ya=-ya:y1=y1-
    2:SOUND 130,44,8,7,1,1:a=((x>bx+5)OR(x<bx+2)):IF a=
    -1 THEN xa=xa*a:x1=x1+xa:y1=y1+1
380 IF y1=25 THEN LOCATE x,y:PRINT" ":GOTO 500
390 GOSUB 250
400 t=TEST((16*x1)-1,-(16*y1)-1)
410 IF t<>0 THEN ya=-ya:xz=x1:yz=y1:y1=y1+ya:GOSUB 590:
    IF t=2 THEN score=score+10:GOSUB 660
420 IF t=3 THEN score=score+20:GOSUB 660
430 IF t=1 THEN score=score+5:GOSUB 660
440 IF y1=1 THEN ya=1
450 LOCATE x,y:PRINT " ":LOCATE x1,y1:PRINT CHR$(233):x
    =x1:y=y1
460 IF y=1 OR x=3 OR x=38 THEN SOUND 129,78,8,7,1,1
470 GOTO 340
480 '
490 '
500 lives=lives-1:SOUND 132,19,46,12,2,2:IF lives=0 THE
    N GOTO 620
510 GOSUB 660:GOTO 280
520 '
530 '
540 IF (INKEY(8)=0 OR INKEY(74)=0) AND bx>2 THEN bx=bx-
    2:RETURN
550 IF (INKEY(1)=0 OR INKEY(75)=0) AND bx<32 THEN bx=bx
    +2:RETURN
560 RETURN
570 '
580 '
590 LOCATE xz,yz:PRINT " ":RETURN
600 '
610 '
620 IF score>=hiscore THEN hiscore=score
630 GOSUB 660:score=0:lives=5:GOTO 130
640 '
650 '
660 SOUND 130,0,20,13,3,0,31:LOCATE 1,25:PRINT TAB(4)"H
    ISCORE";hiscore;
670 LOCATE 18,25:PRINT"SCORE";score:LOCATE 30,25:PRINT"
    LIVES";lives:RETURN
680 LOCATE 11,23:PRINT"PRESS SPACE TO START":RETURN
690 LOCATE 1,25:PRINT SPACE$(40);:RETURN
```

Bomber

Bir oyuncu bilgisayara karşı. Yalnız klavye.

```
10 'BOMBER by DAVE TOWN
20 'copyright (c) AMSOFT 1984
30 '
40 MODE 1:CLS:INK 0,0:BORDER 0:INK 1,18:INK 2,6:INK 3,4
   :INK 5,15:INK 6,2:INK 7,24:INK 8,8:INK 9,26:INK 10,1
   0:INK 11,20:INK 12,12:INK 13,16:INK 14,14:INK 15,21
50 SYMBOL AFTER 240:SYMBOL 241,&40,&60,&70,&7F,&7F,&3F,
   &7,&0:SYMBOL 242,&0,&32,&7A,&FE,&FA,&F2,&E0,&0
60 score=0:hiscore=0:plane$=CHR$(241)+CHR$(242):x=2:y=2
   :drop=0:a=2:b=2
70 GOSUB 480
80 CLS
90 PEN 2:LOCATE 1,15:INPUT"Enter skill : 0 (ACE) to 5 (
   NOVICE) : ",skill
100 IF skill<0 OR skill>5 GOTO 90
110 skill=skill+10
120 LOCATE 1,15:PRINT CHR$(18);:LOCATE 1,15:INPUT"Enter
     speed 0 (FAST) to 100 (SLOW) : ",rate
130 IF rate>100 OR rate<0 GOTO 120
140 '
150 'Buildings
160 '
170 MODE 0:FOR base=5 TO 15:FOR height=21 TO INT(RND(1)
    *8+skill) STEP -1:LOCATE base,height:PEN base-2:PRI
    NT CHR$(143)+CHR$(8)+CHR$(11)+CHR$(244);:NEXT :NEXT
180 PLOT 0,20,4:DRAW 640,20,4
190 LOCATE 1,25:PEN 2:PRINT"SCORE";score;:LOCATE 13,25:
    PRINT"HI";hiscore;
200 '
210 'Main Game
220 '
230 LOCATE x-1,y:PRINT"   ";
240 PEN 1:LOCATE x,y:PRINT plane$;:PEN 2
250 IF y=21 AND x=15 THEN GOTO 290:ELSE GOTO 340
260 '
270 'Landed
280 '
290 FOR c=0 TO 1000:NEXT
```

```
300 score=score+100-(skill*2):skill=skill-1:x=2:y=2:a=2
    :b=2:drop=0
310 IF skill<10 THEN skill=10:rate=rate-20
320 IF rate<0 THEN rate=0
330 GOTO 150
340 FOR c=0 TO rate:NEXT
350 x=x+1
360 IF x=18 THEN LOCATE x-1,y:PRINT CHR$(18);:x=2:y=y+1
    :LOCATE x,y:PEN 1:PRINT plane$;:PEN 2
370 a$=INKEY$:IF a$=" " AND drop=0 THEN drop=1:b=y+2:a=
    x
380 IF y=21 THEN drop=0
390 IF drop=1 THEN LOCATE a,b:PRINT CHR$(252);:LOCATE a
    ,b-1:PRINT" ";:b=b+1:IF b>21 THEN LOCATE a,b:PRINT"
     ";:LOCATE a,b-1:PRINT" ";:a=0:b=0:drop=0:SOUND 3,4
    000,10,12,0,0,10
400 ga=(a-0.5)*32:gb=400-(b*16):bomb=TEST(ga,gb)
410 IF bomb>0 THEN GOTO 670
420 gx=((x+1.5)*32):gy=408-(y*16):crash=TEST(gx,gy)
430 IF crash>0 GOTO 570
440 GOTO 230
450 '
460 'Instructions
470 '
480 LOCATE 1,2:PEN 1:PRINT"You are piloting an aircraft
     over a des-erted city and must clear the buildings
     in order to land and refuel.  Your air- craft move
    s across  the screen from left to right.";:PRINT
490 PRINT:PRINT"On reaching the right, the aircraft
     returns to the left A LINE FURTHER DOWN.You have a
    n unlimited supply of bombs  and you can drop them
     on the buildings  below by pressing the SPACE BAR.
    ";:PRINT
500 PRINT:PRINT"Each time you land, the height of the
     buildings or the speed of your aircraft increases.
    ";:PRINT:PRINT:PRINT"ONCE YOU HAVE RELEASED A BOMB,
     YOU WILL NOT BE ABLE TO RELEASE ANOTHER UNTIL THE
    FIRST HAS EXPLODED !!!!";
510 PEN 2:LOCATE 1,24:PRINT:PRINT"Press any key to star
    t.";
520 a$=INKEY$:IF a$="" GOTO 520
530 RETURN
540 '
550 'Collision
```

```
560 '
570 LOCATE x-1,y:PRINT CHR$(32)+CHR$(32)+CHR$(32)+CHR$(
    253)+CHR$(8)+CHR$(238)+CHR$(8);
580 FOR t=1 TO 10:SOUND 7,4000,5,15,0,0,5:PEN t:PRINT C
    HR$(253)+CHR$(8)+CHR$(238)+CHR$(8)+CHR$(32)+CHR$(8)
    ;:FOR tm=0 TO 50:NEXT:NEXT:PEN 2
590 CLS:LOCATE 1,5:PRINT"You scored";score;
600 IF score>hiscore THEN hiscore=score:LOCATE 1,8:PRIN
    T"TOP SCORE!!";
610 score=0:LOCATE 1,12:PRINT"Press R to restart";
620 a$=INKEY$:IF a$="r" OR a$="R" GOTO 630 ELSE GOTO 62
    0
630 PEN 1:MODE 1:x=2:y=2:a=2:b=2:GOTO 90
640 '
650 'Bombed building
660 '
670 LOCATE a,b-1:PRINT" "+CHR$(8);:PEN 4:FOR tr=1 TO IN
    T(RND(1)*3)+1:score=score+5:SOUND 3,4000,10,12,0,0,
    10:LOCATE a,b:FOR t=0 TO 4:PRINT CHR$(253)+CHR$(8)+
    CHR$(32)+CHR$(8);:NEXT:b=b+1
680 IF b=24 THEN b=b-1
690 NEXT
700 LOCATE 6,25:PRINT score;:drop=0:a=x:b=y:GOTO 230
```

Telley tennis

Bir veya iki oyuncu bilgisayara karşı. Klavye veya Joystick.

```
10 'TELLY TENNIS by DAVID RADISIC
20 'copyright (c) AMSOFT 1985
30 '
40 DEFINT a-z
50 comp=1
60 ENV 1,=11,20,=9,5000
70 MODE 1:INK 0,10:BORDER 10:INK 1,26:INK 2,18:INK 3,0
80 GOSUB 710
90 GOSUB 150
100 GOSUB 330
110 GOSUB 420
120 LOCATE 13,1:PRINT USING"#### ";score1;
130 LOCATE 35,1:PRINT USING"#### ";score2;
140 GOTO 100
150 PEN 2
160 x(1)=3:y(1)=5
170 x(2)=37:y(2)=22
180 edge$=CHR$(233):edge2$=STRING$(2,207)
190 LOCATE 1,3:PRINT STRING$(39,edge$):PRINT STRING$(39
    ,edge$)
200 FOR i=1 TO 19
210 PRINT edge2$;TAB(38);edge2$
220 NEXT
230 PRINT STRING$(39,edge$):PRINT STRING$(39,edge$);
240 WINDOW #1,3,37,5,23
250 CLS#1
260 SYMBOL 240,0,60,126,126,126,126,60,0
270 bat$="|"+CHR$(8)+CHR$(10)+"|"
280 clr$=" "+CHR$(8)+CHR$(10)+" "
290 ball$=CHR$(240)
300 PEN 3
310 LOCATE 2,1:PRINT"Player 1 :    0";:LOCATE 24,1:PRIN
    T"Player 2 :    0";
320 RETURN
330 n=INT(RND*2):CLS #1:scored=0
340 PEN 3
350 FOR i=1 TO 2:LOCATE x(i),y(i):PRINT bat$;:NEXT
360 ON n GOTO 390
```

```
370 xb=21:dx=1
380 GOTO 400
390 xb=19:dx=-1
400 yb=12:dy=INT(RND*3)-1
410 RETURN
420 GOSUB 600
430 oxb=xb:oyb=yb
440 GOSUB 500
450 IF note>0 THEN SOUND 129,note,50,15,1
460 LOCATE oxb,oyb:PRINT" ";
470 LOCATE xb,yb:PRINT ball$
480 IF scored=0 THEN 420
490 RETURN
500 LOCATE xb+dx,yb+dy:ch$=COPYCHR$(#0)
510 note=0
520 IF ch$=" " THEN xb=xb+dx:yb=yb+dy:RETURN
530 IF ch$="|" THEN dx=2-dx-2:dy=INT(RND*3)-1:note=200:
    RETURN
540 IF ch$=LEFT$(edge2$,1) THEN 570
550 IF ch$=edge$ THEN dy=2-dy-2:note=250
560 RETURN
570 IF dx>0 THEN score1=score1+1 ELSE score2=score2+1
580 scored=1:note=2000
590 RETURN
600 p(1)=(INKEY(69)>=0)+(INKEY(72)>=0)+ABS((INKEY(71)>=
    0)+(INKEY(73)>=0))*2
610 IF comp=1 THEN p(2)=ABS(y(2)<yb)*2+(y(2)>yb):GOTO 6
    30
620 p(2)=(INKEY(4)>=0)+(INKEY(48)>=0)+ABS((INKEY(5)>=0)
    +(INKEY(49)>=0))*2
630 PEN 3
640 FOR i=1 TO 2
650 LOCATE x(i),y(i)+p(i):ch$=COPYCHR$(#0)
660 IF ch$=" " THEN LOCATE x(i),y(i):PRINT clr$;:y(i)=y
    (i)+ROUND(p(i)/2)
670 LOCATE x(i),y(i):PRINT bat$;
680 NEXT
690 PEN 1
700 RETURN
710 PEN 2:PRINT:PRINT TAB(15)"Ping-Pong":PRINT TAB(15)"
    ---------"
720 PEN 3:PRINT:PRINT TAB(14)"To move bats :
730 PRINT:PRINT:PEN 1
740 PRINT" Player 1       Player 2       Direction":PRI
    NT
```

```
750 PRINT"     A                6                UP"
760 PRINT"     Z                3                DOWN":PRINT
770 PEN 3:PRINT:PRINT TAB(14)"Or joysticks"
780 PRINT:PRINT:PRINT:PRINT
790 PEN 2
800 PRINT TAB(6)"Select <1> or <2> players"
810 i$=INKEY$:IF i$<>"1" AND i$<>"2" THEN 810
820 IF i$="1" THEN comp=1 ELSE comp=0
830 MODE 1:RETURN
```

Electric Fencing

Yalnız iki oyuncu. Klavye ve Joystick

```
10 'ELECTRIC FENCING by ALEXANDER MARTIN
20 'copyright (c) AMSOFT 1985
30 '
40 DEFINT a-z
50 MODE 0
60 GOSUB 980
70 GOSUB 1370
80 GOSUB 270
90 GOSUB 1520
100 GOSUB 1370
110 GOSUB 1270
120 '
130 '
140 REM start
150 IF finished THEN GOTO 100
160 GOSUB 240
170 FRAME:IF p1dir THEN GOSUB 570 ELSE FRAME:FRAME
180 FRAME:IF p2dir THEN GOSUB 620 ELSE FRAME:FRAME
190 IF p1sa=-1 THEN GOSUB 670
200 IF p2sa=-1 THEN GOSUB 720
210 GOTO 140
220 '
230 '
240 IF j THEN  380 ELSE  480
250 '
260 '
270 CLS:PEN 6
280 PRINT:PRINT"   CHOOSE CONTROL"
290 PRINT:PRINT:PRINT:PRINT"press J/K then ENTER"
300 LOCATE 4,10:PRINT"JOYSTICK";TAB(5);"OR KEYS"
310 LOCATE 12,10:IF j THEN PRINT"*":ELSE PRINT" "
320 LOCATE 12,11:IF j THEN PRINT" ":ELSE PRINT"*"
330 IF NOT(INKEY(45)) THEN j=-1
340 IF NOT(INKEY(37)) THEN j=0
350 IF NOT(INKEY(18)) THEN RETURN ELSE 310
360 '
370 '
380 p1=JOY(0):p2=JOY(1)
390 p1dir=(p1 AND 1)*-1+(p1 AND 2)*0.5
```

```
400 p2dir=(p2 AND 1)*-1+(p2 AND 2)*0.5
410 IF P1 AND 16 THEN p1sa=p1sa-1:IF p1sa=-1 THEN AFTER
     15 GOSUB 770
420 IF P2 AND 16 THEN p2sa=p2sa-1:IF p2sa=-1 THEN AFTER
     15 GOSUB 770
430 IF p1sa THEN p1dir=0
440 IF p2sa THEN p2dir=0
450 RETURN
460 '
470 '
480 p2dir=((INKEY(4)=0)*1)+((INKEY(5)=0)*-1)
490 p1dir=((INKEY(69)=0)*1)+((INKEY(71)=0)*-1)
500 IF INKEY(63)=0 THEN p1sa=p1sa-1:IF p1sa=-1 THEN AFT
    ER 15 GOSUB 770
510 IF INKEY(10)=0 THEN p2sa=p2sa-1:IF p2sa=-1 THEN AFT
    ER 15 GOSUB 770
520 IF p1sa THEN p1dir=0
530 IF p2sa THEN p2dir=0
540 RETURN
550 '
560 '
570 pt=p1wp+p1dir:IF pt>25 OR pt<6 THEN RETURN ELSE p1w
    p=pt
580 p1dir=0
590 PEN 1:LOCATE 3,p1wp:CLS #3:PRINT CHR$(209);:RETURN
600 '
610 '
620 pt=p2wp+p2dir:IF pt>25 OR pt<6 THEN RETURN ELSE p2w
    p=pt
630 p2dir=0
640 PEN 2:LOCATE 18,p2wp:CLS #5:PRINT CHR$(211);:RETURN
650 '
660 '
670 PAPER #4,4:WINDOW #4,4,17,p1wp,p1wp:CLS#4:FRAME:FRA
    ME
680 PAPER #4,0:CLS#4
690 GOTO 570
700 '
710 '
720 PAPER #6,5:WINDOW #6,4,17,p2wp,p2wp:CLS#6:FRAME:FRA
    ME
730 PAPER #6,0:CLS#6
740 GOTO 620
750 '
```

```
760 '
770 pwpe=(p1wp=p2wp):IF p1sa AND NOT(p2sa) AND pwpe THE
    N p1sc=p1sc+1:SOUND 132,120,10,0,1,0:PRINT#1,a$(p1s
    c);:IF p1sc=9 THEN 860
780 IF p2sa AND NOT(p1sa) AND pwpe THEN p2sc=p2sc+1:SOU
    ND 132,100,10,0,1,0:PRINT#2,a$(p2sc);:IF p2sc=9 THE
    N 860
790 IF p1sa THEN SOUND 132,40,70,0,1,1
800 IF p2sa THEN SOUND 132,56,70,0,1,1
810 p1sa=0
820 p2sa=0
830 RETURN
840 '
850 '
860 PEN 6
870 LOCATE 6,10:PRINT"GAME OVER"
880 IF p1sc=9 THEN INK 1,2,20:INK 2,0 ELSE INK 2,6,17:I
    NK 1,0
890 SOUND 129,1000,0,12,3:SOUND 130,900,0,12,3
900 WHILE INKEY$<>"":WEND
910 t!=TIME:WHILE t!+2000>TIME:WEND
920 WHILE INKEY$="":WEND
930 CLS
940 finished=-1
950 RETURN
960 '
970 '
980 a$(0)="111101101101111"
990 a$(1)="001001001001001"
1000 a$(2)="111001111100111"
1010 a$(3)="111001111001111"
1020 a$(4)="100100101111001"
1030 a$(5)="111100111001111"
1040 a$(6)="111100111101111"
1050 a$(7)="111001001010010"
1060 a$(8)="111101111101111"
1070 a$(9)="111101111001001"
1080 FOR n=0 TO 9
1090 howlong=LEN(a$(n))
1100 FOR n2=1 TO howlong
1110 IF MID$(a$(n),n2,1)="1"THEN MID$(a$(n),n2,1)=CHR$(
     143)ELSE MID$(a$(n),n2,1)=CHR$(32)
1120 NEXT n2,n
1130 '
```

```
1140 '
1150 b$="ELECTRIC FENCING"
1160 c$=CHR$(32)+CHR$(164)+" Alexander Martin"
1170 ENV 1,=9,2000:ENT -1,6,3,1
1180 ENV 2,127,0,0,127,0,0,127,0,0,127,0,0,127,0,0
1190 ENV 3,=9,9000
1200 '
1210 '
1220 BORDER 0
1230 INK 0,12:PEN #4,1:PEN #6,2:PEN #1,1:PEN #2,2:PAPER
      #1,3:PAPER #2,3:PEN #0,6
1240 RETURN 'FROM SETTING UP CONSTANTS
1250 '
1260 '
1270 INK 0,12:INK 1,2:INK 2,6:INK 3,13:INK 4,20:INK 5,1
     7:INK 6,20
1280 WINDOW #3,3,3,6,25:WINDOW #5,18,18,6,25
1290 WINDOW #1,3,5,1,5:WINDOW #2,16,18,1,5:WINDOW #7,1,
     20,1,5:PAPER #7,3
1300 CLS:CLS#7:PRINT#1,a$(0);:PRINT#2,a$(0);:p1sc=0:p2s
     c=0:p1wp=5:p2wp=24:p1dir=1:p2dir=1
1310 GOSUB 570:GOSUB 620
1320 SOUND 1,1000,0,12,2:SOUND 2,900,0,12,2
1330 p1sa=0:p2sa=0:finished=0
1340 RETURN 'FROM GAME SHEET RESTORE
1350 '
1360 '
1370 CLS
1380 PEN 7
1390 FOR n=1 TO LEN(b$)
1400 LOCATE 2+n,10
1410 FOR n2=LEN(b$) TO n STEP-1
1420 PRINT MID$(b$,n2,1)
1430 LOCATE 2+n,10
1440 SOUND 135,20*n2,5,12,2,1
1450 NEXT n2,n
1460 SOUND 135,100,0,13,3,1,20
1470 PEN 6:PRINT:PRINT:PRINT:PRINT c$
1480 FOR n=1 TO 5000:NEXT
1490 RETURN
1500 '
1510 '
1520 IF j THEN RETURN
1530 CLS
```

```
1540 LOCATE 7,5
1550 PRINT"CONTROLS"
1560 PRINT
1570 PRINT"  PLAYER1  PLAYER2"
1580 PRINT
1590 PRINT"  A       up       6"
1600 PRINT"  Z      down      3"
1610 PRINT"  X      fire      7"
1620 t!=TIME:WHILE t!+1000>TIME:WEND
1630 RETURN
```

Amthello

Bir oyuncu bilgisayara karşı. Yalnız klavye.

```
10 'AMTHELLO by M.J.GRIBBINS
20 'copyright (c) AMSOFT 1984
30 '
40 BORDER 14
50 CLEAR
60 MODE 1:PEN 0:PAPER 1:CLS
70 INK 0,0:INK 1,14:INK 2,18:INK 3,26
80 LOCATE 2,3:PEN 3:PRINT"A":LOCATE 3,4:PRINT"M":LOCATE
    4,5:PRINT"T":LOCATE 5,6:PRINT"H"
90 LOCATE 6,7:PRINT"E":LOCATE 7,8:PRINT"L":LOCATE 8,9:P
   RINT"L":LOCATE 9,10:PRINT"O"
100 WINDOW #1,2,39,22,25:PAPER #1,1:PEN #1,0:CLS #1
110 PEN 0
120 LOCATE #1,8,1:PRINT #1,"BLACK ALWAYS PLAYS FIRST"
130 LOCATE #1,1,3:PRINT #1,"PRESS B OR W TO CHOOSE BLAC
    K OR WHITE"
140 B$=INKEY$:IF B$="" THEN 140
150 IF B$="W" OR  B$="w" THEN Q%=3:N%=0:GOTO 210
160 IF B$="B" OR  B$="b" THEN Q%=0:N%=3:GOTO 210
170 CLS #1:LOCATE #1,4,3
180 PRINT #1,"        BLACK OR WHITE ONLY"
190 FOR T=0 TO 1000:NEXT T
200 GOTO 140
210 DIM C%(10,10),P%(9,9),C1%(8),C2%(8),CX%(9),CY%(9)
220 I1%=2:J1%=2:I2%=7:J2%=7
230 FOR I%=0 TO 9
240 C%(I%,0%)=6:C%(0,I%)=6
250 C%(9,I%)=6:C%(I%,9)=6
260 NEXT I%
270 FOR I%=1 TO 8
280 READ C1%(I%),C2%(I%)
290 FOR J%= 1 TO 8
300 READ P%(I%,J%)
310 C%(I%,J%)=6
320 NEXT J%:NEXT I%
330 C%(4,4)=3:C%(4,5)=0:C%(5,4)=0:C%(5,5)=3
```

```
340 FOR K%=1 TO 58
350 READ AR%,BR%,CR%,DR%
360 PLOT AR%,BR%:DRAW CR%,DR%,0
370 NEXT K%
380 GOSUB 1460
390 IF Q%=3 GOTO 770
400 CLS #1:INPUT #1,"  WHICH LINE DO YOU WANT ";E%
410 IF E% <1 OR E% >8 GOTO 400
420 LOCATE #1,1,3:INPUT #1,"WHICH COLUMN DO YOU WANT ";
    D%
430 IF D% <1 OR D% >8 GOTO 420
440 IF C%(D%,E%)=6 GOTO 480
450 CLS #1:LOCATE #1,5,2:PRINT #1,"THAT SQUARE IS ALREA
    DY OCCUPIED !"
460 FOR T=1 TO 1000:NEXT T
470 GOTO 400
480 PLOT 270+(30*D%),70+(30*E%):DRAW 290+(30*D%),89+(30
    *E%),Q%
490 PLOT 290+(30*D%),70+(30*E%):DRAW 270+(30*D%),89+(30
    *E%),Q%
500 GOTO 540
510 FOR M%= 0 TO 19 STEP 2:PLOT 270+(30*D%),70+M%+(30*E
    %)
520 DRAW 290+(30*D%),70+M%+(30*E%),6:NEXT M%
530 GOTO 400
540 VRX%=0
550 FOR K%=1 TO 8
560 VR%=0:C3%=D%:C4%=E%
570 C3%=C3%+C1%(K%):C4%=C4%+C2%(K%)
580 IF C%(C3%,C4%)=N% GOTO 590 ELSE 600
590 VR%=VR%+1:GOTO 570
600 IF C%(C3%,C4%)=6 GOTO 610 ELSE 620
610 NEXT K%:GOTO 670
620 IF VR%=0 GOTO 610 ELSE 630
630 VRX%=VRX%+VR%
640 C3%=C3%-C1%(K%):C4%=C4%-C2%(K%)
650 IF C%(C3%,C4%)=6 GOTO 610 ELSE 660
660 C%(C3%,C4%)=Q%:GOTO 640
670 IF VRX%=0 GOTO 680 ELSE 710
680 CLS #1:PRINT #1," THIS IS NOT A POSSIBLE CHOICE"
690 FOR T=1 TO 1000:NEXT T
700 GOTO 510
710 E%=E%:D%=D%:VRX%=VRX%
720 CLS #1:PRINT #1,"YOU HAVE PLAYED LINE NUMBER ";E%
```

```
730 PRINT #1,"           AND COLUMN NUMBER ";D%
740 LOCATE #1,2,4:PRINT #1,"THAT GIVES YOU ";VRX%;" SQU
    ARES(S)"
750 C%(D%,E%)=Q%:GOSUB 1710
760 GOSUB 1460
770 CLS #1:LOCATE #1,10,2:PRINT #1,"NOW IT'S MY TURN ..
    .!"
780 P%=0:VRX%=0:VRY%=0
790 IF I1%*J1%=1 AND I2%*J2%=64 GOTO 860
800 FOR K%=2 TO 7
810 IF C%(2,K%) <> 6 THEN I1%=1
820 IF C%(7,K%) <> 6 THEN I2%=8
830 IF C%(K%,2) <> 6 THEN J1%=1
840 IF C%(K%,7) <> 6 THEN J2%=8
850 NEXT K%
860 FOR I%=I1% TO I2%
870 FOR J%=J1% TO J2%
880 IF C%(I%,J%)=6 GOTO 1030
890 NEXT J%:NEXT I%
900 IF P% > 0 GOTO 1000
910 IF PAS%=1 GOTO 920 ELSE 940
920 CLS #1:PRINT #1," DEADLOCK ! I MUST PASS ALSO.GAME
    OVER"
930 FOR T=1 TO 1000:NEXT T:GOTO 1550
940 CLS #1:LOCATE #1,18,2:PRINT #1,"I MUST PASS"
950 GOSUB 2720
960 IF PAS%=1 GOTO 970 ELSE 990
970 CLS #1:PRINT #1,"DEADLOCK! YOU MUST PASS ALSO.GAME
    OVER"
980 FOR T=1 TO 1000:NEXT T:GOTO 1550
990 GOTO 400
1000 IF LC%=0 THEN LC%=1:RANDOMIZE LC%:RL%=RND(LC%)
1010 CX1%=CX%(RL%):CX2%=CY%(RL%)
1020 GOTO 1220
1030 VRX%=0
1040 FOR K%=1 TO 8
1050 VR%=0:C3%=I%:C4%=J%
1060 C3%=C3%+C1%(K%):C4%=C4%+C2%(K%)
1070 IF C%(C3%,C4%)=Q% GOTO 1080 ELSE 1090
1080 VR%=VR%+1:GOTO 1060
1090 IF C%(C3%,C4%)=6 GOTO 1100 ELSE 1110
1100 NEXT K%:GOTO 1130
1110 IF VR%=0% GOTO 1100 ELSE 1120
1120 VRX%=VRX%+VR%:GOTO 1100
```

```
1130 IF VRX%=0 GOTO 890
1140 IF P%(I%,J%) < P% GOTO 890
1150 IF P%(I%,J%) > P% GOTO 1160 ELSE 1170
1160 P%=P%(I%,J%):VRY%=VRX%:LC%=0:CX%(0)=I%:CY%(0)=J%:G
     OTO 890
1170 IF VRY% > VRX% GOTO 890
1180 IF VRY% < VRX% GOTO 1190 ELSE 1200
1190 LC%=0:VRY%=VRX%:CX%(0)=I%:CY%(0)=J%:GOTO 890
1200 LC%=LC%+1:CX%(LC%)=I%:CY%(LC%)=J%
1210 GOTO 890
1220 CX2%=CX2%:CX1%=CX1%:VRY%=VRY%
1230 CLS #1:PRINT #1," I CHOOSE LINE NUMBER ";CX2%
1240 PRINT #1,"    AND COLUMN NUMBER ";CX1%
1250 LOCATE #1,1,4:PRINT #1,"THAT GIVES ME ";VRY%;" SQU
     ARE(S)"
1260 PLOT 270+(30*CX1%),70+(30*CX2%):DRAW 290+(30*CX1%)
     ,89+(30*CX2%),N%
1270 PLOT 290+(30*CX1%),70+(30*CX2%):DRAW 270+(30*CX1%)
     ,89+(30*CX2%),N%
1280 FOR T=1 TO 1000:NEXT T
1290 FOR K%=1 TO 8
1300 VR%=0:C3%=CX1%:C4%=CX2%
1310 C3%=C3%+C1%(K%):C4%=C4%+C2%(K%)
1320 IF C%(C3%,C4%)=Q% GOTO 1330 ELSE 1340
1330 VR%=VR%+1:GOTO 1310
1340 IF C%(C3%,C4%)=6 GOTO 1350 ELSE 1360
1350 NEXT K%:GOTO 1400
1360 IF VR%=0 GOTO 1350
1370 C3%=C3%-C1%(K%):C4%=C4%-C2%(K%)
1380 IF C%(C3%,C4%)=6 GOTO 1350
1390 C%(C3%,C4%)=N%:GOTO 1370
1400 C%(CX1%,CX2%)=N%
1410 GOSUB 2720
1420 GOSUB 1460
1430 IF PAS%=1 GOTO 1440 ELSE 1450
1440 CLS #1:PRINT #1,"      YOU MUST PASS":FOR T=1 TO 10
     00:NEXT T:GOTO 770
1450 GOTO 400
1460 FOR I%=1 TO 8
1470 FOR J%=1 TO 8
1480 FOR M%=0 TO 19 STEP 2
1490 Z%=270+(30*I%):H%=70+(30*J%):W%=H%+M%
1500 PLOT Z%,W%:DRAW Z%+20,W%,C%(I%,J%)
1510 NEXT M%:NEXT J%:NEXT I%
```

```
1520 X%=X%+1
1530 IF X%=61 GOTO 1550
1540 RETURN
1550 CQ%=0:CN%=0
1560 FOR I%=1 TO 8
1570 FOR J%=1 TO 8
1580 IF C%(I%,J%)=Q% THEN CQ%=CQ%+1
1590 IF C%(I%,J%)=N% THEN CN%=CN%+1
1600 NEXT J%:NEXT I%
1610 IF CQ% > CN% GOTO 1680
1620 IF CQ%=CN% GOTO 1630 ELSE 1650
1630 CLS #1:LOCATE #1,25,2:PRINT #1,"DEADLOCK"
1640 END
1650 CLS #1:LOCATE #1,5,1:PRINT #1,"YOU HAVE ";CQ%;" SQ
     UARES;I HAVE ";CN%
1660 LOCATE #1,11,3:PRINT #1,"I HAVE WON....!!!!"
1670 END
1680 CLS #1:LOCATE #1,5,1:PRINT #1,"YOU HAVE ";CQ%;" SQ
     UARES;I HAVE ";CN%
1690 LOCATE #1,5,3:PRINT #1,"WELL DONE. YOU HAVE WON !!
     "
1700 END
1710 IF C%(2,2)=Q% AND (C%(3,1)=N% OR C%(1,3)=N%) GOTO
     1720 ELSE 1730
1720 P%(3,1)=1:P%(1,3)=1
1730 IF C%(7,7)=Q% AND (C%(8,6)=N% OR C%(6,8)=N%) GOTO
     1740 ELSE 1750
1740 P%(8,6)=1:P%(6,8)=1
1750 IF C%(2,7)=Q% AND (C%(1,6)=N% OR C%(3,8)=N%) GOTO
     1760 ELSE 1770
1760 P%(1,6)=1:P%(3,8)=1
1770 IF C%(7,2)=Q% AND (C%(6,1)=N% OR C%(8,3)=N%) GOTO
     1780 ELSE 1790
1780 P%(6,1)=1:P%(8,3)=1
1790 IF D%=1 OR D%=8 OR E%=1 OR E%=8 GOTO 1820
1800 IF CX1%=1 OR CX1%=8 OR CX2%=1 OR CX2%=8 GOTO 1820
1810 RETURN
1820 FOR J%=1 TO 8 STEP 7
1830 FOR I%=2 TO 7
1840 IF C%(I%,J%)=N% GOTO 1850 ELSE 1860
1850 P%(I%+1,J%)=21:P%(I%-1,J%)=21
1860 IF C%(J%,I%)=N% GOTO 1870 ELSE 1880
1870 P%(J%,I%+1)=21:P%(J%,I%-1)=21
1880 NEXT I%
```

```
1890 FOR I%=2 TO 7
1900 IF C%(I%,J%)=Q% GOTO 1910 ELSE 1920
1910 P%(I%+1,J%)=2:P%(I%-1,J%)=2
1920 IF C%(J%,I%)=Q% GOTO 1930 ELSE 1940
1930 P%(J%,I%+1)=2:P%(J%,I%-1)=2
1940 NEXT I%:NEXT J%
1950 P%(1,2)=1:P%(1,7)=1:P%(2,1)=1:P%(7,1)=1
1960 P%(2,8)=1:P%(7,8)=1:P%(8,2)=1:P%(8,7)=1
1970  FOR I%=2 TO 7
1980 IF C%(1,I%-1)=Q% AND C%(1,I%+1)=Q% THEN P%(1,I%)=2
     5
1990 IF C%(8,I%-1)=Q% AND C%(8,I%+1)=Q% THEN P%(8,I%)=2
     5
2000 IF C%(I%-1,1)=Q% AND C%(I%+1,1)=Q% THEN P%(I%,1)=2
     5
2010 IF C%(I%-1,8)=Q% AND C%(I%+1,8)=Q% THEN P%(I%,8)=2
     5
2020 NEXT I%
2030 FOR J%=1 TO 8 STEP 7
2040 FOR I%=4 TO 8
2050 IF C%(J%,I%) <> N% GOTO 2140
2060 IC%=I%-1:IF C%(J%,IC%)=6 GOTO 2140
2070 IF C%(J%,IC%)=Q% GOTO 2080 ELSE 2090
2080 IC%=IC%-1:GOTO 2070
2090 IF C%(J%,IC%)=6 GOTO 2110
2100 GOTO 2140
2110 IF IC%=0 GOTO 2140
2120 IF C%(J%,I%+1)=Q% AND C%(J%,IC%-1)=6 GOTO 2140
2130 P%(J%,IC%)=26
2140 IF C%(I%,J%) <> N% GOTO 2230
2150 IC%=I%-1:IF C%(IC%,J%)=6 GOTO 2230
2160 IF C%(IC%,J%)=Q% GOTO 2170 ELSE 2180
2170 IC%=IC%-1:GOTO 2160
2180 IF C%(IC%,J%)=6 GOTO 2200
2190 GOTO 2230
2200 IF IC%=0 GOTO 2230
2210 IF C%(I%+1,J%)=Q% AND C%(IC%-1,J%)=6 GOTO 2230
2220 P%(IC%,J%)=26
2230 NEXT I%
2240 FOR I%=1 TO 5
2250 IF C%(J%,I%) <> N% GOTO 2340
2260 IC%=I%+1:IF C%(J%,IC%)=6 GOTO 2340
2270 IF C%(J%,IC%)=Q% GOTO 2280 ELSE 2290
2280 IC%=IC%+1:GOTO 2270
```

```
2290 IF C%(J%,IC%)=6 GOTO 2310
2300 GOTO 2340
2310 IF IC%=9 GOTO 2340
2320 IF C%(J%,I%-1)=Q% AND C%(J%,IC%+1)=6 GOTO 2340
2330 P%(J%,IC%)=26
2340 IF C%(I%,J%) <> N% GOTO 2430
2350 IC%=I%+1:IF C%(IC%,J%)=6 GOTO 2430
2360 IF C%(IC%,J%)=Q% GOTO 2370 ELSE 2380
2370 IC%=IC%+1:GOTO 2360
2380 IF C%(IC%,J%)=6 GOTO 2400
2390 GOTO 2430
2400 IF IC%=9 GOTO 2430
2410 IF C%(I%-1,J%)=Q% AND C%(IC%+1,J%)=6 GOTO 2430
2420 P%(IC%,J%)=26
2430 NEXT I%:NEXT J%
2440 IF C%(1,1)=N% GOTO 2450 ELSE 2460
2450 FOR I%=2 TO 6:P%(1,I%)=20:P%(I%,1)=20:NEXT I%
2460 IF C%(1,8)=N% GOTO 2470 ELSE 2480
2470 FOR I%=2 TO 6:P%(I%,8)=20:P%(1,9-I%)=20:NEXT I%
2480 IF C%(8,1)=N% GOTO 2490 ELSE 2500
2490 FOR I%=2 TO 6:P%(9-I%,1)=20:P%(8,I%)=20:NEXT I%
2500 IF C%(8,8)=N% GOTO 2510 ELSE 2520
2510 FOR I%=3 TO 7:P%(I%,8)=20:P%(8,I%)=20:NEXT I%
2520 IF C%(1,1) <> 6 THEN P%(2,2)=5
2530 IF C%(1,8) <> 6 THEN P%(2,7)=5
2540 IF C%(8,1) <> 6 THEN P%(7,2)=5
2550 IF C%(8,8) <> 6 THEN P%(7,7)=5
2560 P%(1,1)=30:P%(1,8)=30:P%(8,1)=30:P%(8,8)=30
2570 FOR I%=3 TO 6
2580 IF C%(1,I%)=N% THEN P%(2,I%)=4
2590 IF C%(8,I%)=N% THEN P%(7,I%)=4
2600 IF C%(I%,1)=N% THEN P%(I%,2)=4
2610 IF C%(I%,8)=N% THEN P%(I%,7)=4
2620 NEXT I%
2630 IF C%(7,1)=Q% AND C%(4,1)=N% AND C%(6,1)=6 AND C%(
     5,1)=6 THEN P%(6,1)=26
2640 IF C%(1,7)=Q% AND C%(1,4)=N% AND C%(1,6)=6 AND C%(
     1,5)=6 THEN P%(1,6)=26
2650 IF C%(2,1)=Q% AND C%(5,1)=N% AND C%(3,1)=6 AND C%(
     4,1)=6 THEN P%(3,1)=26
2660 IF C%(1,2)=Q% AND C%(1,5)=N% AND C%(1,3)=6 AND C%(
     1,4)=6 THEN P%(1,3)=26
2670 IF C%(8,2)=Q% AND C%(8,5)=N% AND C%(8,3)=6 AND C%(
     8,4)=6 THEN P%(8,3)=26
```

```
2680 IF C%(2,8)=Q% AND C%(5,8)=N% AND C%(3,8)=6 AND C%(
     4,8)=6 THEN P%(3,8)=26
2690 IF C%(8,7)=Q% AND C%(8,4)=N% AND C%(8,5)=6 AND C%(
     8,6)=6 THEN P%(8,6)=26
2700 IF C%(7,8)=Q% AND C%(4,8)=N% AND C%(5,8)=6 AND C%(
     6,8)=6 THEN P%(6,8)=26
2710 RETURN
2720 PAS%=0
2730 FOR I%=1 TO 8
2740 FOR J%=1 TO 8
2750 IF C%(I%,J%)=Q% GOTO 2780
2760 NEXT J%:NEXT I%
2770 PAS%=1:RETURN
2780 FOR K%=1 TO 8
2790 VR%=0:C3%=I%:C4%=J%
2800 C3%=C3%+C1%(K%):C4%=C4%+C2%(K%)
2810 IF C3% < 1 OR C3% > 8 GOTO 2820 ELSE 2830
2820 NEXT K%:GOTO 2760
2830 IF C4% < 1 OR C4% > 8 GOTO 2820 ELSE 2840
2840 IF C%(C3%,C4%)=N% GOTO 2850 ELSE 2860
2850 VR%=VR%+1:GOTO 2800
2860 IF C%(C3%,C4%)=Q% GOTO 2820 ELSE 2870
2870 IF VR% > 0 THEN RETURN
2880 GOTO 2820
2890 DATA 1,0,30,1,20,10,10,20,1,30,1,1,1,1,3
2900 DATA 3,3,3,1,1,0,1,20,3,5,5,5,5,3,20,-1,1,10,3,5
2910 DATA 0,0,5,3,10,-1,0,10,3,5,0,0,5,3,10,-1
2920 DATA -1,20,3,5,5,5,5,3,20,0,-1,1,1,3,3,3,3,1,1,1,-
     1,30,1,20,10,10,20,1,30
2930 DATA 263,100,263,120,270,130,255,130,255,130,255,1
     40,255,140,270,140
2940 DATA 270,140,270,150,270,150,255,150,255,160,270,1
     60,270,160,270,180
2950 DATA 270,180,255,180,270,170,255,170,270,190,270,2
     10,270,200,255,200
2960 DATA 255,200,255,210,255,220,270,220,270,220,270,2
     30,270,230,255,230
2970 DATA 255,230,255,240,255,240,270,240,255,250,270,2
     50,270,250,270,260
2980 DATA 270,260,255,260,255,250,255,270,270,280,270,3
     00,270,300,255,300
2990 DATA 255,310,255,330,255,330,270,330,270,330,270,3
     10,270,310,255,310
3000 DATA 255,320,270,320
```

```
3010 DATA 310,355,310,375,350,355,335,355,335,355,335,3
     65,335,365,350,365
3020 DATA 350,365,350,375,350,375,335,375,365,355,380,3
     55,380,355,380,375
3030 DATA 380,375,365,375,380,365,365,365,410,355,410,3
     75,410,365,395,365
3040 DATA 395,365,395,375,425,355,440,355,440,355,440,3
     65,440,365,425,365
3050 DATA 425,365,425,375,425,375,440,375,455,375,455,3
     55,455,355,470,355
3060 DATA 470,355,470,365,470,365,455,365,485,375,500,3
     75,500,375,500,355
3070 DATA 515,375,515,355,515,355,530,355,530,355,530,3
     75,530,375,515,375
3080 DATA 515,365,530,365
```

Raffles

Bir oyuncu bilgisayara karşı. Yalnız klavye.

```
10 'RAFFLES by DAVID RADISIC
20 'copyright (c) AMSOFT 1985
30 '
40 MODE 0:INK 0,0:BORDER 0:INK 1,26:INK 2,15:INK 3,25
50 INK 4,14:INK 5,24,12:INK 6,0:INK 7,0:INK 8,0:PAPER #
   1,7
60 delay=200
70 DIM objx(5,20),objy(5,20),gemx(5,20),gemy(5,20)
80 GOSUB 380
90 GOSUB 720
100 pause=200:GOSUB 340
110 IF gems=0 THEN GOSUB 970
120 PEN 4
130 FOR i=10 TO 12
140 LOCATE 15,i:PRINT"SWAG";
150 NEXT
160 PAPER 0:CLS#2:PAPER 8
170 GOSUB 1170
180 GOSUB 1230
190 GOSUB 1370
200 GOSUB 1510
210 IF rm=0 THEN GOSUB 1900
220 IF dead=0 THEN 160
230 pause=100:GOSUB 340
240 PAPER 0:CLS:PEN 1
250 LOCATE 4,3:PRINT"Do you want";
260 LOCATE 4,5:PRINT"Another game";
270 PEN 5:LOCATE 9,7:PRINT"Y/N";
280 i$=UPPER$(INKEY$):IF i$<>"Y" AND i$<>"N" THEN 280
290 IF i$="N" THEN MODE 2:PEN 1:STOP
300 RUN
310 IF dog=1 THEN RETURN
320 dog=1:dogx=minx(rm):dogy=miny(rm)
330 RETURN
340 FOR loop=1 TO pause
350 FRAME
360 NEXT
370 RETURN
```

```
380 rm=1:xp=6:yp=4:man$=CHR$(224):dog=0:stolen=0
390 SYMBOL 240,8,8,8,8,8,8,8,8
400 SYMBOL 241,0,0,0,0,255,0,0,0
410 SYMBOL 242,0,0,0,0,15,8,8,8
420 SYMBOL 243,0,0,0,0,248,8,8,8
430 SYMBOL 244,8,8,8,8,248,0,0,0
440 SYMBOL 245,8,8,8,8,15,0,0,0
450 SYMBOL 246,8,12,13,14,12,12,8,8
460 SYMBOL 247,8,12,12,14,13,12,8,8
470 SYMBOL 248,8,24,88,56,24,24,8,8
480 SYMBOL 249,8,24,24,56,88,24,8,8
490 SYMBOL 250,0,0,255,129,129,129,255,0
500 SYMBOL 251,28,20,20,20,20,20,20,28
510 SYMBOL 252,0,0,255,255,255,255,255,0
520 SYMBOL 253,28,28,28,28,28,28,28,28
530 SYMBOL 255,195,165,60,126,90,60,36,24
540 ENT 1,12,-4,1
550 ENT -2,=1000,60,=3000,40
560 ENV 1,10,1,5,2,-4,1,2,-1,20
570 windw$(1)=STRING$(2,250):windw$(2)=CHR$(251)+CHR$(8
    )+CHR$(10)+CHR$(251)+CHR$(8)+CHR$(10)+CHR$(251)
580 door$(1)=STRING$(2,252):door$(2)=CHR$(253)+CHR$(8)+
    CHR$(10)+CHR$(253)+CHR$(8)+CHR$(10)+CHR$(253)
590 switch$(1,0)=CHR$(246):switch$(1,1)=CHR$(247)
600 switch$(2,0)=CHR$(248):switch$(2,1)=CHR$(249)
610 gem$=CHR$(144):obj$=CHR$(233):dog$=CHR$(255)
620 hit$=CHR$(246)+CHR$(248)+CHR$(247)+CHR$(249)+CHR$(2
    52)+CHR$(253)+CHR$(250)+CHR$(251)+gem$+obj$+dog$
630 RESTORE 3010
640 FOR i=1 TO 5
650 READ minx(i),miny(i),maxx(i),maxy(i)
660 READ dir(i,1),dir(i,2),dir(i,3),dir(i,4)
670 NEXT
680 WINDOW #1,minx(rm)-1,maxx(rm)+1,miny(rm)-1,maxy(rm)
    +1
690 WINDOW #2,1,14,1,25
700 CLS #1:PAPER #0,8
710 RETURN
720 ORIGIN 50,50
730 INK 6,24,12
740 RESTORE 3060
750 GOSUB 1280
760 LOCATE 3,20
770 PEN 5:PRINT">";
```

```
780 PEN 1:PRINT"Escape  Routes";
790 PEN 5:PRINT"<";:PEN 1
800 LOCATE 10,2:PRINT"IN";
810 pause=300:GOSUB 340
820 CLS:LOCATE 1,3:INK 6,0
830 PEN 1:PRINT man$;"  You The Thief":PRINT
840 PEN 2:PRINT LEFT$(door$(1),1);LEFT$(door$(2),1);" D
    oors":PRINT
850 PEN 3:PRINT switch$(1,0);switch$(2,0);" LightSwitch
     (OFF)"
860 PEN 3:PRINT switch$(1,1);switch$(2,1);" LightSwitch
     (ON)":PRINT
870 PEN 4:PRINT LEFT$(windw$(1),1);LEFT$(windw$(2),1);"
     Windows":PRINT
880 PEN 5:PRINT gem$;"  Precious Gems":PRINT
890 PAPER 1:PEN 0:PRINT obj$;"  Obstructions ":PEN 1:PA
    PER 0:PRINT
900 PEN 1:PRINT dog$;"  The Dog"
910 PEN 5:PRINT:PRINT:PRINT
920 PRINT"  Use Joystick":PRINT"   Or Cursor keys"
930 dummy=REMAIN(1)
940 AFTER delay*4,1 GOSUB 340
950 RETURN
960 '
970 'Generate Gems/obstacles
980 '
990 FOR room=1 TO 5
1000 gemr=INT(RND*8)+2:objr=INT(RND*10)+5
1010 minx=minx(room):miny=miny(room):maxx=maxx(room):ma
     xy=maxy(room)
1020 FOR i=1 TO gemr
1030 x=INT(RND*(maxx-minx+1))+minx
1040 y=INT(RND*(maxy-miny+1))+miny
1050 gemx(room,i)=x:gemy(room,i)=y
1060 gems=gems+1
1070 NEXT i
1080 FOR i=1 TO objr
1090 x=INT(RND*(maxx-minx+1))+minx
1100 y=INT(RND*(maxy-miny+1))+miny
1110 objx(room,i)=x:objy(room,i)=y
1120 NEXT i
1130 gems(room)=gemr:obj(room)=objr
1140 NEXT room
1150 CLS
```

```
1160 RETURN
1170 ON rm GOTO 1180,1190,1200,1210,1220
1180 RESTORE 2670:RETURN
1190 RESTORE 2740:RETURN
1200 RESTORE 2810:RETURN
1210 RESTORE 2880:RETURN
1220 RESTORE 2960:RETURN
1230 PAPER 0:READ rm$:PAPER 8
1240 WINDOW #1,minx(rm)-1,maxx(rm)+1,miny(rm)-1,maxy(rm
     )+1:CLS #1
1250 PEN 1:LOCATE 1,1:PRINT SPACE$(19);
1260 LOCATE 1,1:PRINT "Room :";rm$;
1270 IF lights(rm) THEN INK 7,10:INK 8,10 ELSE INK 7,0:
     INK 8,0
1280 READ a$:IF a$="END" THEN RETURN
1290 IF a$="D" THEN 2180
1300 IF a$="W" THEN 2260
1310 IF a$="L" THEN GRAPHICS PEN 1:GOTO 2340
1320 IF a$="S" THEN 2420
1330 IF a$="F" THEN GRAPHICS PEN 6:GOTO 2340
1340 PRINT"***ERROR ***";
1350 STOP
1360 '
1370 'Display gems/objects
1380 '
1390 PEN 6
1400 FOR i=1 TO obj(rm)
1410 LOCATE objx(rm,i),objy(rm,i)
1420 PRINT obj$;
1430 NEXT
1440 PEN 5
1450 FOR i=1 TO gems(rm)
1460 LOCATE gemx(rm,i),gemy(rm,i)
1470 PRINT gem$;
1480 NEXT
1490 PEN 1:LOCATE xp,yp:PRINT man$;
1500 RETURN
1510 xf=0:yf=0:PEN 1
1520 IF INKEY(0)<>-1 OR INKEY(72)<>-1 THEN yf=-1
1530 IF INKEY(2)<>-1 OR INKEY(73)<>-1 THEN yf=1
1540 IF INKEY(8)<>-1 OR INKEY(74)<>-1 THEN xf=-1
1550 IF INKEY(1)<>-1 OR INKEY(75)<>-1 THEN xf=1
1560 IF xf=0 AND yf=0 THEN 1630
1570 LOCATE xp+xf,yp+yf:ht$=COPYCHR$(#0)
```

```
1580 IF ASC(ht$)>239 AND ASC(ht$)<246 THEN 1510
1590 IF ht$<>" " THEN 1660
1600 LOCATE xp,yp:PRINT " ";
1610 PAPER 0:LOCATE 15,5:PRINT"     ";:PAPER 8
1620 xp=xp+xf:yp=yp+yf
1630 LOCATE xp,yp:PRINT man$;
1640 IF dog>0 THEN dog=dog MOD 2+1:IF dog=2 THEN 2550
1650 GOTO 1510
1660 hit=INSTR(hit$,ht$):char=ASC(MID$(hit$,hit,1))
1670 ON hit GOTO 1690,1690,1690,1690,1750,1750,1850,190
     0,1970,2090,2650
1680 GOTO 1600
1690 IF hit>2 AND hit<5 THEN char=char-1
1700 IF hit<3 THEN char=char+1
1710 PEN 3:LOCATE xp+xf,yp+yf:PRINT CHR$(char);
1720 lights(rm)=lights(rm) XOR 1
1730 IF lights(rm) THEN INK 7,10:INK 8,10 ELSE INK 7,0:
     INK 8,0
1740 GOTO 1510
1750 IF xf<>0 AND yf<>0 THEN 1630
1760 IF xf<0 THEN dir=4 ELSE IF xf>0 THEN dir=3
1770 IF yf<0 THEN dir=1 ELSE IF yf>0 THEN dir=2
1780 IF dir(rm,dir)=-1 THEN 1630 ELSE rm=dir(rm,dir)
1790 IF dog>0 THEN GOSUB 310
1800 IF dir=1 THEN xp=6:yp=maxy(rm)
1810 IF dir=2 THEN xp=6:yp=miny(rm)
1820 IF dir=3 THEN xp=minx(rm):yp=13
1830 IF dir=4 THEN xp=maxx(rm):yp=13
1840 RETURN
1850 IF xp>5 AND xp<8 THEN 1880
1860 IF xp<6 THEN dir=4 ELSE dir=3
1870 GOTO 1780
1880 IF yp>13 THEN dir=2 ELSE dir=1
1890 GOTO 1780
1900 PAPER 0:CLS:PEN 1
1910 LOCATE 3,3:PRINT"You have escaped";
1920 LOCATE 9,5:PRINT"with";
1930 IF gems=stolen THEN LOCATE 8,7:PRINT"All"; ELSE LO
     CATE 9,7
1940 PRINT USING " ##";stolen;
1950 PEN 5:LOCATE 9,9:PRINT"Gems";
1960 dead=1:RETURN
1970 LOCATE xp,yp:PRINT" ";:xp=xp+xf:yp=yp+yf
```

```
1980 i=0
1990 i=i+1
2000 IF i>gems(rm) THEN 1510
2010 IF gemx(rm,i)<>xp OR gemy(rm,i)<>yp THEN 1990
2020 IF i=gems(rm) THEN 2050
2030 gemx(rm,i)=gemx(rm,gems(rm))
2040 gemy(rm,i)=gemy(rm,gems(rm))
2050 gems(rm)=gems(rm)-1:stolen=stolen+1
2060 MOVE 400,150+(stolen*2),1,1:DRAW 520,150+(stolen*2
     ),1,1
2070 SOUND 129,248,10,12,0,1
2080 GOTO 1980
2090 noise=INT(RND*15)
2100 SOUND 1,3000,10,noise,0,0,10
2110 PAPER 0:LOCATE 15,5:PRINT"Crash ";:PAPER 8
2120 IF noise<10 OR delay=50 THEN 1630
2130 delay=delay-50
2140 dummy=REMAIN(1)
2150 AFTER delay*4,1 GOSUB 310
2160 GOTO 1630
2170 '
2180 'Draw doors
2190 '
2200 READ no,dr$
2210 IF dr$="V" THEN dr=2 ELSE dr=1
2220 PEN 2
2230 pic$=door$(dr):GOSUB 2500
2240 GOTO 1280
2250 '
2260 'Draw windows
2270 '
2280 READ no,wi$
2290 IF wi$="V" THEN wi=2 ELSE wi=1
2300 PEN 4
2310 pic$=windw$(wi):GOSUB 2500
2320 GOTO 1280
2330 '
2340 ' Draw lines
2350 '
2360 READ x1,y1,x2,y2
2370 MOVE x1,y1,,0
2380 DRAW x1,y2,,0:DRAW x2,y2,,0
2390 DRAW x2,y1,,0:DRAW x1,y1,,0
2400 GOTO 1280
```

```
2410 '
2420 'Draw switches
2430 '
2440 READ no,sw$
2450 IF sw$="L" THEN sw=1 ELSE sw=2
2460 PEN 3
2470 pic$=switch$(sw,0):GOSUB 2500
2480 GOTO 1280
2490 '
2500 'Print char
2510 '
2520 READ x,y:LOCATE x,y:PRINT pic$;
2530 no=no-1:IF no>0 THEN 2520
2540 RETURN
2550 PEN 1:LOCATE dogx,dogy:PRINT" ";
2560 man$=CHR$(225)
2570 IF (dogx=xp AND dogy=yp) OR (dogx=xp+xf AND dogy=y
     p+yf) THEN 2650
2580 IF dogx<xp THEN dogx=dogx+1
2590 IF dogx>xp THEN dogx=dogx-1
2600 IF dogy<yp THEN dogy=dogy+1
2610 IF dogy>yp THEN dogy=dogy-1
2620 LOCATE dogx,dogy:PRINT dog$;
2630 SOUND 1,0,RND*40,10,1,2,31
2640 GOTO 1510
2650 PRINT"SNAP";
2660 dead=1:RETURN
2670 DATA Hallway
2680 DATA L,64,308,226,4
2690 DATA D,2,H,6,3,6,22
2700 DATA D,2,V,4,12,9,11
2710 DATA S,1,L,4,11
2720 DATA S,1,R,9,14
2730 DATA END
2740 DATA Lounge
2750 DATA L,2,308,258,4
2760 DATA D,1,V,10,12
2770 DATA W,1,H,6,3
2780 DATA W,1,V,2,12
2790 DATA S,2,R,10,11,10,15
2800 DATA END
2810 DATA Dining room
2820 DATA L,2,308,258,4
2830 DATA W,1,V,10,12
```

```
2840 DATA W,1,H,6,3
2850 DATA D,1,V,2,12
2860 DATA S,2,L,2,11,2,15
2870 DATA END
2880 DATA Kitchen
2890 DATA L,2,276,384,4
2900 DATA D,2,H,6,5,6,22
2910 DATA W,1,H,10,22
2920 DATA W,1,V,14,13
2930 DATA D,1,V,2,13
2940 DATA S,1,L,2,16
2950 DATA END
2960 DATA Pantry
2970 DATA L,2,276,256,4
2980 DATA D,1,V,10,12
2990 DATA S,1,R,10,11
3000 DATA END
3010 DATA 5,4,8,21,0,4,3,2
3020 DATA 3,4,9,21,-1,-1,1,-1
3030 DATA 3,4,9,21,-1,-1,-1,1
3040 DATA 3,6,13,21,1,0,-1,5
3050 DATA 3,6,9,21,-1,-1,4,-1
3060 DATA L,64,308,480,100
3070 DATA F,250,98,294,102
3080 DATA F,250,306,294,310
3090 DATA F,390,94,430,106
3100 DATA F,390,302,430,314
3110 DATA F,474,240,488,270
3120 DATA F,474,124,488,154
3130 DATA F,58,240,72,270
3140 DATA L,226,308,322,180
3150 DATA L,160,180,480,100
3160 DATA L,64,180,160,100
3170 DATA END
```

Index

A

|A 41,69,205
ABS 99
AFTER 99
Ampflikatör 9,311
AMSDOS 69,199,203,227
AMSDOS hata mesajları 211,302,303
AND 100,353
Animasyon (canlılık) 387
AND (LOGO) 247
Anten soketi 3
APV 272
ARCTAN 244
Aritmetik işlemler 31
Aritmetik işlemler (LOGO) 244
ASC 100
ASCII 280,293
ASCII karakterleri 280,281,350
ASCII dosyaları 42,162,201,208,209,210
ASCII (LOGO) 239
ASM 232
ASSIGN, SYS 190
ATN 101
AUTO 92,101
AUX 223

B

|B 41,69,205
|BANKFIND 80,334
|BANKOPEN 79,332
|BANKREAD 80,333
|BANKWRITE 80,332
BASIC 22,97,304,342
BASIC dosyaları 41,162
Bellek bankası 77,329,397
BF 240
BIN$ 102
Binayr dosyalar 42,163,209,210
Binary numaralar 344
Bit 344
BK 257
BL 240
BOOTGEN 228
BORDER 46,102,277
BREAK 102
BRIGHINESS 4
BUTTONP 265
BYE 239,268

Byte....344

C

CALL....103
CAPSLOCK....17
CAT....40,103
CAT (kaset)....192
CATCH....269
CHAIN....103
CHAIN (kaset)....193
CHAINMERGE....104
CHAINMERGE (kaset)....193
CHANGEF....263
CHAR....240
CHR$....51,103,351
CINT....104
CLEAN....253
CLEAR....105
CLEAR INPUT....105
CLG....105
CLOAD....209,210,230
CLOŞEIN....93,106
CLOSEIN (kaset)....195
CLOSEOUT....92,106
CLOSEOUT (kaset)....196
CLR tuşu....18
CLS....22,106,275
CO....268
CON....223
CONT....106,300
CONTENTS....271
CONTRAST kontrol....4
COPYCHR$....107
COPY tuşu....27
COPYON....251
COPYOFF....251
COS....107
COS (LOGO)....245
COUNT....240
CP/M....213
CP/M 2,2....76,191,228,234
ICPM....36,205
CREAL....107
CRT....223
CS....253
CSAVE....209,210,230
CT....251
CURSOR....108,275
CURSOR (LOGO)....252

D

DATA....108,299,365

Data formatı....38,228,318
DATE....226
DDT....232
DEC$....109
,DEF....272
DEFAULTD....263
DEF FN....109,301
DEFINE....249
DEFINT....110
DEFREAL....110
DEFSTR....111
DEG....111
DEL tuşu....16
DELETE....111
,DEPOSIT....271
DERR....112,302,303,304
DEVICE....224
DI....112
DIM....86,113,300
|DIR....206
DIR (CP/M)....36,216,224
DIR (LOGO)....263
DIRPIC....264
DIRS....217
DIRSYS....217
|DISC....71,206
Disk sürücü....8,14,71,201,310
Disk sürücü II soketi....310
|DISC, IN....71,206
DISCKIT2....76,232
DISCKIT3....37,71
Disk organizasyonu....317
|DISC, OUT....71,206
Disket....12,35,183
DOT....253
DOTC....254
DRAW....54,114
DRAWR....114
|DRIVE....206
DRIVERS, GSX....190
Dr, LOGO....233
DUMP....232
Düzeltme....26
Düzeltme (LOGO)....251

E

ED....232
ED (LOGO)....251
EDALL....251
EDF....251
EDIT....26,115

EI 115
EJECT butonu 14
ELSE 28,116
EMPITY 240
,EMT 272
END 116
END (LOGO) 250
,ENL 272
ENT 66,116,375
ENT (LOGO) 267
ENTER tuşu 16
ENV 64,118,372
ENV (LOGO) 267
EDF 119,195,205,301
EDF (CP/M) 223
ER 260
ERA 217
IERA 206
ERALL 260
ERASE 120
ERASE (CP/M) 217,224
ERL 120
ERN 261
ERR 120,304
ERRACT 271
ERROR 121
ERROR (LOGO) 270
ESC tuşu 18,143
EVERY 121,364
,EXAMINE 271
EXP 122

F

FALSE 271
FD 257
FENCE 254
FILECOPY 191,229
FILL 57,122,382
FILL (LOGO) 254
FIRST 241
FIX 122
Floppy disket 11
FN 123
FOR 29,123,301,351
Format (print) 154,357
Format (disket) 35,37,73,228,317
FPUT 241
FRAME 53,123
FRE 124
FS 255

G

GENGRAF 191
GLIST 262
GO 268
GOSUB 29,124
GOTO 23,125
Grafik 44,51,382,390
GRAPHICS PAPER 125,384
GRAPHICS PEN 59,125,382
GPROP 262
GSX 190

H

HELP 185
HEX$ 126
HIMEM 126
HOME 257
HT 257

I

IBM format 228,318
IF 27,127
IF (LOGO) 268
,IN 271
INK 46,129,277
INKEY 128,316
INKEY$ 94,129,316
INP 129
INPUT 24,86,129,300
INPUT (kaset) 195
INSTR 88,131
INT 131
INT (LOGO) 245
ITEM 241

J

JOY 131,316
Joystick komutu (LOGO) 265
Joystick soket 7,310

K

KEY 132,294
KEYDEF 133,294,316
KEYP 265
KEYS, CCP 185,220
KEYS, DRL 220
KEYS, WP 220

L

LABEL 269
LANGUAGE 184,219,323
LAST 241
LC 241
LEFT$ 133
LEN 134
LET 134
LINE INPUT 134
LINE INPUT (kaset) 195
LIST 23,53,135
LIST (LOGO) 242
LISTP 242
LOAD 41,136
LOAD (kaset) 193
LOAD (CP/M) 232
LOAD (LOGO) 264
LOADPIC 264
LOCAL 248
LOCATE 51,136
LOG 136
LOG10 137
LOGO 233
LPT 222
LPUT 242
LST 223
LT 257
LOWER$ 137

M

MAKE 249
MASK 137,383
MAX 138
MEMBERP 242
MEMORY 138,299,319
MERGE 138
MERGE (kaset) 193
MID$ 139
MIN 140
MOD 34,140
Modülatör 3
Monitör 2
MOVCPM 232
MOVE 55,141
MOVER 141

N

NAMEP 249
NEW 142
NEXT 29,56,142,299,301,351

NODES 238,261
NOFORMAT 261
NOT 143,354
NOT (LOGO) 247
NOTRACE 270
NOWATCH 270
NUMBERP 243

O

ON BREAK CONT 143
ON BREAK GOSUB 143
ON BREAK STOP 144
ON ERROR GOTO 144,301
ON GOSUB 85,145
ON GOTO 145
ON SQ GOSUB 146,279,378
OP 269
OPENIN 93,146,301,302
OPENIN (Kaset) 195
OPENOUT 92,147,302
OPENOUT (Kaset) 195
OR 147,354
OR (LOGO) 248
ORIGIN 56,147
OUT 148
OUT (LOGO) 271

P

PADDLE 266
PAL 255
PALETTE 219
PAPER 46,148
PAUSE 270
PD 258
PE 258
PEEK 149
PEN 46,149
PI 150
PIECE 243
PIP 187,223
PLIST 262
PLOT 53,150
PLOTR 150
PO 250
POALL 261
POKE 151
PONS 261
POPS 261
POS 151,195
POTS 250
PPROP 262
PPS 263

PR 252
PRINT 22,152,357
PRINT SPC 153,357
PRINT TAB 153,357
PRINT USING 153,358
Printer soketi 8,313
,PRM 272
PRN 223
PROFILE 184,214
PU 258
PX 258

Q

QUOTIENT 245

R

RAD 156
RAM disk 332
RANDOM 245
RANDOMIZE 156
RC 266
READ 156,299,368
RECYCLE 262
REDEFP 271
RELEASE 157,377
RELEASE (LOGO) 268
REM 29,86,157
,REM 272
REMAIN 158
REMAINDER 245
REMPROP 263
REN 217
IREN 207
RENAME 217,224
RENUM 90,158
REPEAT 269
RERANDOM 246
RESTORE 159,368
RESUME 159,301
RESUME NEXT 159
RETURN 30,160,299
RETURN Tuşu 15,18
RIGHT$ 160
RL 266
RND 161
RQ 267
ROUND 161
RS232 222,339
RSX 318
RT 258
RUN 22,41,161
RUN (kaset) 193

RUN (LOGO).....269

S

SAVE.....42,162
SAVE (kaset).....197
SAVE (LOGO).....264
SAVEPIC.....264
!SCREENCOPY.....78,331,397
!SCREENSWAP.....78,331,397
SE.....243
Seri interface.....222,339
SET.....224
SET 24X80.....221
SETBG.....255
SETCURSOR.....252
SETD.....265
SETDEF.....226
SETH.....259
SETLST.....220
SETKEYS.....220
SETPAL.....255
SETPC.....259
SETPOS.....259
SETSCRUNCH.....255
SETSIO.....222
SETSPLIT.....252
SETUP.....230
SETX.....259
SETY.....259
SF.....256
SGN.....163
SHIFT tuşu.....16
SHOW (CP/M).....227
SHOW (LOGO).....252
SHUFFLE.....243
SIN.....163
SIN (LOGO).....246
SIO.....222
SOUND.....62,164,296
SOUND (LOGO).....267
SPACE$.....165
SPC.....166,357
SPEED INK.....166
SPEED KEY.....166
SPEED WRITE.....167,198
SQ.....167,378
SQR.....33,168
SS.....256
ST.....259
STAT.....229
STEP.....29,168
Stereo soketi.....9,311

STOP....169
STOP (LOGO)....269
STR$....169
STRING$....169
SUBMIT....189,232
SWAP....170
SYMBOL....170,277,356
SYMBOL AFTER....172
Syntax error....19,299
SYSGEN....228
System format....38,317

T

TAB....172,358
TAG....173,385
TAGOFF....174,385
TAN....174
ITAPE....70,207
ITAPE, IN....70,207
ITAPE, OUT....70,207
Tape soketi....310,311
TEST....174
TESTR....174
TEXT....250
TF....260
THEN....28,174
THING....249
THROW....270
TIME....175
TO....175
TO (LOGO)....250
TON zarfı....66,116,369
TOPLEVEL....271
TOWARDS....260
TRACE....270
TROFF....175
TRON....175
TRUE....272
TS....252
TV alıcısı....3,5
TYP....218
TYPE....218,224
TYPE (LOGO)....253

U

UC....244
UNT....176
UPPER$....176
USE....218
USER....218
IUSER....207
USING....218,359

V

VAL 177
VDU 187,321
Vendor formtat 228,318
Volume kontrol 9,67
VPOS 177

W

WAIT 178
WAIT (LOGO) 269
WATCH 270
WEND 178,302
WHERE 244
WHILE 178,302
WIDTH 178
WINDOW 93,179,277,362
WINDOW (LOGO) 256
WINDOW SWAP 179,362
WORD 244
WORDP 244
WRAP 256
WRITE 91,180,362
WRITE (kaset) 196

X

XOR 181,355
XPOS 181
XSUB 232

Y

YPOS 181

Z

ZONE 182

Ofset Hazırlık-Baskı: Stil Matbaacılık A.Ş. 511 96 87-88 İSTANBUL

EKAKOMP

EKAKOMP BİLGİSAYAR SAN. ve TİC. A.Ş.
MECLİSİ MEBUSAN CAD. SOMER HAN NO: 81-83 FINDIKLI-İSTANBUL
TEL: 151 37 24-25 TELEX: 25023 EKOP TR. FAX: (90) (1) 151 22 96

AMSTRAD KULLANMA KILAVUZU EKAHOMP