EMERALD

AMSTRAD CPC 464

USER INSTRUCTIONS

İÇİNDEKİLER:

6- SES ÖN BİLGİLERİ

- CPC 464 ses özellikleri
- Ton ve volume
- Ses sıraları
- Efektler

7- YAZICILAR VE OYUN ÇUBUKLARI

- Oyun çubuklarını kullanma
- Joy komutu
- Paralel bir yazıcı bağlama

8- AMSTRAD BASİC KOMUTLARININ ÖZETLERİ

Alfabetik sıraya göre dizilmiş BASIC komutları

9- BAŞKA PROGRAMLA BİLGİLERİ

- Programların iç yapısı
- İnterrupt'lar ve önemi
- Kontrol Karakterleri
- Makine dili alt program bağlantıları ve yüksek seviyeli BASİC komutları

10- İNTERRUPT OLANAKLARI

Gerçek zaman olanağı
- AFTER, EVERY ve REMAİN

EKLER

I ilk başlayanlara bilgisayarın neler
yapıp neler yapamıyacağı hakkında bilgiler
II "Bit"ler ve "Byte" lar-ikili ve onaltılı tabanlı
sayılar
III ASCII kodları ve karakter seti
Karakter açıklamaları
Klavye kodları
- Efektler
- Tecrübeli kullanıcıya bir ön bakış
- Genişleme olanakları
- Giriş/çıkış bağlantıları
- Görüntü planı ve organizasyonu
- Müzik yapacaklar
- Nota ve ton süreleri
- Hata kodları ve mesajları

C P C 464
KİŞİSEL RENKLİ BİLGİSAYAR

Bilgisayar dünyası çok kısa bir zamanda çok yol katetmiştir. 20. yüzyılın tüm teknolojik gelişmeleri arasında bilgisayar kesinlikle en hayranlık uyandırıcı olanıdır.

Bilgisayar donanım ve yazılım özellikleri, halen bilgisayar kullananların bile izleyemediği bir hızla ilerlemiştir. Bu yüzden CPC 464 sahiplerine ***BASIC***'ın tüm gücünü, inceliklerini, çalışma sistemini ve donanım vasıflarını göstermeye kalkışmak bir kaç bin sayfalık bir kitap gerektirirdi. Bu nedenle elinizdeki kitap CPC 464 ve yazılımına özlü ve kısa bir giriştir. Bu kitap daha bir çok özel ve ayrıntılı kurslar ve yayınlarla tamamlanacaktır.

BASIC'ın başka türlerine aşina olanlar ***AMSTRAD BASIC***'ın yapısına çok çabuk alışacaklardır. Yeni başlayanlar ise kullanılan terimlerin doğrudan ve kolay anlaşılır olmasından memnun kalacaklardır. Burada, özellikle bir çok standart olmayan ***BASIC*** yorumlarında bulunan çeşitli tanımlardan kaçınılmış ve kişisel bilgisayarlarda daha önce bulunmayan 'kullanışlı' özellikler eklenmiştir.

Bu kitap okuyucuya göre üç kısma ayrılmıştır.

Birinci, başlangıç temel kursudur. Bu kısım özellikle yeni başlıyanlara bilgisayar kavramları ve terimlerini tanıtmak için yazılmıştır. Şimdiye kadar kişisel bir bilgisayara sahip olmamış ve kendi başına küçük bir program yazacak kadar bilgisayar kullanmamış olanların Temel kursdan başlayıp buradaki işlemleri yapmalarını öneririz. Bilgisayar deneyimi olanlar temel kursu atlayıp birinci bölümden başlayabilirler. Kurma ve alışkanlık kazanma ile ilgili önemli bazı parçaları konunun içinde kattık,ancak CPC 464 sisteminin belirli özelliklerinin üstünde daha çok durduk ve terimlere alışkanlığınız olduğunu varsaydık.

Başlangıç bölümleri CPC 464 de bulunan bir çok özelliğe geniş kapsamlı bir rehber sağlamak amacıyla yazılmıştır. Bazı önemli temel noktalar okuyucunun kafasında kesinlik kazanması amacıyla tekrarlanmıştır. Çünkü, bilgisayar kullanmaya başlayanların çoğu klavye tanımanın ve ***BASIC*** öğrenmenin yöntemli yanlarını mümkün olduğu kadar kısa bir girişle geçip doğruca grafik ve sese doymak isteyeceklerdir.

AMSTRAD'ın ***BASIC***'e rehber yetiştirme kursu CPC 464 tünüzün bir çok yönünü, hoca oyun masası ve 'saf' bilgisayar olarak potansiyelini tam olarak anlatmaya yönelik bir yaklaşımla, geniş kapsamlı ve gösterimli bir kitap sağlamak amacıyla yazılmıştır.

Bugüne kadar bu kitabı almamışsanız almanızı kuvvetle salık veririz.

Ve nihayet kitabın sonundaki EK'ler bilgisayar kavramlarına genel bir bakış sağladığı gibi elele-özgü refarans noktalarını da içerir.

Başarılar dileriz. Bilgisayarı bu yönüyle anlamak için daha geniş potansiyelli bir bilgisayar bulamazdınız. Bilgisayarı anlamak için bilgisayar kullanmaktan daha akıllıca yol yoktur. CPC 464 özellikle kullanıcıya dost bir bilgisayardır. Daha iyi bir yatırım düşünülemez.

ÖNEMLİ

Bu kitabı okurken, programlamaya ve bilgisayarda olan ama ekranda yazılı karakter olarak sonuç vermeyen *[KEY]* tuşlara yapılan atıf yolları gösteren değişik tipte stillere ve programlama ile ilgili ama komutların bir parçası olarak yazılmaya genel tanımlamalara dikkat etmeniz gerekir

1) Daima ana kabloyu 3 iğneli fiş girişine "kurmak" başlıklı birinci kısımdaki önergelere uygun olarak bağlayın.

2) Hiç bir zaman bilgisayar klavyesini. monitörü, güçkaynağı/modülatörü bu rehberde tanımlanan donatım parçalarının dışında bir güç kaynağına bağlamayın. Bu uyarıya uymamak ciddi hasarlara yol açar ve garantiyi geçersiz yapar.

3) Vazoları, içkileri vs. bilgisayar klavyesinden, monitör ve güçkaynağı modülatörden iyice uzak tutun. Bu ünitelerden birine sıvı dökülürse ciddi hasarla sonuçlanır. Böyle durumlarda yetkili kişilere başvurun.

4) Bilgisayar klavyesi, monitör ve güçkaynağı/ monitördeki havalandırma deliklerini tıkamayın ve örtmeyin.

5) Cereyanı kesmek CPC 464'ün hafızasında depolanmış herşeyi kaybettirir. Bir programı saklamak istiyorsanız temel kursları okuduktan sonra 2. bölümü okuyun.

6) Özellikle bilgisayarla kullanılmak için yapılmış kasetleri kullanmanız salık verilir. Ancak kaliteli ses kasetlerini de, CrO_2 ya da metal olmadıkça ve 90 dakikayı geçmedikçe (C-90) kullanabilirsiniz.

7) Kaydedilen programları kolayca bulabilmeniz için C-12 (her yanı 6 dakika) kasetlerini kullanmanızı salık veririz.

8) Başka tip bilgisayar programını içeren kasetlerin CPC 464 de çalıştırılamayacağı ve yüklenemiyeceğini unutmayın.

9) Kullandığınız kasetin, kazara silinmesini önlemek için emniyet tırnakları kopartılmışsa ***record*** tuşu basmaz. Lütfen bu tuşu zorlamayın. Yoksa mekanızma zedelenebilir. Eğer emniyet kulakcıkları kopartılmış bir kaseti tekrar kayıt için kullanmak isterseniz bunu kulakcıkların deliklerine yapışkan bant yapıştırarak tıkayıp yapabilirsiniz.

10) Bir program saklamadan önce bandın iyice başa alındığına dikkat edin.

11) Ünitelerden hiçbirini doğrudan güneş alan çok sıcak, soğuk, rutubetli ya da tozlu yerlerde kullanmayın ve saklamayın. Ve çok titreşimli yerlerden uzak tutun. Program kasetlerini, oparlör ya da elektrikli motorların yakını gibi manyetik alanlarda saklamayın.

12) Kasetlerin genel bakımı ve veri kaydeden mekanızmanın düzenli temizlenmesi programlarınızın uzun ömürlü ve yanlışsız depolanmasını sağlar.

13) Donatımınızı kurcalamayın. Kullanıcının yapabileceği servis yoktur. Ünitelerde servis için yetkili elemanlara başvurun.

14) Buradaki bilgilerin tümü ya da bazıları ve bu kitapdaki programlar tekrar yayınlanamaz, adapte edilemez.

AMSTRAD CPC 464

Yeni başlayanlara temel kurs

BÖLÜM 1: KURMAK

CPC 464 sistemini kurmak ve bağlamak.

AMSTRAD CPC 464 renkli kişisel bilgisayar aşağıdakilerden birini kullanarak kurulabilir.

1- 1 AMSTRAD GT 64 Yeşil tüplü mönitör.
1- 2 AMSTRAD CTM 640 Renkli monitör.
1- 3 AMSTRAD MP 1 Modülatör/güç kaynağı ve bir (UHF) ev televizyon alıcısı.

Lütfen bilgisayar sisteminizi doğru bir şekilde kurmak ve bağlantılarını yapmak için bu konuyla ilgili bölümü okuyup işleme sonra geçin.

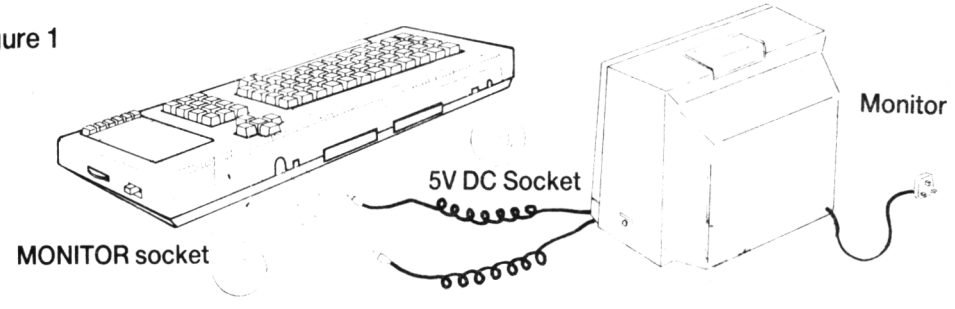
1.1 AMSTRAD GT 64 Yeşil tüplü monitör

Monitörü kutusundan çıkarın ve aşağıda anlatıldığı şekilde bağlayın. İçinden hiç bir bağlantı yapmak gerekmez, dolayısıyla donatımın içini karıştırmayın.

Bilgisayarı monitörün önüne, ana cereyan kaynağına yakın,uygun bir masanın üstüne koyunuz.

Aşağıdaki resimde gösterildiği gibi, *lead*'i daha büyük olan (6 pin *DIN*) monitörden bilgisayarın arkasında *MONİTÖR* yazan girişe bağlayın. Daha küçük *lead*'i DC power monitörden bilgisayarın arkasında 5 VDC yazan girişi bağlayın.
POWER düğmesinin *OFF* durumunda olmasına dikkat edin. Monitörden Mains girişini Mains supply (220 V AC) girişine bağlayın.

Figure 1



Şimdi monitörü açın sonra da bilgisayarı, sağ taraftaki *POWER* yazılı kayan düğmeden açın. Bilgisayarın klavye ünitesinin üst ortasındaki kırmızı *ON* ışığı yanmalıdır ve monitör aşağıdaki resmi görüntüler.

```
Amstrad 64K Microcomputer (v1)
©1984 Amstrad Consumer Electronics plc
        and Locomotive Software Ltd.

BASIC 1.0

Ready
```

Cursor

Göz yorgunluğunu önlemek için *BRIGHTNESS* (parlaklık) yazılı kontrolu rahat bir görüntü elde etmek üzere ayarlayın. Yazılar çok parlak olmamalı.

CONTRAST kontrolunu de rahat görüş sağlıyan en az ayara getirin.

GT64'ün üstünde dikey kontrol *V-HOLD* resmin ekranının ortasında kaymadan, titremeden durmasını sağlıyacak şekilde ayarlayın.

1.2 AMSTRAD CTM 640 Renkli monitör.

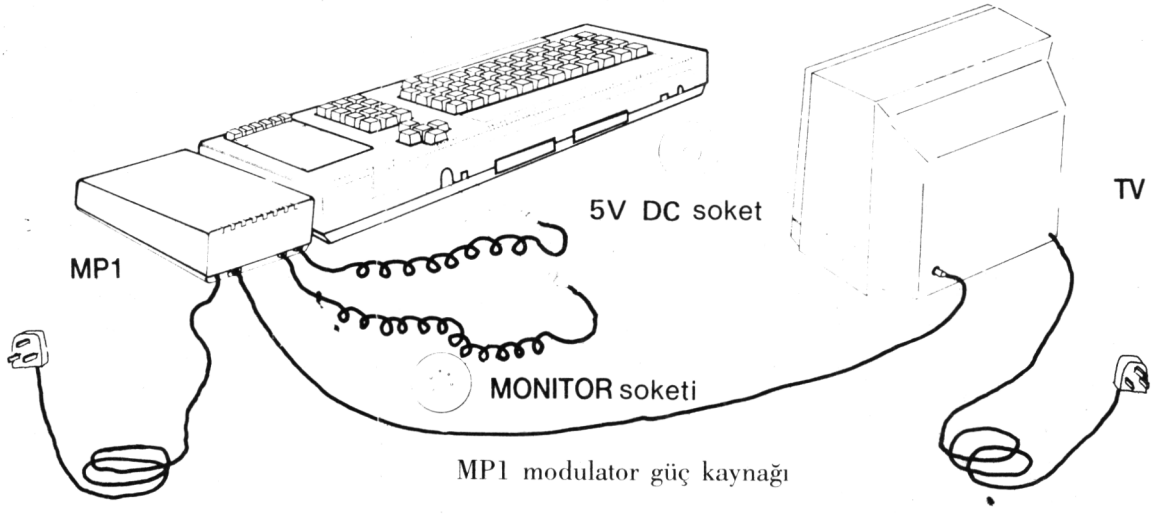
Monitörü kutusundan çıkartın, şöyle bağlayın.

* 5'li kablo monitör yazan girişe
* diğeri 5V yazan kısma.

1.3 AMSTRAD MP 1 MÖDÜLATÖR/GÜÇ Kaynağı ve bir ev Renkli T.V.alıcısı.

Halen CPC 464 bilgisayarı GT 64 yeşil tüp monitör ile kullanıyorsanız *MP1* almak isteyeceğiniz ek bir ünite olabilir. *MP1* bilgisayarınızı evdeki renkli televizyon alıcınızla birlikte kullanmanızı ve böylece CPC 464'ün renk olanaklarından faydalanmanızı sağlar.

Modülatör/güç kaynağını kutusundan çıkartın ve gösterildiği gibi bağlayın.



MP1 modulator güç kaynağı

İçerden hiçbir bağlantı yapmak gerekmez, dolayısıyla donanımın içini kurcalamayın. Modülatör/güç kaynağı *(MPY)* bilgisayarın sağına televizyon alıcısana ve cereyan fişine yakın, uygun bir masaya yerleştirin. Resim 2'de gösterildiği gibi *MP1*'den daha büyük olan fişi (6 pin DIN), bilgisayarın arkasındaki *MONITÖR* yazılı girişe bağlayın. Diğerini *MP1*'den daha küçük (DC power) fişine *MP1*'den bilgisayarın arkasında 5V DC yazılı girişe bağlayın.

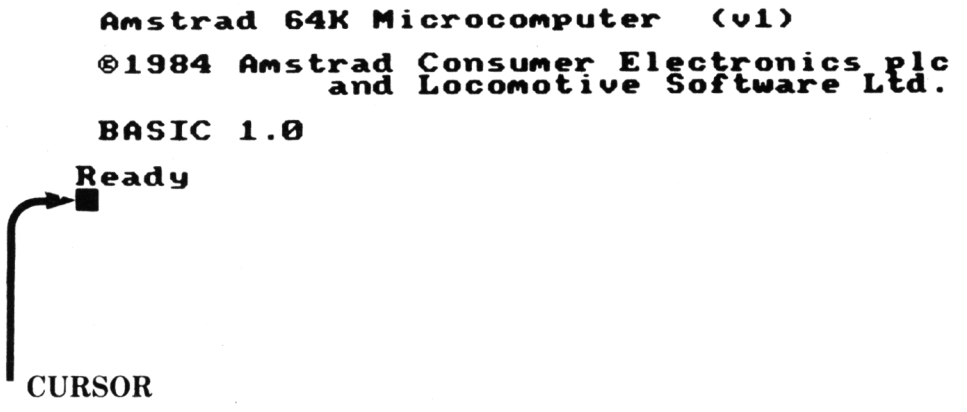
MP1'den anten fişini televizyonun anten girişine bağlayın.

Bilgisayarın sağ tarafında *POWER* yazılı anahtarın *OFF* durumunda olduğuna dikkat ettikten sonra *MP1*'den ana kabloyu cereyan fişine sokun.

Şimdi televizyonunuzun ses ayarını en aza getirin ve televizyonu açın sonra da bilgisayarı sağ taraftaki *POWER* yazılı kayan düğmeden açın.

Bilgisayarın klavyesinin üst ortasındaki kırmızı *ON* lambası yanacak ve TV'nizi bilgisayardan sinyal alacak şekilde ayarlamalısınız.

Eğer TV'niz basma düğmeli kanal seçimli ise alıcınızı kullanılmayan bir kanala getirin TV imalatçılarının önergesine göre TV'nizi aşağıdaki resim gibi bir görüntü elde edinceye kadar ayarlayın.



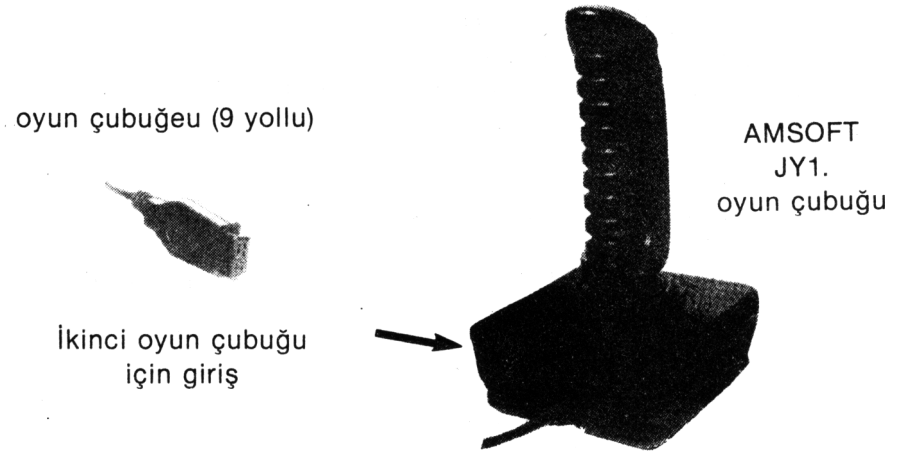
Bu resmi en net şekilde alıncaya kadar ayar yapın. Yazılar altın/sarı, fon ise koyu mavi olmalıdır.

Televizyon alıcınız döner düğmeyle program seçen tipdeyse yukarıdaki resmi görünceye kadar düğmeyi çevirin.

1.4 JOYSTİCK (OYUN ÇUBUĞU)

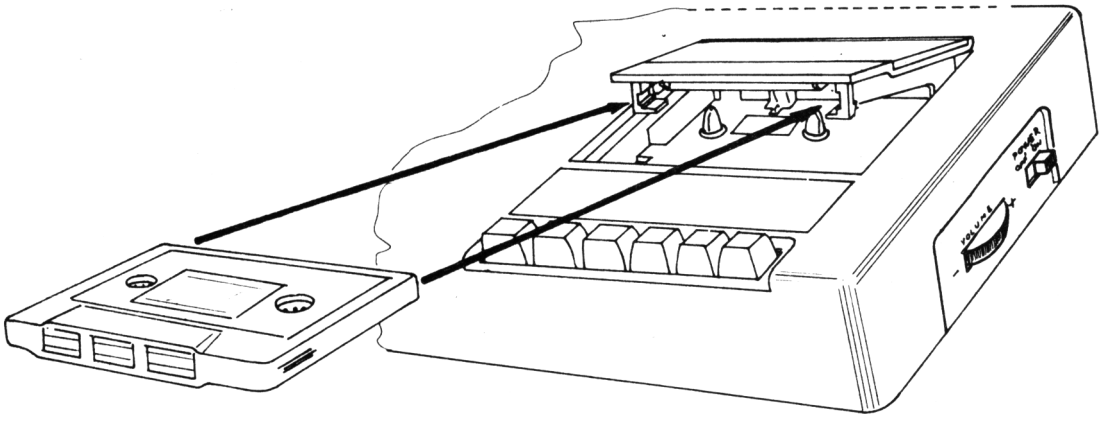
CPC 464 bilgisayarı, *joystick* (oyun çubuğu olanağı olan yazılım oyunlarıyla kullanıyorsanız. *Joystick* (oyun çubuğu) JY1 olan modeli almak isteyeceğiniz ek bir parça olabilir.

JY1 bilgisayarınızın arkasında *USER PORTS* (1Q0) 9'lu girişe bağlanabilir. Amstrad CPC 464 iki *Joystick* (oyun çubuğu) ile kullanılabilir. İkinci *joystick* birincisindeki girişe bağlanmalıdır.



1.5. Tanıtım kaseti.

Bilgisayarınızın ambalajının içinde tanıtım kaseti bulmuş olmanız gerekir. *(Data corder)* veri kaydedenin kapağını *[STOP] (EJECT)* yazılı tuşa basarak açın ve kaseti resim 3'de gösterildiği gibi *SIDE 1* (birinci tarafı üste gelecek gibi veri kaydediciye yerleştirin.



Resim 3. veri kaydediciye kaseti yerleştirmenin doğru şekli.

Kapağı kapatın ve *REW* tuşuna basın. Band başa sarılınca duracaktır. Durur durmaz *[STOP/EJECT]* tuşuna basın: *COUNTER RESET* düğmesine basarak sayaçı 000'e getirin.

[CTRL] yazılı kontrol tuşuna basın. AYNI ZAMANDA veri kaydecinin yanındaki yalnızca sayı olan tuş tablosundaki küçük *ENTER* tuşuna basın. Ekranda şu yazıları göreceksiniz.

RUN"
Pres PLAY Then any key:

PLAY'e basın sonra da herhangi bir tuşa ya da *[ENTER]*'e basın.

Teyp hareket etmeye başlayacaktır. Kısa bir süre sonra ekranda şu mesajı göreceksiniz:

Loading WELCOME 1 bloc 1
WELCOME 1 yükleniyor *bloc 1*

Teybin yüklemesi aşağı yukarı 5 dakika sürecektir. Yüklemeyi blok numaraları 2,3 vs. olmasından izleyebilirsiniz. Sonra teyp döner ve 'Hoş geldin' programı çalışmaya başlar. Rahatça oturup seyredin. Program aralıksız çalışır, izlemeniz bitince iki kez *[ESC]* tuşuna basın. Bu kaseti durdurun. Sonra *[STOP]*'a basarak kaseti çıkarıp tersini, 2. yüzünü çevirebilirsiniz.

Kasetin tersini çevirdikten sonra tekrar *[REW]* tuşuna basarak bu yüzün de en başına geldiğinizden emin olun.

[CTRL] kontrol tuşuna basın AYNI ZAMANDA bilgisayarı sağında veri alıcısına yakın yalnız sayı tuşları olan tuş tablosundaki küçük *[ENTER]* tuşuna basın. Ekranda şunları göreceksiniz:

RUN"
Press PLAY then any key:

Şimdi veri kaydedicisindeki *[PLAY]* tuşuna sıkıca basın sonra herhangi bir sayı ya da harf tuşuna ya da *[ENTER]* tuşuna ya da ara çubuğuna basın.

Teyp hareket etmeye başlıyacak ve kısa bir süre sonra şu mesaj ekranda görünecektir.

Loading WELCOME 2 block 1
(HOŞGELDİN2 *blok 1* yükleniyor.)

Ekranda verilen önergeleri izleyin. Program ilerledikçe size komutlar yazarak katılmaya davet edecektir.

1.6 BAŞKA YAZILIM KASETLERİNİ YÜKLEMEK

HOŞGELDİN KASETİ YALNIZ BİR ÖNCEKİ KISIMDA ANLATILDIĞI GİBİ YÜKLENİP ÇALIŞTIRALABİLİR. (1.5)

Korunmamış *BASIC* kasetleri aşağıdaki yöntemlerden biriyle yüklenebilir. Teybi başa almak için *REW* tuşuna bakın. Makaralar durunca hemen *[STOP EJECTE]* tuşuna basın.

Bilgisayarın hafızasını temizlemek ve onu ilk konumuna getirmek için *[CTRL]*, *[SHIFT]* ve *[ESC]* tuşlarına sırasıyla basıp bir süre tutun. Ekran temizlenip ilk açtığınız zamanki mesaj görünecektir. Aşağıdaki *[ENTER]* komutu *[ENTER]* yazan iki tuştan birine basın demektir. Klavyeden *ENTER* yazmayın. " tırnak işaretini *[SHIFT]* yazılı tuşa ve aynı zamanda 2 tuşuna basarak yazabilirsiniz.

Yazın:

Load " " ENTER

Bilgisayar sizden şunu isteyecektir.

Press PLAY then any key.

Şimdi veri kaydedicinin üstündeki *(PLAY)* tuşuna sıkıca basın ve herhangi bir sayı, harf ya da *[ENTER]* tuşlarından birine ya da ara çubuğuna basın.

Teyp hareket etmeye başlayacak ve kısa bir süre sonra ekranda şu mesaj görünecektir.

Loading (program ismi) *block 1*

Blok numaraları teyp yüklemeyi bitirinceye kadar artacak ve sonuçda ekranda şu mesaj görünecektir.

Ready

İsterseniz yüklemek istediğiniz programı belirtebilirsiniz. Bunu yapmak için yazın:

Load " (başlık) " *[ENTER]*

Bilgisayar şöyle der:

Press PLAY then any key

Şimdi *PLAY* tuşuna sonra da herhangi bir sayı, harf iki *[ENTER]* tuşundan bir ya da ara çubuğuna basın.

Teyp hareket etmeye başlar. İstediğiniz programı teybin başında değilse bilgisayar istediğiniz başlıklı programı buluncaya kadar teypi arar. Program başlığını tam olarak doğru vermeye dikkat etmelisiniz.

Eğer sizin verdiğiniz boşluğu ararken bir başka boşluk bulunursa şu mesajı yayınlar:

found (öteki başlık) *blok 1*

Bilgisayar bu programı yüklemez, tam sizin verdiğiniz başlıktaki programı aramayı sürdürür.

Sizin verdiğiniz program başlığına tam uyan başlığı buluncaya kadar ya da siz *[ESC]* tuşuna basıp aramayı durduruncaya kadar bilgisayar arar.

Program bulununca şu mesajı görürsünüz:

Loading (başlık) *blok 1*

Yükleme bitinceye kadar blok sayıları artar. Yükleme bitince şu mesaj görünür.

Ready.

O zaman yazın:

run [ENTER]

Ve yüklediğiniz program çalışmaya başlar. Eğer hafızada bir program varsa silinir ve bu yüklediğiniz program onun yerini alır.

Bir programı önceden yüklemeden doğrudan çalıştırmak isterseniz yalnızca

Run " " ENTER

yazın.

Bilgisayar şu sözlerle yanıtlar:

Press PLAY Then any key.

[PLAY] tuşuna sonra sayı, harf, iki *[ENTER]* tuşundan biri ya da ara çubuğuna bastıktan sonra, Bilgisayar önce programınızı arar sonra da programı yükler ve çalıştırır. Sizden daha fazla klavye işlemi beklemez. Bu işlemleri istediğiniz zaman *ESC* tuşuna basarak durdurabilirsiniz.

1.7. ÖNCEDEN KAYDEDİLMİŞ YAZILIM KASETLERİNİ YÜKLEMEK

Buraya kadar verilen önerilerle CPC 464 için hazırlanmış ve değişik başlıklar altında toplanmış bir çok yazılımı yükleyebilirsiniz. Ancak yine de bu yazılım paketinin üstündeki doğru yükleme önergelerine bakın ve bunları uygulayın.

1.8. (SAVE) KAYDETME

Bir program daha ilerde kullanılmak üzere saklanabilir (kaydedilebilir). Kaseti doğru bir şekilde yerleştirip kapağını kapatın. (Kayıttan koruma tırnaklarının kapatılmamış olmasına dikkat edin). *[REW]* tuşuna basarak teybi boşa almayı ve makaralar durur durmaz *[STOP/EJECT]* tuşuna basmayı unutmayın ve yazın:

Save "Programın başlığı" *[ENTER]*

Bilgisayar şöyle yanıtlar:*Press REC PLAY then any key:*

Şimdi veri kaydedicisinde *[REC]* ve *[PLAY]* tuşlarına elinizi çektiğinizde de basılı kalacak şekilde basın ve bir de herhangi bir tuşa basın. (sayı, harf *[ENTER]* tuşlarından biri ya da ara çubuğu.)

Bilgisayar şöyle yanıtlar:

Saving (program başlığı) *block 1*

Programı saklandığında (kaydedildiğinde) makaralar durur ve ekranda *Ready* sözcüğünü görürsünüz. Şimdi *[STOP/EJECT]* tuşuna basın: Programınız kaydedilmiştir.

Önceden kaydedilmiş yazılım ve oyunları kendi kasetinize kayıt yapamazsınız. Bu programlar izinsiz kopye çıkartılmasına karşı korunmuştur.

BÖLÜM 2:

KLAVYE:

Burada bilgisayarlardaki bazı tuşların fonksiyonlarını açıklayacağız. Bilgisayar kullanmakta deneyimi olanlar bu bölümü atlıyabilirler.

[ENTER]

İki ***[ENTER]*** tuşu vardır. Bunların ikiside yazdığınız bilgilerin, bilgisayar tarafından işleme konulması için kullanılır. Her ***[ENTER]*** tuşuna bastığınızda ekranda yeni bir satır başlar. Bilgisayara gireceğiniz her yeni bilgi için ***[ENTER]***a basmanız gerekir.

Bundan sonra her ***[ENTER]*** sözcüğünü gördüğünüzde yazılanların arkasından ***[ENTER]*** tuşuna basmanız gerektiğini anlamalısınız.

[DEL]

Bu tuş ekranda kürsörün solundaki karakteri silmek için kullanılır.

Sırasıyla a, b, c, d yazın: d harfinin kürsörün solunda kaldığını göreceksiniz. d harfini istemediğinize karar verdiğinizde ***[DEL]*** tuşuna bir kez basın, d harfinin kaybolduğunu göreceksiniz. Parmağınızı çekmeden ***[DEL]*** tuşuna basmayı sürdürürseniz c, b, a da sırasıyla kaybolacaktır.

[SHIFT]

Bu tuştan da bilgisayar üzerinde iki tane vardır. Bunlardan birine basıp, parmağınızı kaldırmadan başka bir tuşa bastığınızda, eğer o tuş bir harfi gösteriyorsa o harfin büyüğü, eğer üst kısımdaki tuşlardan biriyse tuşun üst tarafında görünen karakter ekranda yazılacaktır.

A tuşuna basın, parmağınızı ***[SHIFT]***'e basıp, tekrar A tuşuna basın. Ekranda:

a A
göreceksiniz.

Şimdi biraz ara vermek için ara çubuğuna basın ve üst sıradaki tuşlarından 2 rakamının bulunduğu tuşa basın. Sonra ***[SHIFT]***'e basarak tekrar aynı tuşa basın.

Ekranda:
2 "
göreceksiniz.

Artık *[SHIFT]* tuşu ile başka bir tuşa basıldığında nasıl bir sonuç alacağınızı biliyorsunuz. Ama alışmak için başka tuşlara da önce yalnız, sonra *[SHIFT]* ile basın ve sonucu görün.

[CAPS LOCK]

Bu *[SHIFT]*'e benzer bir işlemdir. Ancak bu tuşa bir kez basmak yeterlidir. Bu tuşa bir kez bastıktan sonra yazacağınız bütün harfler büyük harf olarak çıkacaktır. Ancak rakkamlar değişmiyecektir. Bir denemesini yapın *[CAPS LOCK]*'a basın ve A B C D E F 1, 2, 3, 4, 5, 6 tuşlarına basınız.

Ekranda harflerin büyük harf olarak yazıldığını , ama rakamların sembollere dönüşmediğini göreceksiniz. Eğer *[CAPS LOCK]* durumunda rakkam tuşlarının üst kısmındaki sembolleri yazmak isterseniz *SHIFT* tuşuna basın ve parmağınızı kaldırmadan şunları yazın.

ABCDEF 1 2 3 4 5 6
Ekranda:

ABCDEF!" # $ % & ' göreceksiniz.

Eğer yeniden küçük harflere dönmek isterseniz [CAPS LOCK]'a bir kez daha basın. Eğer tüm büyük harf hem de tuşların üst kısmındaki sembolleri sürekli *[SHIFT]* tuşuna basmaksızın kullanmak isterseniz bunu *[CTRL]* tuşuna basarken bir yandan da *[CAPS LOCK]*a basarak yapabilirsiniz. Bunun da bir denemesini yapın.

ABCDEF 1 2 3 4 5 6 tuşlarına basın

Ekranda **A B C D E F !" # $ &** göreceksiniz. Hem [CTRL] hem de *[CCAPS LOCK]* çalışır durumundayken rakkam yazmak olanağınız da vardır. Ana klavyenin sağındaki tuşları kullanabilirsiniz.

Tuşların alt kısmını kullanmak istediğinizde *[CTRL]*'ye basarak *[CAPS LOCK]*'a bir kez basın. Küçük harflere dönmek için *[CAPS LOCK]*'a bir kez daha basın.

[CLR]

Bu tuş kürsörün içindeki harf veya rakkamı silmek için kullanılır.

A B C D E F G H yazın. Kürsör son harfin **(H)** sağında yer alacaktır.

Şimdi sol kürsör tuşuna dört kez basın. **[←]** kürsör dört karakter geri gidecek ve **E** harfinin üstüne gelecektir. Şu anda kürsörün içinde **E** harfi görülür. Ama *[CLR]* tuşuna bastığınızda **E** harfi silinecektir. **FGH** ise birer hane geri gelecektir. Bu durumda **F** kürsörün içinde olmalıdır. Parmağınızı *[CLR]* tuşundan çekmeden basın. **FGH** harflerinin sırayla silindiğini göreceksiniz.

[ESC]

Bu tuş bilgisiyarın yapmakta olduğu işlemi durdurması için kullanılır. Bu tuşa bir kez basmak bilgisayarı bir tuşa basılana kadar durdurur.

[ESC] tuşuna iki kez basarsanız bilgisayar yapmakta olduğu işlemi tamamen durdurur ve sizden yeni talimat almaya hazır durumuna gelir.

Şimdi *[ESC]* tuşuna iki kez basın.

ÖNEMLİ:

Ekranda soldan sağa 40 harflik yer vardır. Bu sayıdan fazla harf yazarsanız bunlar otomatik olarak ikinci satın başına yazılır. Bu durumda *[ENTER]* tuşuna **BASMAYIN.**

Bilgisayar bunu otomatik olarak yapmıştır. Fazladan *[ENTER]*'a basmak ya hemen, ya da program çalışırken hata mesajına (çoğu kez, **Syntax error yazılım hatası)** yol açar.

SYNTAX ERROR (yazılım hatası)

Ekranda **syntax error** mesajı görünürse, Bilgisayar size verdiğiniz talimatı anlamadığını belirtiyor demektir.

Örnek:

Prıntt [ENTER] yazdığınızda:

Ekranda **Syntax Error** mesajı göreceksiniz.

Bu mesaj bilgisayar **print** yerine iki t'li **prınt** sözcüğünün anlamını çıkarmadığı için ekrana gelmiştir. Eğer bir program satırında hata yaparsanız örneğin:

10 prıntt"a b c" [ENTER] yazarsanız **Syntax error** mesajı, bilgisayar programı çalıştırırken verdiğiniz komutu işleme sokuncaya kadar ekranda görünmez.

Şimdi **run [ENTER]** yazın.

Ekranda:

Syntax Error , in 10
10 prıntt"a b c"

göreceksiniz.

Bu mesaj size hatanın hangi satırda olduğunu belirtiyor ve program satırı ile birlikte hatayı düzeltmek için kürsörü de gösteriyor.

Sağ kürsör tuşuna (→) kürsör *prınt*'deki t harfine gelinceye kadar basın. Sonra da *[CLR]* tuşuna basarak fazla t yi silin, düzeltilmiş satırı *[ENTER]*'a basarak bilgisayara girin.

run [ENTER] yazın.

Bilgisayar komutu kabul edecek ve **a b c** yazacaktır.

AMSTRAD BASIC ile tanışma

8. Bölümde Amstrad Basıc'daki bütün temel komutları ayrıntılı tanımlamalarla bulacaksınız. Bu bölümde daha yaygın kullanılan *Basıc* komutları tanıtacağız.

CLS

CLS (ekran temizleme) yazın:

Büyük harf ya da küçük harf kullanabilirsiniz. *(ENTER)*'e basın. Ekranın temizlendiğini ve sol üst köşede kürsör ile ***Ready*** (hazır) sözcüğünün belirdiğini göreceksiniz.

PRINT

Bu sözcüğü, program içinde yazdığınız kelime ve sayıların ekrana yazılmasını istediğiniz zaman kullanacaksınız.

Aşağıdaki satır yazın.

Prınt "merhaba" *[ENTER]*

Ekranda merhaba göreceksiniz.

Tırnak işaretleri " " bilgisayara neyin yazılması gerektiğini belirtmek için kullanılır. (Merhaba) sözcüğü *[ENTER]* tuşuna basılır basılmaz ekrana çıkmıştır. Şimdi *CLS [ENTER]* yazarak ekranı temizleyin.

RUN

Bir önceki örnek tek satırlık bir programı gösteriyordu. Oysa çoğu programlar bir çok satırdan oluşur. Her satırın önüne bir sayı yazılır. Bu sayılar bilgisayara programın çalışma sırasını belirtir. *[ENTER]* tuşuna basıldığında satır, program çalışıncaya kadar belleğe alınır.

Örneğin: aşağıdaki satırı yazın.

10 prınt "Merhaba" *[ENTER]*

görüyorsunuz ki *[ENTER]*'e basınca merhaba çıkmadı. Ekrana çıkabilmesi için ***RUN*** sözcüğünün yazılması gerekir.

RUN [ENTER]

Şimdi ekranda merhaba göreceksiniz. Sürekli prınt yazacak yerde ? işareti de kullanabilirsiniz. Örneğin:

10 ? "Merhaba" *[ENTER]*

LIST

Program belleğe alındıktan sonra yazılanları kontrol etmek için program listesi istenebilir.

Lıst [ENTER] yazın.

Ekranda bellekte bulunan

10 PRINT "Merhaba" program satırını göreceksiniz.

PRINT Sözcüğünün büyük harfler ile çıktığı dikkatinizi çekti mi? Bu bilgisayarın *PRINT*'i *BASIC* komutu olarak kabul ettiğini gösterir.

Şimdi *CLS [ENTER]* yazarak ekranı temizleyin. *CLS [ENTER]* yazdığınızda ekran temizlendiği halde programınız hafızadan silinmemiştir.

GOTO

Goto komutu bilgisayara bir satırdan bir başka satıra, ya da bazı satırları atlamak ya da bir çevrim oluşturmak için geçmesini söylemeye yarar. Denemek için şunları yazın.

1Ø *Print* "Merhaba" *[ENTER]*

2Ø *GOTO 1Ø [ENTER]*

RUN [ENTER]

Ekranda Merhaba sözcüğünün solda alt alta devamlı yazıldığını göreceksiniz. Bu programı durdurmak için *[ESC]* tuşuna bir kez basın. Tekrar başlatmak için herhangi bir tuşa basın. Başka komut verebilmek üzere bunu tamamen durdurmak için *[ESC]* tuşuna iki kez basın.

CLS [ENTER]

tuşuna basarak ekranı temizleyin.

Merhaba sözcüğünü sürekli yanyana yazarak ekranı doldurmasını isterseniz bir önceki programı yeniden yazın. Fakat, tırnak işaretlerinden sonra bir noktalı virgül koyun.

1Ø *Print* "Merhaba"; *[ENTER]*

2Ø *Goto* 1Ø *[ENTER]*

Run [ENTER]

Böylece noktalı virgül ; bilgisayara daha sonra ilk yazacağı karakteri bir öncekinin hemen arkasına koymasını belirtmiş oluyor. Bu programı durdurmak için de *(ESC)* tuşuna iki kez basın. Şimdi yine satır 1Ø'u yazın ama, bu kez noktalı virgül ; yerine virgül , koyun.

1Ø *Print* "Merhaba", *[ENTER]*

Run [ENTER]

Görüyorsunuz ki virgül, bilgisayara sonraki karakteri bir öncekinin başlangıcından 13 kolon ara vererek yazmasını belirtmiş oluyor. Bu özellik bilgileri ayrı sütunlarda yazabilmek açısından faydalıdır. Ancak dikkat edin, eğer ilk grupdaki karakterin sayısı 12'yi geçerse sonradan gelen grup ayrıcı 13 kolon ara vererek yazılır.

Bu programı da durdurmak için *[ESC]* tuşuna 2 kez basın. Bilgisayarın hafızasını tamamen temizlemek için ise *[SHIFT] [CTRL]* ve *[ESC]* tuşlarına aynı anda basarak, bilgisayarı ilk konumuna getirin.

INPUT

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

Örneklemek için şöyle yazın:

1Ø *Input* "kaç yaşındasınız"; yaş *[ENTER]*

2Ø *Print* "Bence siz" ;yaş; "yaşında göstermiyorsunuz" *[ENTER]*

RUN [ENTER]

Ekranda:

Kaç yaşındasınız? göreceksiniz.

Şimdi yaşınızı yazın ve *[ENTER]*'e basın. Eğer yaşınız 18 ise ekranda şunu göreceksiniz. Bence siz 18 yaşında göstermiyorsunuz.

Bu örnek *input* emrinin ve bir sayısal değişkenin kullanımını gösteriyor. Yaş sözcüğü 10.

satırın sonunda hafızaya alınmış ve böylece bilgisayarın yaş sözcüğü ile yazılan herhangi bir sayı arasında bağlantı kurması ve bu sayı ya da sayıları 20. satırda yaş sözcüğün yerine yazması sağlanmış oluyor. Biz yukardaki örnekte yaş sözcüğünü yaş değişkeni için kullandıksa da bunun yerine herhangi bir harf da kullanabilirdik. Örneğin: b

Hafızayı temizleyerek bilgisayarın ilk konumuna getirin (***[CTRL] [SHIFT]*** ve ***[ESC]*** tuşları.)

Eğer girişin harf veya harflerle rakkamlardan oluşmasını isterseniz değişkenin sonunda dolar $ kullanmak gerekir.

Aşağıdaki programı yazın. Yalnız dikkat edin 20. sütunda Merhaba'nın a'sıyla benim'in b'si arasına mesafe koymanız gereklidir.

1◊ ***Input*** "adınız ne" ; ad$ ***[ENTER]***

2◊ ***Print*** "Merhaba"; ad$ "benim adım Amstrad" ***[ENTER]***

RUN (ENTER)

Ekranda

Adınız ne? göreceksiniz.

Adınızı yazın.

Eğer örneğin adınız Selim ise, ekranda Merhaba Selim, benim adım Amstrad göreceksiniz.

Yukarıda karakter dizi değişkeni için ad$ kullandık ama, burada da rahatlıkla bir harf kullanabilirdik. Örneğin a$. Şimdi yukarıdaki 2 örneği bir programla birleştirebiliriz.

[CTRL] [SHIFT] ve [ESC] tuşlarına basarak bilgisayarı yeniden ilk konumuna getirin.

5 ***CLS [ENTER]***

1◊ ***Input*** "adınız ne" ; a$ ***[ENTER]***

2◊ ***Input*** "yaşın kaç" ; b ***[ENTER]***

3◊ ***Print*** "Doğru söylüyorum" ; a$; "hiç de" ; b ; yaşında göstermiyorsun" ***[ENTER]***

RUN [ENTER]

Bu proğramda 2 değişken kullanıldı, a$ isim için, b'de yaş için.

Ekranda şunu göreceksiniz.

Adınız ne?

Şimdi adınızı yazın (Örnek: Selim) ve ***ENTER***'e basın ... Şimdi size şu sorulacak.

Yaşın kaç

Şimdi yaşınızı yazın (Ör. 18) ve ***[ENTER]***

Eğer adınız Selim, yaşınız da 18 ise ekranda şunu göreceksiniz. Doğru söylüyorum Selim, hiç de 18 yaşında göstermiyorsun.

PROGRAM DÜZELTME

Bir proğramda satırlardan birisi yanlış yazılmış ve ***Syntax Error*** ya da bir başka hata mesajı alınmışsa hatalı satırı tekrar yazmaktansa düzeltmek mümkündür. Örneklemek için önceki programı bir de hatalı yazalım.

5 CLSS

1Ø ***Input*** "Senin adı ne" ; a$ ***[ENTER]***

2Ø ***Input*** "Kaç yaşındasın"; b ***[ENTER]***

3Ø ***Print*** "doğru söylüyorum"; a$; "hiç de"; b; "yaşında durmuyorsun" ***[ENTER]***

Yukarıda yazılı programda üç yanlış var.

5. satırda ***CLS*** yerine ***CLSS*** yazdık.

1Ø. satırda adın yerine adı yazdık.

3Ø. satırda söylüyorum-dan sonra ara koymadan tırnak işareti koyduk.

Bir programı düzeltmenin üç yolu vardır. Birincisi hatalı satırı yeniden yazmaktır. Yeniden yazılan satır en son hafızada olan aynı numaralı satırın yerini alır.

İkincisi ***EDİT*** metodudur. Bir de kürsör kopya metodu vardır.

EDIT METODU

5.satırdaki hatayı düzeltmek için ***EDIT*** 5 ***[ENTER]***

5.satır, kürsör 5'in üstünde olarak ekrana gelecektir. Fazla S'i çıkartmak için sağ kürsör tuşuna (→) kürsör *S* harfinin üstüne gelinceye kadar basın. Sonra da ***[CLR]*** tuşuna basın. *S* harfinin silindiğini göreceksiniz. Şimdi ***[ENTER]***'e basın. 5. satır hafızada düzeltilmiştir

LIST [ENTER]

yazarak. 5. satırı kontrol edin.

KÜRSÖR KOPYA METODU

1Ø. ve 3Ø satırları düzeltmek için ***(SHIFT)*** tuşuna basarak, parmağınızı çekmeden yukarı kadar (↑) tuşuna, kürsör ekranda 1Ø. satırın başına gelinceye kadar basın. Dikkat ederseniz esas kürsör yerinden oynamamıştır ve şu anda ekranda 2 kürsör bulunuyor. Şimdi de ***(COPY)*** tuşuna copy kürsörü, adı ile ve arasındaki boşluğa gelinceye kadar basın.

Gördüğünüz gibi 1Ø. satır en alt satırda yeniden yazılıyor ve ana kürsör ***copy kürsör***'le aynı yerde duruyor. Şimdi ***n*** harfini koyun. bu sadece en alt satırda görünür.

Bu kez ***copy*** kürsörü olduğu yerde kalmış ve esas kürsör hareket etmiştir. Şimdi ***[COPY]*** tuşuna 1Ø. satırın tümü yazılıncaya kadar basın. Sonra da ***[ENTER]*** tuşuna basın. Yeni 1Ø. satır hafızaya geçmiştir. ***Copy*** kürsörü yok olur ve esas kürsör yeni 1Ø. satırın altında yer alır. İkinci hatayı düzeltmek için ***[SHIFT]*** tuşuna basarak yukarı kürsör (↑) tuşuna ***copy*** kürsörü 3Ø. satırın başına gelinceye kadar basın.

[COPY] tuşuna da copy kürsörü söylüyorum dan sonraki tırnak işaretinin üstüne gelinceye kadar basın. Şimdi ara çubuğuna bir kez dokunun, en alt sırada boşluk olacaktır.

3Ø. satırın tamamı yazılıncaya kadar *[COPY]* tuşuna basın. Sonra da *(ENTER)* tuşuna basın. Artık liste isteyerek hafızada düzeltilen programı kontrol edebilirsiniz. Şöyle:

LIST [ENTER]

Şimdi bilgisayarı ilk durumuna getirmek için *[CTRL] [SHIFT] ve [ESC]* tuşlarına basın.

IF THEN

Şimdi yukarıdaki programı *IF* ve *THEN* emirleriyle genişletebilirsiniz. Aşağıdakileri yazın. Dikkat ederseniz iki yeni işaret daha ekledik.

< daha küçük demektir. Klavyede M tuşunun yanındadır. > ise daha büyük demektir. < (daha küçük) tuşunun yanındadır.

5 CLS (ENTER)

1Ø Input "Senin adın ne" ; a$ [ENTER]
2Ø Input "Kaç yaşındasın" ; yaş [ENTER]
3Ø if yaş <13 then 7Ø [ENTER]
4Ø if yaş >2Ø then 6Ø [ENTER]
5Ø if yaş <21 then 8Ø [ENTER]
6Ø Print a$; "sen" ; yaş ; "yaşında olduğuna göre artık çocuk sayılmazsın" [ENTER]
65 END [ENTER]
7Ø Print a$; "sen"; yaş; "yaşında henüz ufak sayılırsın": END [ENTER]
8Ø Print a$; "sen"; yaş; "yaşındasın. Oldukça büyümüşsün" [ENTER]

Programın doğru yazıldığını kontrol için:

List [ENTER]

Şimdi de:

RUN [ENTER]

Şimdi bilgisayarın sorduğu soruları cevaplayın ve sonucu görün.

If ve *Then* emirlerinin programdaki etkisini gördünüz. Bir de *END* sözcüğünü ekledik. *END* sözcüğü programın akışını durdurmak için kullanılır. Eğer 65. satırda *END* olmasaydı programı 7Ø ve 8Ø. satırları uygulamaya geçerdi.

Aynı şekilde 7Ø. satırın sonunda da olmasaydı program 8Ø. satırı uygulamaya geçerdi. *END* sözcüğünden önceki iki nokta üst üste: *END*'i bir önceki komuttan ayırıyor.

İki nokta üstüsteyi kullanarak bir program satırına birden fazla komut yazabilirsiniz. Bir de 5. satırı program çalışmadan önce ekranı temizlemek için kullandık.

Bu ekrandaki görüntünün düzenli olması için bundan sonraki programlarda da yer alacaktır.

If Then komutuyla ilgili başka sözcükler arasında *ELSE, OR* ve *GOTO* da vardır. Bu kitapla çalışmayı sürdürdükçe bu sözcüklerin kullanımı kafanızda aydınlığa kavuşacaktır. Şimdi *(CTRL) (SHIFT)* ve *(ESC)* tuşlarına basarak bilgisayarı ilk konumuna getirin.

FOR...TO/NEXT

Şimdi *FOR, TO* ve *NEXT* emirlerinin kullanılışını göreceğiz. Bunun için bilgisayar ile 12'li çarpım cetvelini oluşturacağız. Şimdi aşağıdakileri yazın. * işaretinin çarpma olduğuna dikkat edin.

5 CLS [ENTER]

1Ø for a = 1 to 2Ø (ENTER)
2Ø Print a"* 12 =" a *12 [ENTER]
3Ø Next a [ENTER]

RUN [ENTER]

Sütunlar biraz düzensiz görünüyor. Şöyle yazın.

5 CLS [ENTER]

1Ø FOR a = 1 to 9 [ENTER]
2Ø PRINT a" * 12 = "a * 12 [ENTER]
3Ø NEXT a [ENTER]
4Ø FOR a= 1Ø to 2Ø [ENTER]
5Ø PRINT a" * 12 = * 12 [ENTER]
6Ø NEXT a [ENTER]

RUN [ENTER]

Bu çarpım cetveli programını başka sayılarla da deneyin. Örneğin 17'nin çarpım cetvelini görmek için 2Ø. ve 5Ø. satırları 12 sayısının yerine 17 koyarak değiştirin. Şimdi *[CTRL] [SHIFT] ve [ESC]* basarak bilgisayarı ilk konumuna getirin.

FOR TO NEXT emirlerinde her atılan adımın değerini belirlemek *STEP* emirleriyle mümkündür. Bu konuda daha geniş bilgi için 8. bölümdeki *FOR* kullanımını inceleyin.

BASİT ARİTMETİK

CPC 464 rahatlıkla hesap makinası olarak kullanılabilir.

Bu işlemi daha iyi anlamak için aşağıdaki örnekleri yapın. Bu kısımda print yerine ? işaretini kullanacağız. Cevaplar *[ENTER]* tuşuna basar basmaz yazılır.

TOPLAMA:

(Artı için [SHIFT] ve ; kullanın)
? 3+3 [ENTER]
6

Farkındaysanız, = işareti kullanmıyorsunuz.
? 8 + 4 [ENTER]
12

ÇIKARTMA:

(*Shıftsız* = işaretini eksi için kullanın)
? 4—3 [ENTER]
1

? 8—4 [ENTER]
4

ÇARPMA:

(SHIFT) ve : işaretini çarpma için kullanın.
* çarpma işareti yerine geçer.
? 3 * 3 (ENTER)
9
? 8 * 4 (ENTER)
32

BÖLME:

(Shıftsız ? işaretini bölmek için kullanın./ bölü işareti anlamındadır.
? 3/3 (ENTER)
1
? 8/4 (ENTER)
2

KARE KÖK:

Bir sayının karekökünü bulmak için ***sqr*** ( ) yazın ve iki parantezin arasına karakökünü istediğiniz sayıyı koyun.
? sqr (16) [ENTER] (bunun anlamı $\sqrt{16}$'dır)
4
? Sqr (100) [ENTER]
10

ÜS ALMA İŞLEMİ

? 3 ↑ 3 (ENTER) (bu 3^3 demektir)
? 8 ↑ 4 [ENTER] (Bu 8^4 demektir)
4096

KÜP KÖK:

Küp kök işlemi son üs alma örneğinde kullanılan metoda benzer. Bu yöntemle kolaylıkla hesaplayabilirsiniz.

Örnek:

27'nin küp kökünü bulmak için

? 27 ↑ (1/3) [ENTER]
3

125'in küp kökünü bulmak için yazın.

? 125 ↑ (1/3) [ENTER]
5

KARIŞIK HESAPLAR (+, —, *, /)

Bilgisayarla karışık sıralı hesaplarda yapmak mümkündür. Ancak bunları bazı öncelikler çerçevesinde hesaplanır.

İlk öncelik bölmelere verilir, sonra çarpma, toplama ve çıkartma gelir. Bu öncelikler yalnızca bu işlemleri içeren hesaplarda geçerlidir. Öncelikler daha sonra da açıklanacaktır, ve üs alma vs. de içerecektir.

Yapılacak hesap aşağıdaki gibiyse:

3 + 7 — 2 * 7/4

Bunun şöyle hesaplanacağını düşünebilirsiniz.

3 + 7 — 2 = 7/4
= 8 * 7 / 4
= 56 / 4
= 14

Aslında şu şekilde hesaplanmıştır.

3 + 7 — 2 * 7/4
= 3 + 7 — 14/4
= 3 + 7 — 3.5
= 6.5

Bunu kanıtlamak için bu hesabı bilgisayara verin.

? 3 + 7 — 2 * 7/4 [ENTER]
6.5

Bilgisayarın hesaplama yolunu parentezler ekleyerek değiştirebilirsiniz. Bilgisayar parentez içindeki hesapları parentez dışındaki çarpma, bölme vs.den önce yapacaktır. Şimdi bunu doğrulamak için bu hesabı parentez kullanarak yapın.

? (3 + 7 — 2) * 7/4 (ENTER)
14

İLERİ İŞLEMLER

Eğer hesaplarınızda çok büyük ya da çok küçük sayılar kullanmak istiyorsanız bazen matematiksel özel semboller kullanmak yararlıdır. 10'un katı olan sayıların üssü için küçük ya da büyük e E harfi kullanılır.

Orneğin 300 ile $3 * 10^2$ aynı sayıyı gösterir. Bunun üslü gösterimi ise 3E2 dir. Aynı şekilde 0.03, $3 * 10^{-2}$ ile aynıdır. Bu yazılımın üslü gösterimi de 3E-2 dir. Aşağıdaki örnekleri deneyin.

3Ø * 1Ø

Şöyle yazabilirsiniz:

? 3Ø * 1Ø [ENTER]
3ØØ

ya da şöyle yazabilirsiniz.

? 3 e 1 * 1 e 1 [ENTER]
3ØØ
3ØØØ * 1ØØØ
? 3 e 3 * 1 e 3 [ENTER]
3ØØØØØ
3000 * 0.001
? 3 e 3 * 1 e — 3 [ENTER]
3

BÖLÜM 3

GRAFİK VE SES ÖZELLİKLERİ:

Amstrad CPC 464'ün üç tane ekran modu vardır.
MOD 0, MOD 1 ve MOD 2
Bilgisayar ilk açıldığında otomatik olarak MOD 1 açılmış olur.
Değişik modları anlamak için bilgisayarı açın ve 1 tuşuna basın. Ekranda iki satır 1'lerle doluncaya kadar parmağınızı tuşdan çekmeyin. Bir satırda olan birleri sayarsanız 40 tane olduğunu görürsünüz. Demek ki MOD 1 de 40 kolon var. ***[ENTER]*** tuşuna basarsanız ***Syntax Error*** mesajını alırsınız. Ama, merak etmeyin bu yalnızca Ready mesajı alarak bilgisayarı yeni komutlara hazırlamanın kestirme bir yoludur.

Şimdi şunu deneyin.

MODE 0 |ENTER|

Karakterin daha büyük olduğunu göreceksiniz. 1 tuşuna basın ve ekranda iki satır tamamen 1 ile doluncaya kadar parmağınızı çekmeyin. Bir sıradaki 1 leri sayarsanız 20 tane olduğunu görürsünüz. Demek ki Mod 0 da 20 kolon var. Şimdi yine |ENTER| tuşuna basın.

Şimdi de: Mode 2 |ENTER|

görüyorsunuz ki Mod 2 en küçük karakterleri içerir. Bu sefer sürekli 1 tuşuna bastığınızda tek sutunda 80 tane 1 sayabilirsiniz. Yani Mod 2'de 80 kolon vardır.

Tekrar bakalım.

MOD 0 = 20 kolon
MOD 1 = 40 kolon
MOD 2 = 80 kolon

Son olarak tekrar |ENTER| tuşuna basın.

RENKLER

27 renk seçeneği vardır. Bunlar yeşil monitör de (GT.64) yeşilin değişik tonları olarak gösterilmiştir. Eğer GT 64 monitörü aldıysanız bilgisayarın renk olanaklarını evdeki renkli TV'de kullanabilmek için AMSTARD MP 1 modulatör güç kaynağı satın alabilirsiniz.

MOD 0'da eldeki 27 renkten 16'sı istediğiniz zaman ekrana getirebilirsiniz.

MOD 1'de 27 renkten 4'ünü seçme olanağınız vardır.

MOD 2'de 27 renkten 2'sini kullanabiliriz.

BORDER, ***PAPER*** ve ***İNK*** komutları ile ekranda oluşan çerçevenin, ekranın ve yazının renklerini değiştirmek mümkündür.

RENK NO	RENK	RENK NO	RENK
0	Siyah	14	Pastel mavi
1	Mavi	15	Turuncu
2	Parlak mavi	16	Pembe
3	Kırmızı	17	Pastel magenta.
4	Magenta (Çingene pembesi)	18	Parlak yeşil
5	Eflatun	19	Deniz yeşili
6	Parlak kırmızı	20	Parlak Cyan
7	Mor	21	Ham limon yeşil
8	Parlak magenta	22	Pastel yeşili
9	Yeşil	23	Pastel Cyan
10	Açık mavi (Cyan)	24	Parlak sarı
11	Gök mavisi	25	Pastel sarı
12	Sarı	26	Parlak beyaz
13	Beyaz		

TABLO 1:

RENK KOD NUMARALARI VE RENKLER:

Daha önce de belirtildiği gibi bilgisayar ilk açıldığında Mod 1'dedir. Mod 1'e dönmek için:

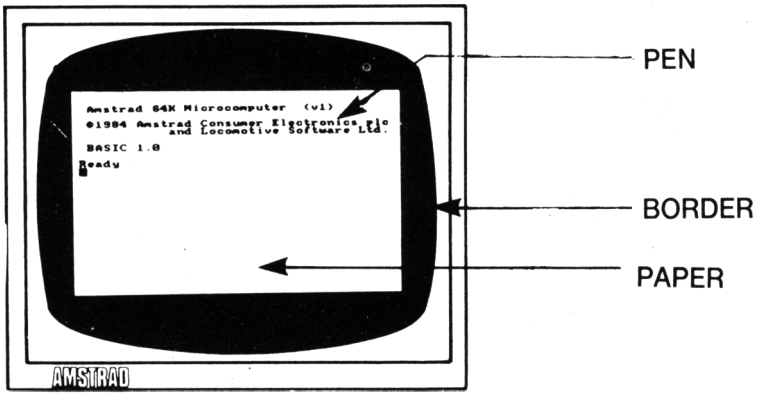
Mode 1 ***[ENTER]***

Bilgisayar açıldığında ***BORDER*** (çerçeve), ***PAPER*** (ekran), ***PEN*** (yazı) renkleri şöyledir.

BORDER (çerçeve) 1 numaralı renk (mavi)

PAPER (ekran) 1 numaralı renk (mavi)

PEN (karakterler) 24 numaralı renk (Parlak sarı)



BORDER kısmı *PAPER* bölümünü çevreleyen kısımdır. (dikkat ederseniz bilgisayar ilk açıldığında *BORDER*'da *PAPER*'da mavidir.)

PAPER, *BORDER*'in içinde kolon ve karakterlerin gözüktüğü ekran kısmıdır. *PEN* karakterlerin rengidir. Bunu daha iyi açıklamak için monitördeki *PEN* (kalem) ve *PAPER* (kağıt) normal kalem kağıda benzetebiliriz. Bir kalemdeki mürekkebin renkleri nasıl değiştirilirse *PEN*'deki karekterlerin rengi de değiştirilebiliyor. Aynı şekilde kağıdın rengini değiştirebildiğimiz gibi ekranda *PAPER*'in rengini de değiştirmek mümkündür.

BORDER'ın rengini değiştirmek için:

*border 0 [ENTER]*yazın

Border renginin maviden siyaha değiştiğini göreceksiniz. Tablo 1'e bakarsanız siyah rengin numarasının 0 olduğunu göreceksiniz. Yine tablodaki renklere göre *BORDER*'ın rengini istediğiniz renk yapabilirsiniz. Border yazdıktan sonra yanına istediğiniz rengin numarasını yazın.

Şimdi:

CLS [ENTER]

yazarak ekranı temizleyin.

PAPER'in rengini değiştirmek için:

PAPER 2 [ENTER]

Ready yazısının fonunun parlak mavi olduğunu göreceksiniz.

Şimdi:

CLS [ENTER]

yazarak ekranı yeni *PAPER* rengine göre temizleyin.

PEN rengini değiştirmek için:

PEN 3 [ENTER]

PEN'in renginin değiştiğini ve *Ready* yazısının parlak kırmızı ile yazıldığını göreceksiniz.

Ekranı temizleyin:

CLS [ENTER]

Burada olanları daha kolay anlamak için Tablo 2'ye bakın. Bilgisayar ilk açıldığında kullanılan *PAPER* 0 numaradadır. Tablo 2'de birinci sutuna bakarsanız *PEN* numarasını 0 görürsünüz. Şimdi aynı sırada MOD 1 sütununa bakın, renk numarası 1'i göreceksiniz. Tekrar ana renk Tablosu 1'e bakarsanız 1 numarasının mavi renk olduğunu, yani bilgisayar ilk açıldığı zaman görünen *PAPER* rengi olduğunu göreceksiniz.

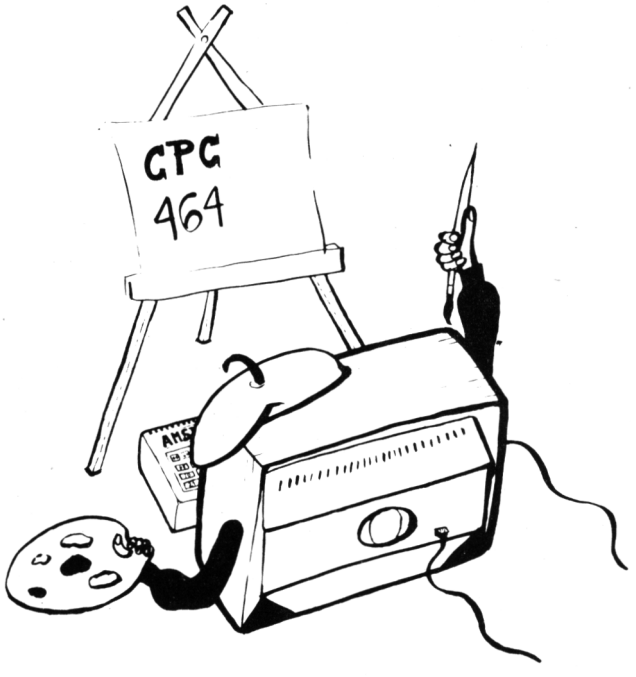
Biraz önce *PAPER* rengini 2 numaraya değiştirdik. Tablo 2'de sol sütunda *PAPER* numara 2'ye bakın. MOD 1 sütununda bu rengin 20 olduğunu göreceksiniz. Tablo 1'e bakın. Şimdi 20 numaranın parlak *Cyan* olduğunu göreceksiniz.

Bilgisayar ilk açıldığında kullanılan *PEN* numarası 1 idi. Tablo 2'ye bakın. MOD 1 sütununda 1 numaralı *PEN* renk numarası 24'dür. Bu renk numarasına Tablo 1'de bakarsanız parlak sarı olduğunu görürsünüz. Bilgisayarı ilk açtığınızda karekterlerin *(PEN)* rengi parlak sarı idi.

	Ink Rengi		
Paper/Pen No.	**Mode 0**	**Mode 1**	**Mode 2**
0	1	1	1
1	24	24	24
2	20	20	1
3	6	6	24
4	26	1	1
5	0	24	24
6	2	20	1
7	8	6	24
8	10	1	1
9	12	24	24
10	14	20	1
11	16	6	24
12	18	1	1
13	22	24	24
14	Yanıp sönme 1,24	20	1
15	Yanıp sönme 16,11	6	24

PAPER/PEN/MODE/INK referans tablosu

TABLO: 2



Biraz önce *PEN* rengini 3 numaraya çevirdik. Tablo 2'de *PEN* numara 3'ün MOD 1'de renk no 6 olduğunu ve tablo 1'de 6 numaranın da parlak kırmızı olduğunu göreceksiniz. Şu anda *PAPER* no 2 ve *PEN* no 3 kullanıyoruz. Bunların renklerini değiştirebiliriz. Bunu *INK* (MÜREKKEP) emriyle yapabiliriz.

INK emrinin iki girdisi vardır. Birincisi değişecek *PEN* ya da *PAPER* numarası ikincisi *PEN* ya da *PAPER*'ın değiştirileceği rengin numarası. Renk numaraları için Tablo 1'e bakın. Örnek olarak, şimdi, ***PAPER 2'nin rengini siyaha PEN*** 3'ün rengini de parlak beyaza çevirelim.

Tablo 1'de siyahın renk numarasının 0, parlak beyazın renk numarasının da 26 olduğunu göreceksiniz.

Şimdi yazın:

Ink 2,0 [ENTER]

(Daha önce de açıklandığı gibi 2 *PAPER*'in şimdiki rengidir ve 0 siyahtır.)

INK 3, 26 [ENTER]

yazalım.

(3 *PEN*'in şimdiki rengi 26'da parlak beyazdır.)

Şimdi *[CTRL] [SHIFT]* ve *[ESC]* tuşlarına basarak bilgisayarı ilk konumuna getirin.

Daha önce de açıklandığı gibi bilgisayar ilk açıldığında ya da *[CTRL] [SHIFT]* ve *[ESC]* tuşları kullanılarak ilk konumuna getirildiğinde *PAPER* numarası 0, *PEN* numarası da 1'dir. *PAPER*'in rengi 1 (mavi) *PEN* rengi ise 24 (parlak sarı). Bunları derhal siyah fon *(PAPER)* üstüne parlak beyaz karakterlere *(PEN)* değiştirmek için şunları yazın:

İnk 0, 0 [ENTER]
ve:
İnk 1, 26 [ENTER]

YANIP SÖNEN RENKLER

Karakterlerin iki ayrı renkte yanıp sönmelerini sağlamak mümkündür. Bunu *PEN*'in *INK* emrine bir fazla renk numarası koyarak yapabiliriz.

Karakterlerin arkasında *PAPER*'ın da yeşil ve parlak sarı arasında yanıp söndüğünü görmek için şunları yazabiliriz.

INK 1, 26, 6 [ENTER]

Bu durumda 1 *PEN* numarası, 26 parlak beyaz ve 6'da parlak kırmızıdır. Karakterlerin arkasındaki fon *(PAPER)* renginin de iki renk arasında yanıp sönmelerini sağlamak mümkündür. Buna da *PAPER*'in *INK* emrine bir fazla renk numarası koyarak yapabiliriz.

Karakterlerin arkasında *PAPER*'ın da yeşil ve parlak sarı arasında yanıp söndüğünü görmek için şunları yazabiliriz.

INK 0, 9, 24 [ENTER]

Bu durumda 0 *PAPER* numarası, 9 yeşil 24 ise parlak sarıdır.

Şimdi *[CTRL] [SHIFT]* ve *[ESC]* tuşlarına basarak bilgisayarı ilk konumuna getirin. (Mod 0'da *PEN*'in 14 ve 15 numaraları ve *PAPER*'in 14 ve 14 numaraları zaten yanan-sönen renkler olduğu için onlara fazladan *INK* emir numarası koymak gerekmez.)

Şimdi deneyelim.

MODE 0 [ENTER]

PEN 15 [ENTER]

Ekranda *Ready* sözcüğünü gök mavisi ve pembe arasında yanıp söndüğünü göreceksiniz.

Şimdi de:

PAPER 14 [ENTER]

CLS [ENTER]

Görüyorsunuz ki bir yandan *Ready* sözcüğü gök mavisi ve pembe yanıp-sönerken fondaki *PAPER*'da sarı ve mavi yanıp sönmekte.

Bu yanan sönen renkleri de *PEN* veya *PAPER* için yeni bir *INK* emriyle değiştirmek mümkündür. *PEN*'in rengini siyah ve parlak beyaza döndürmek için aşağıdakini yazmak gerekir.

INK 15, 0 26 [ENTER]

Bu durumda 15 *PEN* numarası, 0 siyah rengin numarası 26 da -parlak beyaz rengin numarasıdır. *BORDER*'ın de renginin iki renk arasında yanıp-sönmesini sağlamak mümkündür.

UYGULAMA PROGRAMI

Eldeki renk olanaklarının daha geniş uygulamalarını görmek için aşağıdaki programı yazın ve çalıştırın.

Programa bazı sesler de ekledik, bunlar ileride açıklanacak.

```
10 mode 0: ink 0,2: ink 1,24: paper 0 [ENTER]
20 pen 1: for b= 0 to 26: border b [ENTER]
30 locate 3,12: print "BORDER"; B [ENTER]
40 sound 4, (40-b) [ENTER]
50 for t= to 600: next t:next b:cls [ENTER]
60 for p= 0 to 15: paper p:pen 5:print "paper"; p:print [ENTER]
70 for n=0 to 15:pen n:print "pen"; n [ENTER]
80 sound 1, (n*20+p) [ENTER]
90 for t= 1 to 100: next t:next n [ENTER]
100 for t= 1 to 1000 : next t: cls: next p [ENTER]
110 cls:paper 0:pen 1:locate 7,12:print "SON" : for t=1 to 2000: next t
[ENTER]
120 mode 1: border 1: ink 0,1: ink 1,24: paper 0: pen 1 [ENTER]
run [ENTER]
```

GRAFİK

Bundan böyle, her satırdan sonra size *[ENTER]* tuşuna basmanızı hatırlatmıyacağız. Bunu otomatik olarak yaptığınızı varsayıyoruz.

Bilgisayarın hafızasında bir çok karakter sembolleri vardır. Bunlardan birini yazmak için *CHR $ ( )* fonksiyonunu kullanın. Parentezin içine 32'den 255'e kadar uzanan sembol numarasını koymanız gerekir.

[CTRL] [SHIFT] ve *[ESC]* tuşlarına basarak bilgisayarı ilk durumuna getirin.
Şimdi aşağıdakileri yazınız.

PRINT CHR $ (250)

Ekranda 250 kod numaralı karakteri göreceksiniz. (250 numaralı karakter sağa doğru yürüyen bir adamdır.)
Bütün karakterleri ve sembolleri ilgili numarayla birlikte görmek için aşağıdaki programı yazın.

10 FOR n=32 to 255: print n;chr$ (n);
20 next n
run
(Bu satırdan sonra *(ENTER)*'e basmayı unutmayın)
Gerektiğinde başvurmanız için karakterlerin türleri ve bu birim refarans numarası kitabın arkasında *Appendix* 3'de verilmiştir.

LOCATE

Bu emir kürsörün ekranın belirlenen bir tarafına getirmek için kullanılır. ***Locate*** emriyle değiştirilmediği takdirde karakter kürsörü ekranın sol üst köşesinden başlar.. Yani x,y koordinatları 1,1 (x yatay durum, y düşey durumdur)
MOD 1'de 40 sütun ve 20 satır vardır.
Bir karakteri en üst satırın arkasına koymak için aşağıdaki programı yazınız. (Bu satırın sonunda *[ENTER]*tuşuna basmayı unutmayın)
MODE 1 ..... ekran temizlenir, kürsör sol üst köşeye gider.

10 Locate 20, 1
20 Print CHR $ (250)
RUN
Bunun en üst satır olduğunu kanıtlamak için aşağıdakini yazınız.
BORDER 0
Şimdi *BORDER* siyah olduğuna göre adamın en üst sıranın arkasında olduğunu göreceksiniz.
MOD O'da yalnızca 20 sütun vardır. Ama, satır sayısı yine 25'dir. Şimdi bunu deneyelim:
MODE 0
RUN
Adamın üst sağ köşede belirdiğini göreceksiniz. Bunun nedeni MOD O'da 20,0 koordinatının en son sütun olmasıdır.
MOD 2'de 80 sütun ve yine 25 satır vardır. Aynı programı MOD 2'ye uygularsanız adamın nerede görüneceğini herhalde tahmin edersiniz.

Yine de aşağıdakini yazarsanız.
MODE 2
RUN
Şimdi kendiniz ***Locate*** ve *CHR $ ( )* komutundaki numaraları değiştirerek bu komutları deneyin.
Biz bir örnek verelim.

Locate 2Ø 12:print CHR $ (24Ø)
Ekranın ortasında bir ok göreceksiniz.
Hatırlayacağınız gibi bu komutta:
20 × koordinatıydı (1'den 40'a kadar sıralamada)
12 y koordinatıydı (1'den 25'e kadar sıralamada)
240 karakter sembol numarasıydı (32'den 255'e kadar sıralamada)
Ekranda 250 numaralı karakter sembolünü yinelemek için aşağıdaki programı yazın.

```
5 CLS
1Ø FOR ×=1 TO 39
2Ø LOCATE ×,2Ø
4Ø PRİNT CHR$ (25Ø)
5Ø NEXT X
6Ø GOTO 5
RUN
```

[ESC] tuşuna 2 kez basarak programı kesin.
Yeni bir karakter yazmadan önce bir evvelki karakteri silmek için şunları yazın.

```
4Ø print " " ; CHR $ (25Ø)
```

Yeni yazılmış olan 40. satır bir önceki 40. satırın yerini alacaktır.. Eğer şimdi *RUN* komutunu verirsek.
Ekranda karakterin hareketini geliştirmek için değişik karakter ve semboller kullanılabilir.
Program bazı geciktirme çevrimleriyle zenginleştirilebilir.

```
30 call & bd19
RUN
```

Şimdi aşağıdaki programı ekleyin.

```
6Ø for n = 1 to 3ØØ: next n
65 for x = 39 to 1 step-1
7Ø locate x, 2Ø
75 call & bd19
8Ø print chr$ (251); " "
85 next x
9Ø for n = 1 to 3ØØ: next n
95 goto 1Ø
run
```

Bu ilginç programı da bir deneyin.
Daha sonraki bölümlerde açıklanacak bir kaç emirde ekledik. Şimdilik aşağıdakileri yazın.

```
 1Ø mode 1
 2Ø locate 21, 14:print chr $ (244)
 3Ø tag
 4Ø for x = Ø  to 624 step 2
 5Ø move - 16,Ø
 6Ø if x < 308 or x> 34Ø then y= 196 : goto 9Ø
 8Ø y= 536 - ×
 85 sound 1, Ø, 2Ø, 7
 9Ø move ox, oy:print " "; :ox = oy=y
1ØØ mov x,y
11Ø if (x mod 4) = Ø then print chr$ (25Ø); else print chr$ (251);
12Ø for n= to 4 call & bdl9: next n
13Ø next x
14Ø tagoff
15Ø goto 2Ø
run
```

PLOT

Locate emrinin tersine plot pixel koordinatını kullanarak grafik kürsörünün yerini saptamak için kullanılır. *(Pixel)* ekranın son derece küçük bir parçasıdır.

Grafik kürsör ekranda görünmez ve karakter kürsöründen farklıdır.

640 yatay ***pixel***'e 400 dikey ***pixel*** vardır.

x,y koordinatları ekranın x:y koordanitları, 0,0 olan sol alt köşesine göre yeralmıştır.

Locate emrinin tersine bunlar MOD 0.1 yada 2'de değişmezler

Bunu görmek için önce [CTRL] [SHIFT] ve ***[ESC]*** tuşlarına basarak bilgisayarı ilk konuma getirin ve şunları ***[ENTER]***'i unutmadan yazın.

PLOT 32Ø, 2ØØ
Ekranın ortasında küçük bir nokta belirecektir. Şimdi MOD'u değiştirin:
MODE Ø
PLOT 32Ø, 2ØØ
Noktanın hala ortada olduğunu ama, biraz daha büyüdüğünü göreceksiniz.
Şimdi yine MOD değiştirin, MOD 2'nin etkisini görrmek için;
MODE 2
PLOT 32Ø, 2ØØ
Nokta hala ortada ama, çok daha küçük.
PLOT emrine kendinizi iyice alıştırmak için ekranın değişik yerlerine değişik MOD'lar kullanarak noktalar koyun.
Bitirince tekrar MOD 1'e dönün ve
MODE 1: CLS [ENTER]
yazarak ekranı temizleyin.

DRAW (ÇİZMEK)

Önce bilgisayarı ***[CTRL] [SHIFT]*** ve ***[ESC]*** tuşlarına basarak ilk konumuna getirin.
Draw (çizmek) emri grafik kürsörunun son durumundan başlayarak çizgi çizmeye yarar. Bunu daha ayrıntılı görmek için aşağıdaki programı kullanarak ekrana bir dikdörtgen çizin. ***PLOT*** emri kullanarak grafik kürsörü istediğimiz konuma getiriyoruz ve buradan yukarı doğru sol köşeye geliyoruz.. Oradan sağ köşeye VS.
yazın:

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

Programı durdurmak için ***[ESC]*** tuşuna iki kez basın.
Şimdi programa aşağıdaki satırları ekliyerek ilk dikdörtgenin içine bir ikincisini çizin.

```
6Ø plot 2Ø, 2Ø
7Ø draw 2Ø, 38Ø
8Ø draw 62Ø, 38Ø
9Ø draw 62Ø, 2Ø
```

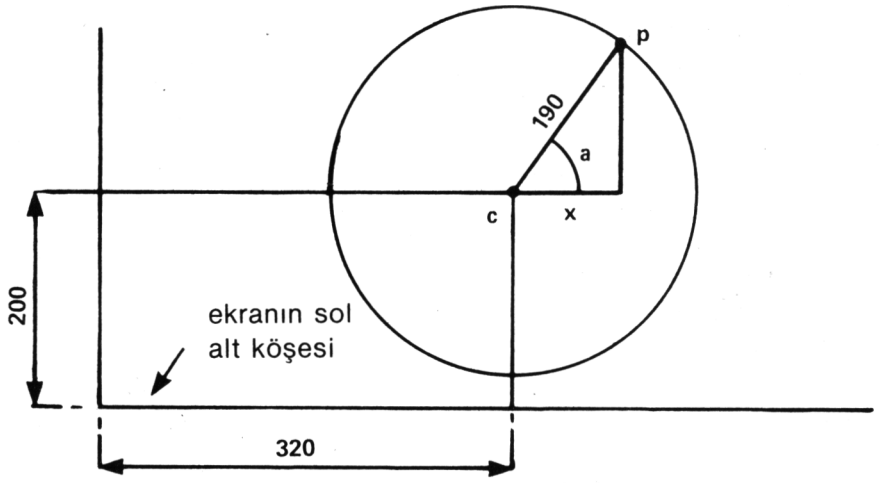
100 draw 20, 20
200 goto 10
run
[ESC] tuşuna iki kez basın.

Çember Çizimi:

Çemberler *PLOT* emri kullanarak yapılabilir. *PLOT*'la çember oluşturma, yöntemlerinden biri bu noktanın x,y koordinatlarını uygun şekilde değiştirerek çember çizmektir.

Aşağıdaki çizime bakın. X,y koordinatları kullanılarak P noktasının koordinatlarının sürekli değişimi ile nasıl bir çember çizildiğini göreceksiniz.
x,y koordinatları şöyledir:

x = 190 * cos (a) ve y = 190 * Sin (a)



NEW

Programı yazmadan önce şimdi de ***new*** komutunu kullanıyoruz. Bu komut bilgisayara *[CTRL] [SHIFT]* ve *[ESC]* tuşlarıyla yaptığımız gibi, hafızayı tamamen temizlemesini bildirir. Ancak,, bir farkla ki, o'da ekranın temizlenmemesidir

Bu, yeni programı yazarken eskisinden yararlanabilmek açısından faydalıdır. (Yeni programı yazarken eskisinden yararlanmak için ekranda kalmasını isterseniz *NEW* tuş sözcüğü faydalı olur.)

Bir çember çizimi:

Bir önceki programda noktalar alt ekranın sol alt köşesine göre konmuştu. Ama, dairenin ekranın ortasında yer almasını istesek, dairenin merkezini 320, 200 koordinatlarına taşımamız gerekir. Sonra da dairenin bu noktasını merkez konuma göre, merkez koordinat konumuna ekliyerek ayarlamamız gerekir. Buna göre daire şöyle çizilir.

```
 5 CLS
10 FOR a=1 to 360
20 DEG
30 PLOT 320, 200
```

4Ø PLOT 32Ø, 19Ø * COS(a), 2ØØ + 19Ø * sin (a)
5Ø Next
RUN

Dairenin yarıçapı 19Ø sayısını azaltarak küçültebilir.
(190 *pixel* dir)
Dairenin daha değişik bir biçimde çizilimini görmek için 15. satırı 15 yazarak programdan çıkartın.
Merkezden çizgilerle çizilmiş * içi dolu daire yapmak için 40. sırada ***plot*** sözcüğünü ***draw*** sözcüğü ile değiştirin.
4Ø. satırın yeni şekli aşağıdaki gibidir.

Draw 32Ø + 19Ø * cos (a), 2ØØ + 19Ø sin (a)

Bunu yine 2Ø. satır çıkmış olarak deneyin.
Dikkat ederseniz bu programın 5Ø. satırına yalnızca ***next*** yazmak yeterli olur. Bilgisayar bunun hangi ***for***'la ilgili olduğunu çıkartabilir. Bir çok ***for*** ve ***next*** çevrimleri varsa, programı incelerken ***next*** ifadesini tanıyabilmek açısından ***next***'den sonra uygun kontrol değişkenini koymak isteyebilirsiniz.

ORIGIN (BAŞLANGIÇ NOKTASI)

Bir önceki programda dairenin merkezini belirtmek için ***plot*** emrini kullandık. Sonra da bu merkez konumuna x, y koordinatlarını ekledik. Bu merkez koordinatlarını ***plot*** komutuyla belirtilmiş noktaya eklemek yerine ***ORIGIN*** emrini kullanabiliriz.. Bu emir dairenin merkezini konumlar. Sonra da x,y koordinatlarını değiştirerek çemberdeki bütün noktaları 1 derecelik farklılık ile çizer.
Bunu görmek için aşağıdaki programı yazınız.

```
new
5 cls
1Ø for a = 1 to 36Ø
15 deg
2Ø origin 32Ø, 2ØØ
3Ø plot 19Ø * cos (a), 19Ø*sin (a
4Ø next
run
```

Daha önce yaptığınız gibi 15 ve 30. satırları değiştirip ***deg*** emrini kaldırarak çemberi içi dolu bir daireye, *DRAW* emriyle çevirebilirsiniz. Ekrana daha küçük dört çember çizmek için aşağıdaki programı yazın.

```
new
5 cls
1Ø for a =1 to 36Ø
15 deg
2Ø origin 196, 282
3Ø plot 5Ø*cos (a), 5Ø*sin (a)
4Ø origin 442, 282
5Ø plot 5Ø*cos (a), 5Ø*sin (a)
6Ø origin 196, 116
7Ø plot 5Ø*cos (a), 5Ø*sin (a)
8Ø origin 442, 116
9Ø plot 5Ø*cos (a), 5Ø*sin (a)
1ØØ next
run
```

Yine hatırlatalım, 15. satırı kaldırın 30, 50, 70 ve 90. satırları değiştirerek *DRAW* emrini kullanabirsiniz.

GOSUB RETURN:

Bir programda bir kaç kere işleme girecek komutlar dizisi varsa, bu komutlar alt program olarak yazılıp *GOSUB* emrinden sonra verilen satır numarasıyla işleme sokulabilirler..

Alt programın sonu *RETURN* komutuyla belirlenir.

Bu aşamada bilgisayar biraz önce uyguladığı *GOSUB* emrinden sonra gelen satıra döner..

Bir önceki programda *PLOT 5ϕ * cos(a), 5ϕ * sin(a)* komutu 4 kez yinelenmiştir.. Bu komutu alt program olarak yazıp her gerektiğinde *GOSUB* sözcüğü kullanılarak işleme sokulabilir.
Aşağıdakilerini yazın:

```
new
5 cls
1ϕ for a= 1 to 36ϕ
15 deg
2ϕ origin 196, 282
3ϕ gosub 12ϕ
4ϕ origin 442, 282
5ϕ gosub 12ϕ
6ϕ origin 196, 116
7ϕ gosub 12ϕ
8ϕ origin 442, 116
9ϕ gosub 12ϕ
1ϕϕ next
11ϕ end
12ϕ plot 5ϕ*cos(a), 5ϕ*sin(a)
13ϕ return
run
```

Dikkat ederseniz 110. satırda *end* komutu kullanılmıştır: Böyle yapılmasaydı program doğal olarak 100. komuttan sonra 120'yi uygulamaya geçerdi. Oysa bu satır ancak *GOSUB*'la çağırıldığı zaman gereklidir.

Bu bölümü bitirmeden önce aşağıdaki programı deneyelim. Bu program artık anlıyor olmanız gereken bir çok programlama emirleri içeriyor.

```
new
1ϕ mode ϕ:border 6:paper ϕ:ink ϕ,ϕ
2ϕ gosub 16ϕ:for x=1 to 19:locate x,3
3ϕ pen 15:print " ";chr$ (238)
4ϕ for t=1 to 5ϕ:next t:sound 2, (x+1ϕϕ)
5ϕ next x:gosub 16ϕ:for b=3 to 22
6ϕ locate 2ϕ, b:pen 7:print chr$ (252)
7ϕ cls: gosub 16ϕ: next b
8ϕ sound 2,ϕ, 1ϕϕ, 15, ϕ, ϕ, 1
9ϕ gosub 16ϕ:border 16,24: locate 2ϕ,25
1ϕϕ pen 14:print chr$ (253);
11ϕ for t = 1 to 1ϕϕϕ: next t
```

```
120 border 6:gosub 160:for f=3 to 24
130 locate 10, (25-f): pen 2
140 print chr$ (144): cls:gosub 160
150 sound 7, (100-f), 5 next f:goto 10
160 locate 10,25: pen 12
170 print chr$ (239): return
run
```

SOUND (SES)

Ses, bilgisayarın içindeki oparlörden çıkar.

Eğer MP 1 modülatör güç kaynağı ve bir televizyon kullanıyorsanız, televizyonun ses kontrol düğmesini en aza getiriniz.

Ses seviyesi bilgisayarın sağ tarafındaki *VOLUME* kontrolunden ayarlanabilir. Ses ayrıca stereo setinizin ***auxilary input*** girişine bilgisayarın arka panelindeki (1/0) kullanılarak verilebilir. Böylece bilgisayardan çıkan sesleri ***HI-FI*** oparlörler veya kulaklıklarıyla dinleyebilirsiniz.

SOUND (ses) emrinin yedi girdisi vardır. Bunlardan ilk ikisinin kullanılması şarttır. Ötekiler isteğe bağlıdır.. Emir şöyle yazılır..

SOUND kanal durumu, ton periodu, süre, ses düzeyi, ses düzeyi envelopu, ton envelopu, gürültü periodu.

Aşağıdaki örnekte kanal durumu için 1 (yani kanal durumunun referans numarasını) yazacağız.

TONE PERİOD

Ek 7'ye bakarsanız orta do'nun ton periodu 478 dir.
Örnek:

```
NEW
10 sound 1,478
RUN
```

0.2 saniye süren kısa bir nota duyacaksınız.

.....

SÜRE

Sesin süresi belirtilmezse 0.2 saniye sürer. Sesin süre ünitesi 0.01 saniyedir. Sesin 1 saniye sürmesini sağlamak için 100, 2 saniye sürmesi için 200 kullanılmalıdır..
Örnek:

```
10 sound 1,478, 200
RUN
```

do.. notasını 2 saniye duyacaksınız.

SES DÜZEYİ

Bu sayı notanın başlangıcındaki ses düzeyini belirler. Sayılar 0'dan 7'ye kadar sıralanmıştır. Ama, ses düzeyi ***envelopu*** belirtilirse bu sıralama 0'dan 15 çıkar. Hiç numara kullanılmazsa 4 varsayılır. Örnek:

```
10 sound 1,478, 200,3
RUN
```

Bu sesin düzeyine dikkat edin ve daha değişik ses düzeyi kullanarak yeniden yazın:

10 sound, 1,478, 200,7
RUN
Bu sesi şimdi çok daha yüksek duyacaksınız.

SES DÜZEYİ ENVELOPU

Ses düzeyi ***envelopu*** emri ***env.***'dir. Bunun normalda 4 girdisi vardır. Sondan 3 girdi 5'e kadar var olan ***envelope*** bölümlerinde görülür. Biz burada bunlardan yalnız birini kullanıyoruz. Daha geniş açıklama 6. bölümde vardır..

Env. numarası, basamak sayısı, basamak genliği, basamak zamanı.

ENVELOPE NUMARASI

Belirli bir ***envelop***'un ses emrinde belirlenebilmesi için ona ***envelope*** numarası verilir. ***Env.*** numaralarının sıralanması 0'dan 15'e kadardır.

BASAMAK SAYISI

Basamak sayısı, basamak zamanına bağlantılı olarak kullanılır. örneğin 1'er saniyelik 10 basamak elde etmek isteyebilirsiniz. Bu durumda basamak sayısı 10'dur. Basamak sayılarının sıralanması 0'dan 127'ye kadardır.

BASAMAK GENLİĞİ

Her basamak ses genliği bakımından esas basamağa göre 0 seviyesinden 15'e kadar değişebilir. 15 ses seviyesi ***sound*** emrindeki seviyelerle aynıdır. Ancak basamak --128+127'ye kadar ayarlanabilir. Böylece yalnızca ses gücünü belli şekilde aşağı yukarı ayarlamakta kalmayıp aynı zamanda 15 yukarı numaraları kullanarak çok değişik sesler elde edebilirsiniz. Basamak ses gücü numaraları -128'den+127'yekadar değişebilen bir sıralamadır.

BASAMAK ZAMANI

Bu numaralar 0.01 saniyede (saniyenin 1/100'ü) basamaklar arasında geçen zamanı belirler. Basamak numaraları 0'dan 255'e kadar değişen bir sıralamadır. Böylece basamaklar arasındaki en uzun zaman süresi 2.55 saniyedir.

Bu ses düzeyi ***envelopuyla*** deneme yapmak için aşağıdaki programı yazın.

```
5 env 1,10, 1,100
10 sound 1,28, 1000, 1,1.
RUN
```

10. satır, 10 saniye sürer, başlangıç ses düzeyi 1 olan ve 5. satırda gösterilmiş olan 1 numaralı envelop numarası kullanan, 284 ton periodlu sesdir. (la sesi) satır 5, her bir basamak ses düzeyinin her saniyede 1 attırarak yükselen, basamaklardan oluşmuştur. (100 x 0.01 saniye)

Satır 5'i aşağıda gösterilen şekillerde değiştirip çalıştırın. Her seferinde değişen envelopun etkisini dinleyin.

```
5 env 1,100, 1,10
5 env 1,100, 2,10
5 env 1,100, 4,10
5 env 1,50, 20,20
5 env 1,50, 2,20
5 env 1,50, 15,30
```

Birde bunu deneyin ve dinleyin.

5 env 1, 5Ø, 2, 1Ø

Sesin yarısında seviyenin sabit kaldığını fark etmişsinizdir. Bunun nedeni basamak sayısının 50 olması ve her basamak arasındaki zamanın 0.1 saniye olmasıydı. Böylece ses genliğinin değiştiği süre yalnızca 5 saniye idi. Oysa 10. satırdaki sound komutunda sesin süresi 10 idi. (1000 sayısı). Kendi kendinize deneme yapıp ne tür sesler yaratabileceğinizi deneyin.

TON ENVELOPU

Ton envelopu komutu ***ent***'dir. Normalde 4 girdisi vardır. Bunlardan son 3'ü mevcut envelop kısımlarının herhangi birinde görülür. Burada bunlardan yalnızca birini kullanacağız. 6. bölümde daha fazla açıklama bulacaksınız. ***Ent:*** envelope numarası, basamak sayısı, basamak ton periodu, basamak zamanı.

ENVELOP NUMARASI

Envelop numarası belli bir envelopu ***sound*** komutunda belirlemek için kullanılır. Envelop numaraları 1'den 18'e kadar bir sıralamayla yükselir.

BASAMAK SAYISI

Bu sayı basamak zamanına bağlantılı olarak kullanılır. Örneğin her biri bir saniye olan 10 basamak elde etmek istiyebilirsiniz. Basamak sayıları 0'dan 239'a kadar yükselen bir sıralamadır.

BASAMAKLARIN TON PERİODLARI

Her basamağın ton periodu - 128'den + 127 arasında değişir. Negatif - basamaklar notaların frekanslarını arttırır. (sesleri tizleştirir) En kısa ton periodu 0'dır. Ton envelopunu hesaplarken bunu akılda tutmalıdır. EK 7'de basamakların Periodlarının - 128'den + 127'ye kadar sıralanmaları gösterilmiştir.

BASAMAK ZAMANI

Bu sayı 0.01 saniyede (saniyenin 1/100'ünde) basamaklar arasında geçen zamanı belirler. Basamak zamanı 0'dan 255'e kadar yükselen bir sıralamadır. Böylece basamaklar arasında en uzun süre 2.55 saniyedir. Ton envelop denemesi yapmak için aşağıdaki programı yazın.

```
5 ent 1,1ØØ, 2,2
10 sound 1,284, 2ØØ, 15,Ø,1
RUN
```

10. satır 2 saniye sürer. Başlangıç ses düzeyi 15 (en çok) olan, ses düzeyi envelop'u olmayan (0 ile gösteriliyor) ve ton envelop numarası 1 olan, 284 ton Periodlu bir sesi belirler. (la)

5. satır, 100 basamaktan oluşan ve her 0.02 saniyede ton periodunu iki katına çıkartan (frekansı düşürerek) ton envelop numara 1'dir.

Şimdi ton envelopunun değişmesinin etkisini duymak için 5. satırı aşağıdaki biçimlerde değiştirin ve programı çalıştırın.

```
5 ent 1,1ØØ, -2,2
5 ent 1,1Ø, 4, 2Ø
5 ent 1.1Ø, -4, 2Ø
```

Şimdi de ***sound*** komutunu ve ton envelopunu aşağıdakilere göre değiştirin.

5 ent 1,2,17,7Ø
10 sound 1,142,14Ø,15,Ø,1
15 go to 5
RUN

Programı kesmek için *(ESC)* tuşuna 2 kez basın.

Şimdi de yeni sesler yaratmak için ses düzeyi envelopu, ton envelopu ve sound komutunu birlikte koyabilirsiniz.

NEW
5 env 1,1ØØ, 1,3
1Ø ent 1,1ØØ, 5,3
2Ø Sound 1,284, 3ØØ, 1, 1, 1
RUN

10. satırın yerine aşağıdakini koyun.

1Ø ent 1, 1ØØ, -2,3

Şimdi de bütün satırları aşağıdakilere göre değiştirin:

Yine değiştirin:

5 env 1,1ØØ, 2,2
1Ø ent 1,1ØØ, -2,2
20 sound 1,284, 2ØØ, 1,1,1
RUN

Artık sizde daha değişik denemelere girişebilirsiniz.

NOİSE (Gürültü)

Sound komutunun sonuna gürültü eklenebilir. 1'den 31'e kadar değişen gürültü sıralaması vardır. Envelop komutunu kullanmayı sürdürecek gürültü numarasını eklemeyi deneyelim:

6. ve 20. satırların yerine aşağıdakileri koyarak yazın:

5 env 1, 1ØØ, 3,1
20 sound 1, 2ØØ, 1ØØ,1,1,1,5
RUN

Noise kullanarak ve kullanmıyarak ses düzeyi ***envelop*** ve ***sound*** komutunu değiştirerek alışılmamış sesler bulmaya çalışın.

1. BAŞLARKEN

Kitabın giriş bölümüne yeni başlayanlar için koyduğumuz bilgisayarı bağlamak, açmak klavyeyi tanımak gibi ayrıntılarını atlamış olanlar, burada kullanılan terimleri anlamakta güçlük çekerlerse başa dönüp giriş bölümündeki temel kursu okumalıdırlar.

Bu bölümün kapsadığı konular:

Bu el kitabında kullanılan uygulamalar, bilgisayarı çalıştırmak, klavye alışkanlığı.

Bilgisayarlara ve bilgisayar programlamasına ne kadar alışkın olursanız olun, lütfen bu bölümde verilen bilgisayarın kurulması ile ilgili bölümleri okumaya özen gösterin. Kutunuzu yeni açtıysanız ve bir an önce başlamak için sabırsızlanıyorsanız bu bölümde merakınızı tatmin edecek ve hemen *BASIC* uygulamasına geçmenizi sağlayacak bilgiler vardır.

Bu bölüm bilgisayar kullanmakta deneyimi olanlar içindir. Yeni başlayanlar giriş bölümünü okumalıdırlar.

ÖNEMLİ - Aşağıdakini okumanız GEREKLİDİR

Terimler:

Metinde tuşlara ve klavyeye yapılan referansları ve program listelerini oluşturan metni açıklığa kavuşturmak için, bu el kitabında kullanılacak uygulamalar şunlardır:

[ENTER]: Ekranda yazılı karakter olarak karşılığı olmıyan, yani ekranda yazılı olarak görünmeyen tuşlar [ ] bu kare parantez işaretlerin arasında gösterilmiştir.

QWERTYUIP: Ekranda yazılı karakter olarak karşılığı olan tuşlar kare parentez içinde yazılmamıştır.

10 FOR N = 1 to 1000 : Ekranda görünen ya da klavyede yazılan metinler bu biçimde yazılırlar.

Büyük harf O ile **Ø** sayısının farklı yazıldığına dikkat edin.

Her programı veya direkt emiri ***[ENTER]*** tuşuna basarak bitireceğiniz varsayılıyor.

Bilgisayar küçük harf girilen bütün ***BASİC*** komutları programı ***LIST*** edilince BÜYÜK HARFE çevirir.. Bundan böyle gösterilen örnekler ***LIST*** edilince ***BASIC*** komutlar BÜYÜK HARF çıkacağı için bizde büyük harflerle göstereceğiz. Ancak siz küçük harf kullanırsanız hatalarınızı daha kolay ve çabuk saptayabilirsiniz. Çünkü yanlış yazılan basic komutlar ***LIST*** edilince küçük harf olarak kalır.

1.1.1. KUTUYU AÇIYORUZ

RENKLİ MONİTÖR

ÖNEMLİ

Lütfen bu el kitabında aletinizin ***main Plug*** ile ***main Lead*** bağlantılarını anlatan ayrıntılı KURULUŞ önerilerini dikkatle okuyun.

Bilgisayarı resim 1'de gösterdiği gibi bağladıktan sonra, önce monitörü, sonra da bilgisayarı sağ taraftaki kayan düğmeden açınız. 30 saniye kadar ısınma süresinden sonra monitörde aşağıdakileri göreceksiniz.

```
Amstrad 64K Microcomputer  (v1)
©1984 Amstrad Consumer Electronics plc
        and Locomotive Software Ltd.

BASIC 1.0

Ready
```

Bu durum, bilgisayar açıldığında ve klavyede gereken tuşlara, gerektiği sırayla basarak meydana gelir.

[CTRL] [SHIFT] ve *[ESC]* tuşlarına aynı anda basıp kısa bir süre parmaklar kaldırılmaz. Bilgisayara başka bir şey girmeden önce bunu bir deneyin.

Monitörün sağındaki parlaklık *(BRIGHTNESS)* kontrol düğmesinden parlaklığı uyarlayın. Renk fabrikada önceden ayarlanmıştır. (mavi fon üstüne altın sarısı karakterler). Bu renk birleşimini değiştirmek isterseniz program komutları vermeniz gerekir. Bu konuda 8. bölümde geniş bilgi vardır. Ama şimdi hemen bölüm atlamazsanız daha iyi olur. Biz aşağıda metin yazmak için en uygun ve okunaklı renk birleşimini en çok karakter sığmasına olanak sağlıyan MOD 2'de verdik.

```
10 MODE 2
20 INK 1,0
30 INK 0,13
40 BORDER 13
```

Yukarıdaki satırları direkt komut olarak da girebilirsiniz.

Eğer yukarıdaki program size bir şey ifade etmiyorsa ya da bir türlü "doğru" olarak yazamıyorsanız, hemen bu el kitabının baş tarafındaki "Yeni başlayanlar" için Temel kursa dönmeniz gerekir: 80 Sütunluk metin görüntüsü program yazılımı açısından en kullanışlı olanıdır. Yukarıda verdiğimiz programı her seferinde yeniden yazmamak için boş kasete alıp saklamak isterseniz bunu 2. bölümü okuduktan sonra yapabilirsiniz.

DİKKAT
Renkli monitörün fazla parlak durumu gözlerinizi yorabilir. Çok yakın oturmayın ve parlaklığı normalin üstüne çıkartmayın. Gözlerinizin yorulmaya başladığını hissederseniz aletinizi kapatıp, bir süre başka işlemle meşgul olun. Ama bunu baştan önlemek için aşağıdakilere dikkat edin:

1- Çalıştığınız yerde her zaman bu kitabı rahatlıkla okuyabilecek kadar ek ışık olmalıdır.. Kitabı ekrandan gelen ışıkla okumanızı kesinlikle salık vermeyiz.
2- Yaptığınızı rahatlıkla görebilmek için gereken en az parlaklık derecesini kullanın.
3- Ekrandan mümkün oldukça uzak oturun.

Monitörle birlikte bir masa lambası kullanmanız göz yorulmasını azaltır. Lambanın ekranda parlama yapmaması için monitörün arka tarafında olması gerekir.

1.1.2. YEŞİL MONİTÖR

GT. 64 monitörünün ön yüzünün alt tarafında 3 kontrol düğmesi vardır. Bunlar 3 ayrı ayar içindir. ***BRIGHTNESS*** (parlaklık) ***CONTRAST*** (kontrast , yani görüntünün en parlak kısımlarıyla en sönük kısımları arasındaki frak) ve ***Vertical HOLD*** (dikey durum) Bu görüntünün yukarıdan aşağı kaymasını engeleyip sabit tutmak içindir. ***Vertical HOLD'***u sık sık ayarlamak gerekmez ve bir kez ayarladıktan sonra varlığı unutulabilir. ***BRIGHTNESS*** ve ***CONTRAST,*** CPC 464'u kullandığınız odanın ışık şartlarına göre zaman zaman ayarlanabilir.

Tek renkli monitörde (görüntüde değişen elemanların kontrastını sağlamak için karakterlerin yoğunluğu değişir) başlangıç mesajı renkli monitörünkiyle aynıdır. Ancak metin açık yeşil fon üstüne daha koyu ve parlak karakterlerle yazılır.

GT 64 monitör ile aralıksız çok uzun süreler çalışmak gözleri yorabilirse de bu monitördeki görüntü uzun süreler çalışmaya elverişlidir. Görüntü özellikle 80 kolonluk modda program yazılımı için çok elverişlidir. Parlaklığı, ***(BRIGHTNESS)*** karakterlerin çizgilerini oluşturan noktalar netliğini fazla kaybetmeden yeterince aydınlık bir ekran elde edecek şekilde ayarlayın.

80 kolon MODU için aşağıdaki kısa programı CPC 464'e yazın. Görüntü için seçenekler elde edeceksiniz.

```
10 REM GÖRÜNTÜ
20 FOR n=0 TO 26
30 MODE 2
40 INK 1,n.
50 INK 0,(26-n)
55 BORDER n.
60 LOCATE 15,12: PRINT "herhangi bir tuşa basın"
70 a$= INKEY$
80 IF a$="" " GOTO 70
90 NEXT.
100 GOTO 20
```

Yukarıdaki örnek bu kitaptaki program listelerinin yazılımı hakkında önemli bir noktayı daha ortaya koyuyor.. Bazı programlardaki satırlar listesini verdiğimiz satırları kestikten sonra koyduğumuz boşlukların aslında program için gerekli olmadığını, bunu yalnızca program listesinin düzeni açısından yaptığımızı bilmeniz önemlidir.

Bu örnek gri tonlarının karışımlarını bütün olanakları ile göstermiyor ama eldeki olanaklar hakkında bir fikir veriyor. Hoşunuza giden bir karışım bulunca ***[ESC]*** tuşuna iki kez basarak programı durdurun. "***Break***" mesajını göreceksiniz.

Buradan itibaren bu kitapda renkli monitöre özel bir çok program yapacağız. İlginç renk ve grafik etkiler gösteren programlar yeşil monitöre, pek görünmeyecektir. (5. bölümde ayrıntılı açıklanmıştır.)

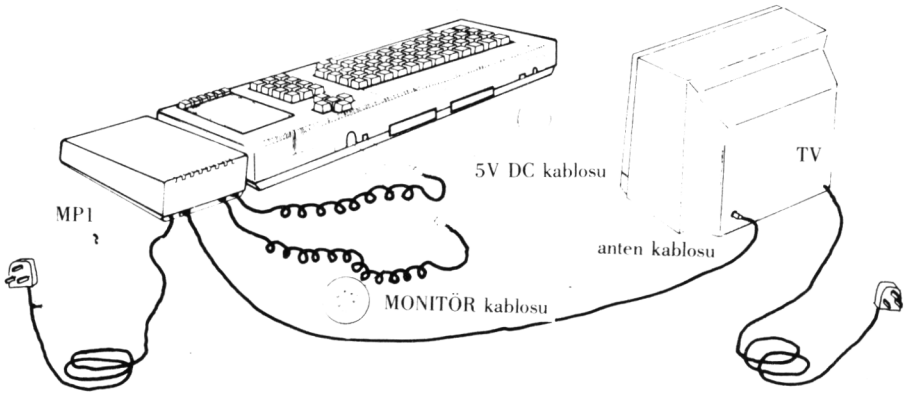
Yeşil monitörün avantajı, programı yazarken daha kesin ve daha az yorucu bir görüntü vermesidir. Ama bilgisayar hastalığına yakalanmışsanız, bir de renkli monitör almaktan kendinizi kurtaramazsınız.

1.1.3. MP1. V Modulatör güç kaynağı.

Şu anda CPC 464 bilgisayar ile GT 64 yeşil monitör kullanıyorsanız MP1 ek olarak almak isteyeceğiniz bir parça olabilir. MP1 bilgisayarınızı evdeki renkli televizyonlarınızla birlikte kullanıp CPC 464 nu bütünlemek olanaklarından yararlanabilmenizi sağlar.

ÖNEMLİ

Bu kitabın baş tarafındaki bilgisayarın kurulması bölümündeki aletinizin kablo bağlantılarını ayrıntılı bir şekilde anlatan önerileri dikkatle okuyun.



Resim 2. MP.1, bilgisayar ve televizyonunuzun anten giriş bağlantısı.

Modülatör güç kaynağı bilgisayarın sağında Televizyon arkasına yakın bir masaya konulmalıdır. (yukarıda resim 2 de gösterildiği gibi)

Şimdi televizyonunuzun ses ayarı düğmesini en aza getirin. CPC 464'ün iç oparlörü vardır. Bu yüzden Bilgisayar açıkken ses yüksek ise TV'den uğultu gelmesi normaldir. TV'yi açın sonra da bilgisayarı sağ taraftaki *POWER* yazılı düğmeden açın.

Klavyenin üst kısmında ortadaki kırmızı on ışığın yanması gerekir ve şimdi de TV alıcısına bilgisayardan sinyal alacak şekilde ayarlamanız gerekir.. Televizyonda yedek ya da kullanılmayan bir kanal bulun ve televizyonunuzun markasına göre açılış görüntüsünü elde edecek şekilde ayarlayın.

```
Amstrad 64K Microcomputer  (v1)
©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.

BASIC 1.0

Ready
```

Bu görüntüyü net ve temiz görecek şekilde TV alıcısını ayarlayın. Yazılar yoğun bir mavi fon üstüne altınsarısı renkte olmalıdır. Ama bu TV'nızın markasına göre değişebilir.

Sinyaller önce modüle sonra demodüle olma sürecinden geçtiği için video sinyalinde belli bir miktarda bozulma olacaktır. Sonuçlar monitör kullanıldığında olduğu kadar iyi olamaz ve hatta televizyon alıcınızın kalitesine göre 80 sütun metin MOD'u (mode 2) çok net bir sonuç vermeyebilir. Bu durumda olanaklar elverdikçe MOD 1'i kullanmalısınız.

1.2. İlk adımlar.

Şimdi her şeyiniz hazır. Güç kaynağı bağlanmış, Video çıkışı monitöre takılı, ve cihazınızı açtınız. Bilgisayarınız giriş bekliyor. Bilgisayarınızı çalıştırın, gördüğünüz mesaj klavyede bir şey yazmadan görebileceğiniz tek metindir. ***BASIC*** programlama diliyle daha önce işlem yaptıysanız herhalde kısa bir "tanışma" programı yazmışsınızdır bile. Bir çok bakımdan ***AMSTRAD BASIC***'e alışmak kolay olacaktır. Bir an önce başlamanız için size bilgisayarın içine yerleştirilmiş karekterlerin hepsini sergileyen kısa bir program göstereceğiz. Buna karakter seti diyoruz. Bu tabir klavyeden yazarak ekranda görüntüleyebileceğiniz sayı, figür ve başka elemanların sıralanmasıdır. Göreceğiniz bazı karakterler doğrudan tuşlara basarak elde edilemezler. Bunlar ancak ***PRINT CHRS $*** (girdi) kullanılarak görüntülenebilirler. Bunun nedeni bilgisayarda saklanmış bir elemanın ***'byte'*** olarak bilinen bir veri unitesiyle depolanmasıdır. EK 2'yi incelerken göreceğiniz gibi bir byte'ın 256 değişik değer alma olanağı vardır.Bilgisayar bu sakladığı karakter için en az bir bütün byte kullandığına göre çoğu daktiloda bulunan 96 kadar karakterle yetinip diğer 160 olanağı ziyan edeceğimize 256 olanağın tümünü kullanmakta tabiki yarar vardır.

Karekterlerin standart sıralanması ***"subset"*** olarak bilinir. Bilgisayar dünyasında bu ***'ASC II'***kodlama sistemi olarak sınıflandırılır. Bu terim: ***American Standard Code for Information Interchance***'ın baş harflerinden oluşmuştur. Esas amacı bir bilgisayara gönderilen bir mesajın anlaşılır biçimde olmasını sağlamaktır. Bilgisayar dünyasının tek gerçek anlamda evrensel yönüdür.. Bu yüzden ASCII'nin bu yönünü iyi tanımanızı kuvvetle salık veriyoruz, Ek 3 ASCII'nin kod sıralaması da CPC 464 de bulunan ek karekterleri ve sayı kodları da gösterilmiştir. Tek tuşla elde edilemiyen karakterlerin bazıları da ***KONTROL*** tuşu (klavyede ***CTRL*** yazılı tuş) ile birlikte kullanılarak görüntülenebilir. Kontrol tuşunun fonksiyonunu anlayıncaya kadar yapacağınız rastgele denemelerden hayırdan çok zarar gelir.

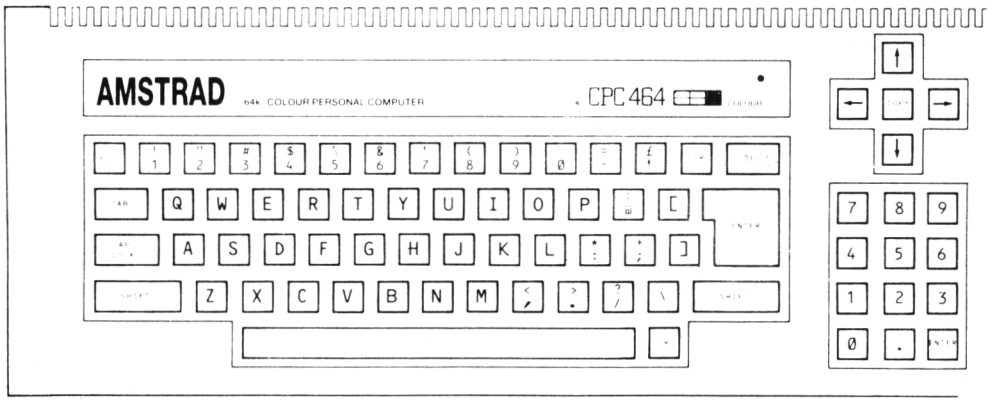
1.2.2.

Bu karakterlerin tam olarak neye benzediklerini görmek için aşağıdaki programı yazın. Hem merakınız gider hem de CPC 464'ün bir takım marifetlerini görmüş olursunuz. Bu programı yalnızca program yapma kolaylığı açısından güven vermekle kalmayacak aynı zamanda aleti açtığınızda açılış mesajını aldığınız sürece donanım bakımından problem ile karşılaşmıyacağınızı ve CPC 464'ün doğru bilgiyle programlamaya hazır olduğunu belirterek size iç rahatlığı sağlayacaktır.

(Bu programı yazarken hata yaparsanız 1.2.7'ye atlıyan ve bütün programı en baştan yazmadan hatalarınızı düzeltmeye çalışın)

Arka sayfadaki programı yazarken küçük harf ***(a b c)***ya da ***[SHIFT]***'le büyük harf (ABC) kullanabilirsiniz. Her iki şekilde bilgisayar programı düzenler. Ama mutlaka kelimelerin bittiğini ara ya da başka işaretlerle gereken yerlerde belirtmelisiniz. (virgül, iki nokta üstüste vs. ne gerekiyorsa). Bu önemlidir, çünkü ***AAMSTRAD BASIC*** değişken isimler içinde bazı özel sözcüklerin kulianması yasaktır.

Resim 4'de gösterilen klavyeden sözde klavye olarak söz edilir.Çünkü tuşlardan çoğu yeniden tanımlanabilir. Bunlar ilerki bölümlerde daha açıklığa kavuşacaktır.



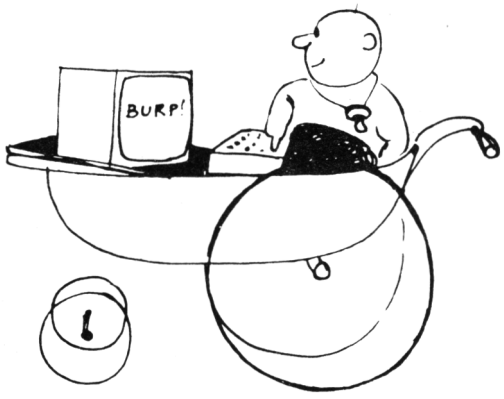
Resim 4-CPC 464'ün klavyesi.

[ENTER] tuşuna basmak yazdığımız emir ya da program satırını bilgisayara girmek ya da satır bir numarayla başlıyorsa programı başka bir yerinde kullanmak üzere saklama etkisi yapar.

[ENTER]'den "***carriage return***" olarak söz edilir. Bu bilgisayarın ilk dönemlerinde kullanılan terimlerin mekanik daktilo terimlerinden alınmasından kalmıştır. ASCII karakter setinde ***[ENTER]***'ın kodu CR olduğuna göre artık hep kalacak demektir.

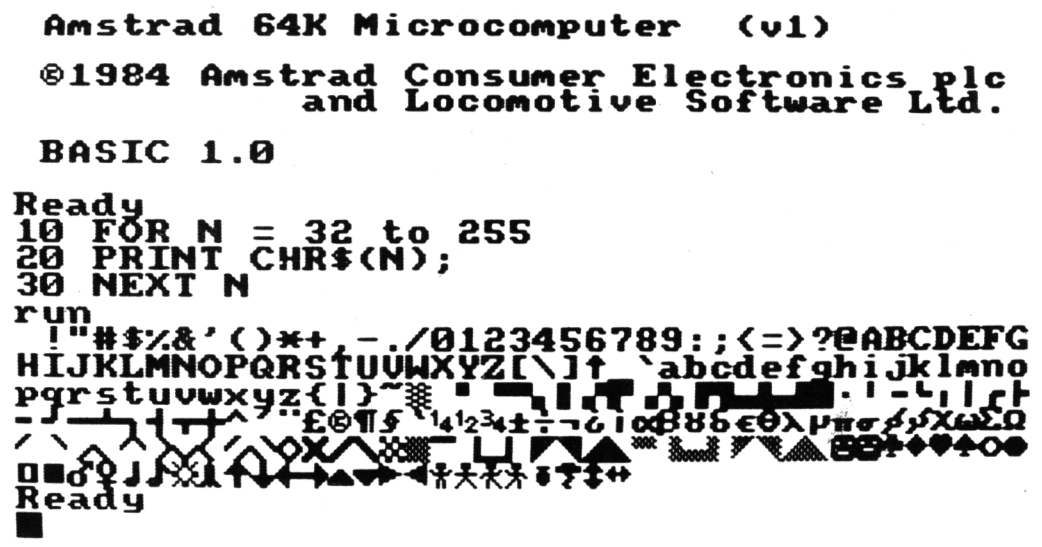
Örnek

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



1.2.3.

Bakın ekranda ne oldu:



Bilgisayara şimdi yazdığınız kısa programı tüm karakter setini sergilemesi söylendi. Eğer yukarıdaki görüntü çıkmadıysa demekki programı yazarken fark etmediğiniz bir yanlış yaptınız. 1.2.7'ye atlayarak problemi çözmeye çalışın.

Her şeyin yolunda gittiğini ve gereken sonucu aldığınızı varsayarak görüntünün arkasındaki satırları inceleyelim. Bu sizin CPC464'ün karakter sıralamasıyla bir iletişimi olduğunu da anlamanıza yarayacak.

Farketmeniz gereken ilk nokta bilgisayara *PRINT "a b c d e f g h j k l m vs"* komutu değil de *PRINT CHR $ (N)* komutunun verilmesidir.

N yalnızca değişkeni belirten kullanışlı bir harfdir. Seçim harfiydi -bu tür uygulamalarda bu harf matematikçilerin gözdesidir. Değişkenler bilgisayar programlama tekniğinin bir parçasıdır ve programda verilen komutlara göre değişir. 5 gibi bir sayı sabittir 4 ile 6 arasındaki yer alır-değişken değildir. *N* de sabittir alfabede de bir harftir.

Peki bilgisayar bu ayrımı nasıl yapabilirdi? *N* Harfi alfabetik bir karakter olarak söylenseydi, bunu tırnak işareti arasında yazardık.

"N"

Bilgisayar buna ***syntax error*** mesajı verirdi. Çünkü *FOR "N"* diye bir emir tanımıyor. N'i yalnız kullanarak bilgisayara onun bir değişken olduğunu söylemiş olduk. *BASIC* de *FOR* bildirisinin tanımı ondan sonra bir değişken gelmesini gerektirir. Dolayısıyla *FOR*'un arkasından ne gelirse onu değişken kabul eder.

Bilgisayara *N= 32 to 255* olduğunu da söyledik. Yani değişkenin sıralanmasını da açıkla-

dık. Gerçekten de 32 den başlayıp 255 de biten bir seridir bu. Değişkeni de açıkladıktan sonra bilgisayara onunla ne yapması gerektiğini de söylemeliyiz. Onu da şöyle yaptık:

2ø PRINT CHR $ (N):

Bu komut bilgisayara N'ye verilen sayı değerini sayı karşılığındaki karaktere değiştirmesini söyler.. ***CHR $ (N)***'in ***BASIC*** komutundaki fonksiyonu budur.*N* karşılığındaki karakterleri bulup ekrana yazar.

Satır sonundaki noktalı virgül bilgisayara satır atlatmayı önlemesini söyler. (sonraki karakteri tekrar sol sütuna koyarak yeni bir satıra başlama) yoksa bunu otomatik olarak yapardı ve karakterler birbiri ardından sıra halinde durmayıp ekranın en solunda aşağı doğru görünürlerdi.

Sonraki satır bilgisayara seri de (32) ye yapması gerekeni yaptıktan sonra ***FOR*** bulunan satıra dönüp aynı şeyi şimdi de***NEXT*** sayesinde *N*'den sonra gelen değişkene yapmasını söylüyor. Bu çevrim olarak bilinir ve bilgisayar programcılığının en temel ve hayati yönlerinden biridir.

Bütün bilgisayar dillerinde çevirimler şu ya da bu şekilde mutlaka bulunur. Siz de bunu programlarınızda kullanmaya çabuk alışacaksınız. ***FOR*** çevrimindeki kontrol değişkeni son değerine gelince (255) işlem durur ve bilgisayar 30.satırdan sonraki satıra bakar. Olmadığına göre durup ***ready*** yazarak emir bekler. ***RUN***'a basarak tekrar çalıştırabilirsiniz. Bunlar artık bilgisayarın hafızasında siz silinmesini söyleyinceye kadar ya da elektrik kesilinceye kadar saklanacaktır. Elektrik kesildiğinde bilgisayardaki tüm veri (programlar, değişkenler vs.) kasete alınıp saklanmamışsa kaybolur.

Bu program bilgisayar konusunda bir temel nokta ortaya koyuyor: tüm işlemlerini sayılara bağlantılı olarak yapar. Bilgisayar alfabeyi ve koca bir karekter serisini, gereken karekter için kod numarasını kullanarak görüntüleri *A* tuşuna bastığınız zaman bilgisayara ekrana *A* yazmasını değil, hafızasında *A* harfini görüntülemesini sağlıyan sayısal bilgi bulunan kısma bakmasını söylüyorsunuz.

Bu verinin yeri, klavyede *A* tuşuna basarak harekete geçirilen bir sayıdır. Her karekteri karşılayan bir numara vardır: bunların listesi bu kitabın EK 3 bölümünde vardır.

1.2.4.

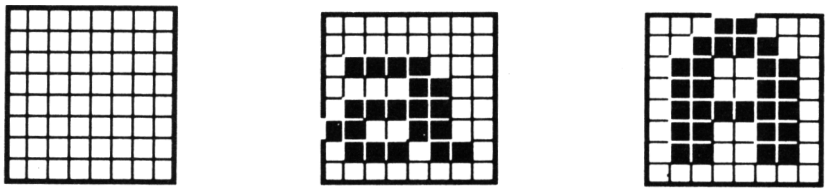
Bu sayfada da anlatılan teknik noktaları ya da kullanılan dili tam olarak anlamazsanız hiç canınızı sıkmayın. Bilgisayarın sizin komutlarınızı nasıl bir işlemden geçirdikten sonra gereken cevabı verdiğinizi tam olarak açıklamak önemli, ama büyük bir olasılıkla bunları yalnızca kafası tekniğe çok yakın olanlar tam olarak anlıyacaktır. Bu kısmı çok ağır bulursanız 1.2.5'e atlıyabilirsiniz.

Örneğin a harfinin kodu 97 dir ama, bilgisayar 97 yi de anlamaz. Bu sayı ondalık sistemden bilgisayarın ilişki kurabileceği bir koda değiştirilmesi gerekir. Buna genellikle Makine kodu denir. Makinenin bu yönünü altında yatan prensipler (ilkeler) EK 2'de ele alınmıştır. İlk bakışta, günlük hayatta alıştığımız bir ondalık sayı notasyonunun bilgisayarın 16 lık sistemdeki notasyonuna çevirmek işi yokuşa sürmek gibi görünecektir. 10 ünitesine dayalı sayıları düşünmek o kadar doğal olmuştur ki başka türlüsü kulağını ters göstermek gibi bir şey oluyor.

Hex notasyonu (16'lık sistem için bunu kullanacağız) anlamak için belli bir zihinsel ustalığa varmak gerek ama bir kez bunu başardıktan sonra bilgisayar konusunda bir sürü şey

kafanızda yerli yerine oturacak ve makina kodunun ince yapısı açık görünecektir.

Bilgisayar *A* tuşuna basılmasını kendi anladığı tip numaraya çevirdikten sonra, hafızasında belirtilen bölüme bakar ve bunun sonucu karekteri tanımlayan bir başka sayı serisidir. Yani ekranda gördüğünüz karekter hafızada sayısal matris olarak depolanmış bir blok veri olarak oluşmuştur.



Resim 5. Boş karekter, küçük harf ve büyük harf A

Matrıs'ın elemanları, noktadan oluşmuş sıralar ve sütunlardır. Karekter gerekenlerin boş olmasıyla görüntülenir. Her noktayı bilgisayarın hafızasında olan veriler saptar. CPC 464 görüntü tablosunda her karekter hücresinde 8 sıra ve 8 sütun vardır. (bu bir sıra bir byte tanımlar) 255'lik karekter setinde istediğiniz karekteri bulamazsanız, 8 bölümde anlatılan *SYMBOL* ve *SYMBOL AFTER* tuş sözcüklerini kulanabilirsiniz.

Kullanıcı tarafından tanımlanan karekterler, rastgele bir düzende ayarlanmış 0-64 noktadan oluşmuştur. Böylece bu matrisin bütün olanaklarını kullanan tam bir karekter seti bir sürü daha değişik karekter içerir. Buna bir de eleman bloklarını birleştirerek daha büyük blok karekterler oluşturabileceğinizi eklerseniz kullanıcının tanımlayacağı karekterlerin sayısı sizin zamanınız ve yaratıcılığınızla sınırlanmış olur.

1.2.5 Programlamaya dönüş:

Yazdığınız program biraz düzensiz görünüyor. Ekranda açılıştaki mesajının kalıntıları var. Programı çalıştırmadan önce bunları silip ekranı temizlesek daha düzenli bir görüntü olur. Bunu ayarlamak için bir satır daha ekliyelim.

Aşağıdaki satırı kürsörün durduğu yere yazın. (Kürsör ***Ready*** mesajının altındaki içi dolu karedir. Eğer bunu bilmiyorsanız dönüp temel kursu okuyun.)

5 CLS
RUN

Bakın ekran karekter setini yazmadan önce nasıl tertemiz oldu.

Bu da size *BASIC* programlama dilinin en anlayışlı yönünü gösteriyor yani program numaralarını hangi sırada verdiğinize bile aldırmaz. Hatta bir kere program hafızaya girdikten sonra bunu eklemek için görüntülemenize bile gerek yok.

Bilgisayar programı uygulamadan önce numara sıralarını titizlikle düzenler. *LIST* komutunu vererek kontrol edin.

1.2.6 LIST komutu.

Bilgisayarın program hafızasına ne sakladığını *LIST* komutuyla kolaylıkla kontrol edebilirsiniz.

LIST yazalım.

Ekrandaki sonuç şudur.

```
5CLS
1Ø FOR N = 32 TO 255
2Ø PRINT CHR S (N) ;
3Ø NEXT N
Ready
```

Bu program siz aşağıdakilerden birini yapana kadar kalır.

* Aleti kapatmak.
* *[CTRL] [SHIFT]* ve *[ESC]* tuşlarıyla ilk konumuna getirmek *(RESET)*
* Kaset üniteden *LOAD* veya *RUN*'la programı yüklemek.
* *NEW* yazarak *[ENTER]*'e basmak. Bu komut renk ve görüntü MOD'unu ilk konumuna getirmeden bütün değişkenleri ilk konumuna getirir ve program hafızasını temizler.

Şimdi fonksiyon tuşlarından birini *[ENTER] CLS: LIST [ENTER]* yazma görevini üstlenmek üzere ayarlayın (Bu program girişini ve gelişmesini son derece hızlandıran bir özelliktir). Bunu yapmak için şunları yazın:

```
KEY 138, CHR $ (13) + "CLS : LIST" + CHR $ (13)
```

Şimdi (sağdaki) sayısal tuş tablosunda nokta tuşuna basın.

Bu şekilde ön programlama yapabileceğiniz 32 tuş vardır. (sayısal) tuş tablosunu esas amacında kullanmak isterseniz klavyedeki tuşlardan herhangi birini yeniden tanımlamak için kullanabilirsiniz. 8. bölümde *KEY* komutuna (emrine) bakınız.

Programınız uzunsa tuşu *(KEY)* şöyle tanımlayın:

```
KEY 138, CHR $ (13) + "CLS:LIST"
```

Bundan sonra tuş size gereken satır numaralarını girmenize izin verir ya da ona üst üste iki kez basarak tam *listing* yapar.

Renk denemeleri yaptığınızda fon ve yazılar aynı değere ayarlandığı için görüntüde okunması olanaksız olabilen renk karışımları arasında yolunuzu şaşırabilirsiniz. Şöyle bir ayarlama yaparsanız:

```
KEY 139, CHR $ (13) + "Mode 2: ink 1, Ø : ink Ø, 9" + CHR $ (13)
```

...sadece sayı tablosundaki küçük *[ENTER]* tuşuna basarak, görünebilen esas renk konumuna (mavi-altın/sarı) dönebilirsiniz. (hafızadaki programı yitirmezsiniz)

Kullanıcının tanımladığı *(KEY)* tuş kodları aletle beraber bunlarda *(reset)* ilk konumuna geldiklerinden, hoşunuza gidenleri bulduktan sonra kolayca elde edebilmek için teype program olarak yazıp saklayın.

1.2.7 Düzeltmeye Giriş

Programı yazarken yanlış yapmanız kaçınılmazdır. 1.2.2'den atlayanlar bu kısmı dikkatle okusunlar.

CPC 464, hem yaptığınız yanlışları düzeltme işini mümkün olduğunca basit tutmaya hem de değiştirmek istemediğiniz karekterleri yazma probleminden kaçınmaya çalışır.

Kürsörün hareketini kontrol tuş grubu bilgisayarın dikkatini ekranda değiştirmek istediğiniz kısma yönlendirir.

Örneğin aşağıdaki on numaralı satırı girerken:

1Ø FOR N = 332 TO 255

bir kaç seçeneğiniz var:

1- *(ENTER)* tuşuna basarak bütün satırı yeniden yazmak bu durumla yanlış satır hafızadan silinir ve aynı satır numaralı yeni satır onun yerini alır.

2- (←) tuşuna basarak kürsörün yanlış girişin üstüne getirip silebilirsiniz.

1Ø FOR N = 3 3 2 TO 255

Dikkat ederseniz normalde kürsör renginde olan karekter kürsörün altında fon renginde ve kolayca görünebiliyor.

Yanlış karekteri silmek için *[CLR]* tuşuna basın. Yanlış karekter ortadan kalktığı gibi sıra boşluğu kapamak için yakınlaşır:

1Ø FOR N = 32 TO 255

Tekrar *(ENTER)*'e basın ve bilgisayarın belleğindeki satır düzeltilmiş satır olacaktır. Kürsörün ille de satırın sonunda olması gerekmez. Bilgisayar yeni satırı, kürsörün yerine aldırmadan girer.

3- Bir de kürsörü silmek istediğiniz karekterin sağına getirebilirsiniz.

1Ø FOR N = 33 2 TO 255

Şimdi *[DEL]* tuşuna basın. Kürsörün solundaki karekter silinir.

[ENTER]'a basın daha önce olduğu gibi satır hafızaya kalkar.

1Ø FOR N = 32 TO 255

1.2.8. Sonradan düşünülenler:

Şimdiye kadar yazdığınız düzeltme metodları satır sonuna gelip *[ENTER]* tuşuna basmadan önce kullanılabilecek yöntemler. Ancak yanlışların çoğu dalgınlıkla yapıldığından, ancak programı çalıştırdığınızda ortaya çıkar ve bilgisayar yanlış (error) mesajı verir. (EK 8) Bu takım yanlışları bilgisayar birinci sütunun başına düzeltme kürsörü koyup hatalı satırı yazarak görüntüye getirir. Bu durumda yukarda gösterilen işlemleri, yanlışları önceden fark etmiş gibi, kullanabilirsiniz.

Ama hatalı satır size yanlışınızı göstermek üzere ekrana gelmediyse programın listesini alarak ekrana getirip yanlışı bulmanız ve sonra düzeltmeniz gerekecektir.

1.2.9 COPY kürsörüyle düzeltme:

Önce *LIST* kullanarak programı görün (tek seferde) ekranda görülebilen kısa bir deney programıyla çalıştığınızı varsayıyoruz .

5 CLS
1Ø FOR N = 32 TO 255

2Ø PRINT CHRS (N) ;
3Ø NEXT N

Hata 20. satırdadır. $ yerine S yazılmış($ bilgisayara kendinden hemen sonra gelen karekteri, sayısal veri olarak değil de karekter olarak almasını bildirir) ya hemen 20. satırı yeniden yazıp yeniden girersiniz ya da ekranı aşağıda gösterildiği gibi kullanırsınız.

[SHIFT] tuşunu basık tutarak (hangisi olsa olur) (↑) yukarı kürsör tuşuna basın.

İster satırları tek tek çıkartın, ister parmağınızı kaldırmadan istediğiniz satıra kadar gelmesini bekleyin. Parmağınızı üstünden çeker çekmez kürsör durur. Bunu bir kaç kere deneyin alışkanlık edinmenizde yarar var. Bir sıra üste çıkarsanız bu kez (↓) aşağı sürsör tuşunu kullanın *[SHIFT]*'e basarak tabi.

Bu işlem *copy* kürsörünün esas kürsörden ayrılmasına neden olur. (ikisinin görünümü aynıdır) ve *copy* kürsörü değiştirmek istediğiniz satıra getirilir. *Copy* kürsörü satır ilk karekterin üstünde olarak başlayın.

```
5 CLS
10 FOR N = 32 TO 255
20 PRINT CHRS(N);
30 NEXT N
Ready
■
```

(SHIFT) tuşunu basılı tutmadan esas kürsörü düzeltilecek satıra getirseydiniz bilgisayar bunu geçerli bir hareket saymazdı. Çünkü yalnızca ana kürsörden hemen sonra girilen karekterler geçerli komut olarak kabul edilir. Satırı bu şekilde fazladan yazarsanız *[ENTER]* tuşuna ya da *CHR $* (13) içeren fonksiyon tuşuna basmadan önce *[ESC]* tuşuna basarak bu karmaşıklıktan kolaylıkla kurtulabilirsiniz.

Yanlışlıkla *NEW* sözcüğünü yazar ve emri *(ENTER)* emriyle girerseniz programınız tamamen kaybolur onun için dikkatli olun.

Bir kez bir satıra yazmaya başladıktan sonra *(ENTER)* tuşuna basmadan ondan kurtulmak isterseniz, bilgisayar *"bliip"* gibi bir ses çıkartır (yanlış sınırlara girdiğinizde hep bu sesi duyacaksınız) Ama bu problemden *(ENTER)*'a basarak kurtulursanız programdan hiç bir şey kaybolmaz. Ancak ilk yazılan karekterler kadar geçerli bir satır numarası yazdıysanız satır fazladan yazılmış olur ve yeniden yazılması gerekir.

Copy kürsörü doğru yerine gelince, değiştirilmesi gereken karekterin üstüne gelinceye kadar *[COPY]* tuşuna basın.(*Copy* kürsörünün hızına alıştıktan sonra bunu parmağınızı kaldırmadan yapabilirsiniz).

5 CLS
1Ø FOR N = 32 TO 255
2 Ø PRİNT CHR$ (N) ;
3Ø NEXT N
Ready

Şimdi *copy* kürsörünü bırakın ve $ yazın. $ esas kürsörün altında çıkacak kürsörde herzamanki gibi bir adım sağa gidecektir.

2Ø PRINT CHR$

Yanlış yazılan S harfinin üstünden *Copy* kürsörünü geçirmeniz gerekir.

Bunu yapmak için *[SHIFT]* tuşunu basılı tutarak sağ kürsör tuşuna (→) bir kez basmalısınız. Bundan sonra *Copy* kürsörü (üstünde durur. *[SHIFT]* tuşunu bırakıp *[COPY]* tuşunu satırın sonunu aşıncaya kadar bastırın *[ENTER]* tuşuna dokunun.

Ekranın alt tarafındaki düzeltilmiş satır liste de görülen hatalı satırın yerini alır.

Bütün bu teknikleri, hatalı satırı *[COPY]* ederek ve sonunda *[ENTER]* tuşuna basmadan esas kürsörün düzeltme özelliklerini kürsör tuşları ve *[CLR]* ve *[DEL]* tuşlarını *[SHIFT]* tuşuna basmadan kullanarak düzeltebilirsiniz. *[CTRL]* tuşunu bastırarak kullanılan sol (←) ya da sağ (→) kürsör tuşu kürsörü düzeltilecek satırı ya da sağına tek hareketle götürür.

Bunu biraz deneyin, kısa sürede kolay gelecektir.

Nihayet şöyle yazarak da düzeltebilirsiniz.

EDIT 20

Bilgisayarın cevabı şudur.

20 PRINT CHR$ (N) ;

Daha önce anlatılan şekilde esas kürsörü *[CLR]* ve *[DEL]* tuşlarıyla birlikte kulların ve sonuçtan memnun olunca da *[ENTER]* tuşuna basın. Bir karışıklık olursa *[ESC]* tuşuna basıp yeniden liste isteyin *[ESC]* tuşunun kaldığı satırın üstüne yazılmış olmaz.

Şimdi yine *LIST* girin, programın düzeltilmiş halini göreceksiniz, olmamışsa tekrar deneyin.

Şimdiye kadar CPC 464'ün yalnızca temel çalışma düzeyinde araştırmalar yapmaya başladık. Diğer 2 MOD'u görmek için şunları yazın:

MODE 0
RUN

Ekranın önce temizlendiğini sonra da enine 20 karekter sergilediğine dikkat edin.

Tekrar ilk görüntüye dönmek için şunları yazın:

MODE 1
RUN

İşte başladığımız yerdeyiz. 80 sütunluk görüntü için yazın:

MODE 2
RUN

Artık işe girişmiş sayılırız. Başlangıçtaki merakınızın bir kısmını yenmiş olmanız gerekir. Daha ileri kullanıcılar en gözde programlarını *BASIC*'in bu türüne çevirmeye ve çalıştırmaya başlamışlardır bile. Daha az alışkın kullanıcılar bu birinci bölüme *BASIC*'in makina özelinde kullanılan genel bir bakış edinebilmek için, devam etsinler.

2 KASET TEYP İŞLEMLERİ

Bu bölümün içerdiği konular:

* Veri kayıt kasetleriyle ses kasetleri arasında benzerlik ve ayrılıklar.
* Teypten program yükleme ve çalıştırma-Tanıtım bandı.
* İsteğe bağlı banda kayıt hızı ayarı.
* Kasede program kaydetme.
* Yükleme hataları.

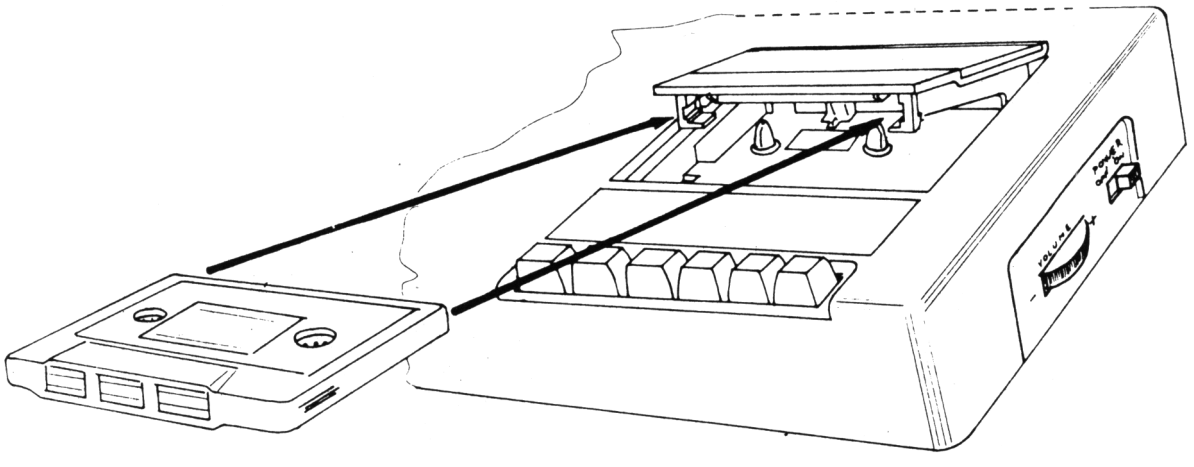
CPC 464'ün hafızası sadece gerilim varken ve bilgisayar açıkken veri depolayabilir. Elektrik kesilmesine karşı program ya da veri depolamak isterseniz bunları kasete veya disk sistemine kaydetmeniz gerekir.

2.1 Kaset kontrolu:

Klavyenin sağ tarafında kaset teybi bulacaksınız.

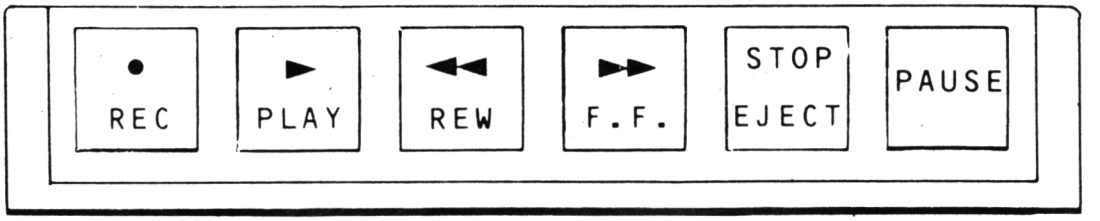
kasedi teybe yerleştirme

Bu kaset mekanızması esasda ses kaset mekanızmasıyla aynıdır. Ancak teybi kontrol eden sinyaller özellikle veri depolama sistemiyle çalışabilmesi için optimuma çıkartılmıştır.



Resim 2.1-Veri kaydedicisine kaset yerleştirmenin doğru yolu.

Veri kaydedicisinin tuşları çoğu ses kayıt cihazlarınkiyle aynıdır.



Resim 2.2. CPC 464 veri kayıt kontrolu

Bu tuşlara daktilo tipi klavyeli tuşlarından daha fazla güçle bastırılması gereğine dikkat edin.

REC = (RECORD) Kayıt (Program komut verince) veri kaydı için ***[PLAY]*** ile birlikte çalışır. Kayıt konumu bir kaset (resim 2.2) konup kapak kapanmadıkça çalışmaz. Çalıştırmak için ***[PLAY]*** tuşuna basmalısınız. Bu durum sağlandıktan sonra bilgisayar ya programdan aldığı komutla ya da doğrudan klavyeden gelecek ***SAVE*** emriyle verileri kaydeder.

(PLAY) = Play tuşu kasete programı yükleme ya da kaset sistemden program çalıştırmak için kullanılır.

Bilgisayar ya hafızadaki en son programdan komut aldığında, ya da klavyeden direkt emirle kasetteki verileri alır. Kasetteki band bitince mekanik bir hareketle ***[REC]*** ve ***[PLAY]*** tuşları ilk konumlarına getirilirler.

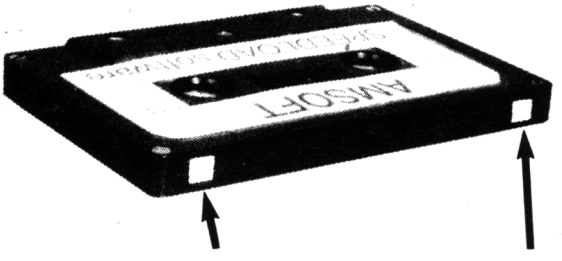
[REW] geri sarma tuşu
Sağ makaradan sol makaraya bandı sarar. Bu tuşun teyp bitince otomatik durması yoktur. Bu yüzden dikkatli olup fazla uzun süre bırakarak motorun ısınmasına neden olunmamalıdır.

[F.F] (ileri hızlı sarma tuşu) bandı sol makaradan sağ makaraya hızla sarar. Bu fonksiyonun da band sonunda otomatik durması yoktur. Yine dikkatli olup motorun lüzumsuz çalışma sonucu ısınmasına meydan bırakılmamalıdır.

[STOP-EJECT] Kaset hareketini durdurup bütün tuşları ilk konumuna getirir. Bu tuşa ikinci kez basıldığında kaset kapağı açılır ve kaset konulur ya da çıkartılır. Hareket durdurulmadan kaset çıkartılmaz.

(PAUSE)-(PLAY) ya da ***(REC)*** ve ***(PLAY)***'e bağlı olarak mekanik bir duraklama işlemidir. Ancak (data) veri okunması ya da yazılması işlemleri sırasında kullanılmamalıdır. Aksi taktirde hataya yol açar.

2.2 KORUMALI KAYIT



KAYIT YAPILMASINI ENGELLEYEN TIRNAKLAR

Kasette saklamak istediğiniz verilerin yanlışlıkla silinmesini önlemek için mekanik iç kilit sistemi bütün standart kasetlerde sağlanmıştır. Arka kenarın iki yanındaki küçük parçalar kopartılarak bu bandın üstüne yeniden kayıt yapılması önlenir. Ve bu durumdaki bir kaset teypteyken, ***[REC]*** tuşu basmaz. Kasetin bu iki yüzündeki verilerin korunabilmesi için küçük parçaların ikisinin kopartılması gerekir. İlerde bu kaseti yeniden kullanmak isterseniz arkada kopardığınız parçaları kapatacak bir şekilde seloteyp benzeri bir yapışkan kâğıt koyarak kullanabilirsiniz.

2.3 KASETTEN VERİ YÜKLEME

Resim 2.1 teybe kasedi nasıl yerleştireceğinizi gösteriyor. Kaset tam başa alınmış olmalıdır. Tam sarılmamışsa ***(REW)*** tuşuna basarak başa alın. Eğer band kasetin önünden kazayla çıkmışsa, kasetin içine sardıktan sonra yerleştirin. Böyle yapmazsanız hem kaseti, hem doldurmak istediğiniz bilgiyi yitirebilirsiniz. Unutmayın ki müzik kasetlerinde bandta ufak tefek eğilmeler bükülmeler fazla önemli değildir, ve gene de müzik verir. Ama veri kayıt kaseti bu tür zedelemelere dayanamaz. Zedeli kaset güvenilir olamaz.

Örneğin kazayla bandın bir kısmı kasetten çıkmış, bir bandın üstüne kapak kapatarak ya da bir başka şekilde zedelediğiniz halde kayıt yaptığınızı görürseniz, hemen verileri daha bilgisayar hafızasındayken yeni ve zedelenmemiş bir kasete çekin ve zedeli kaseti bir daha yanlışlıkla kullanmamak için çöpe atın.

2.4 TANITIM BANDININ KULLANIMI

CPC 464 ile birlikte verilen kasette, bilgisayarın ve onun içine yerleştirilmiş programını ve ***AMSTRAD BASİC MOS***'un yapabildiği grafik ve ses marifetlerinden bazıları örnekleniyor.

MOS— Cihaz işletme sisteminin fonksiyonlarıdır ve teyp ile gerekli işlemleri de kapsar. Bu teypdeki programları ve verileri yazma ve okumak için bir seri kısa komut içerir. En sık rastlanan ***RUN"*** (çalıştırma) ve ***LOAD*** (yükleme) komutlarıdır.

İşletmeyi mümkün olduğu kadar basitleştirmek için CPC 464, klavyeyi kullanmakta yeterince deneyim sahibi oluncaya kadar kullanacağınız ve klavyeden girmeyi basitleştiren bir çok özel fonksiyon içerir.

Bilgisayarlarınızı açmış altında ***Ready*** sözcüğüyle açılış mesajına bakıyorsanız, kasedi Resim 2.1 de gösterildiği gibi yerine yerleştirin. Teyp numaratörünü, kaset penceresinin yanındaki küçük düğmeye basarak 000 konumuna getirin ve ***[CTRL]*** tuşuna basarak parmağınızı kaldırmadan yalnızca sayı tuşları bulunan sayısal tuş tablosundaki küçük ***[ENTER]*** tuşuna basın. Şu mesajı göreceksiniz.

RUN"

Press *PLAY* then any key:

Bu 1. Bölümde kısaca tanıtılan ***(KEY)*** tuş emirlerinin biraz genişlemiş bir örneğidir. Yani basit ve sık kullanılması gereken komutlar serisini hızlandırmak için bilgisayar bütün bir emirler serisini kısa yoldan klavyeden tuş girişleriyle yerine getirmesi söyleniyor. ***RUN"*** komutunu yazıp ***[ENTER]***'e de basabilirdiniz. Ama, aynı sonucu iki tuşa basarak almak daha kolay.

Çoğu yerde iki ***[ENTER]*** tuşunun görevi aynıdır. Ama bazı durumlarda küçük ***[ENTER]*** tuşu programdan gelen komutlar kullanılarak başka bir işleme göre tanımlanır. Kullanıcının tanımladığı tuşlar konusu ilerde daha ayrıntılı işlenecektir.

Basmanız söylenen *(PLAY)* tuşu kaset işletme mekanizmasındadır. Bu tuşa kendiliğinden basılı kalması için sıkıca basın. Mesaj da, herhangi bir tuşa basın sözü bilgisayar dilinde sık kullanılan bir deyimdir. Yeni başlayanları yanlış yapmaya götürebilir. Bir programa, klavyeden daha fazla komut beklemeden programda bir sonraki işleme geçmesini belirten seri işlemleri basitleştirmek için kullanılır.

Aslında 'herhangi bir tuşa basın' yerine: *(SHIFT)*, *(CAPS LOCK)*, *(CTRL)*, *(ESC)* ya da kaset yan panelinde bulunan tuşlar dışında herhangi bir tuşa basın, denmesi daha doğru olurdu. Ama bütün bilgisayarların işleri basitleştirmek için belli bir bilgi birikimini varsaymaları kaçınılmazdır.

Bu kitapta ne zaman 'her hangi bir tuşa basın' sözü görürseniz, biraz önce belirttiğimiz tuşların dışında bir tuş olduğunu anlarsınız. Bastığınız tuş ekranda görünmez, yalnızca teyp motorunu harekete geçirir. Başka bir tuşa basın *([ENTER]* tuşuna basmaya çalışmak iyi olur) *(PAUSE)* tuşunun yanlışlıkla basılı olmamasına dikkat edin.

Birden fazla tuşa basarsanız bilgisayar buna aldırmaz. Programı yüklemeye başladıktan sonra basacağınız tuşları basılmamış sayar.

Dikkat ederseniz özel bir program ismi belirtmediniz.

RUN"

Komutundan sonra aynı sırada başka bir şey yer almazsa bilgisayar teypde bulabileceği ilk programı arar ve onu yüklemeye başlar. Bilgisayar teypde ilk düzgün kaydedilmiş programı bulduğunda aşağıdaki mesaj görünür.

Loading WELCOME 1 block 1

Bu size, bilgisayarın *"welcome 1"* (hoş geldin) başlıklı bir seri program blokundan ilkini bulduğunu belirtir. Her program teypde veri blokları şeklinde saklanır. (2 k byte boyuna kadar) Her veri bloku teypde ayrı ayrı tanımlanır ve ekrandaki mesaj size şu anda okunan blok hakkında bilgi veriyor. Her blok veriden sonra teyp bir an duraklar ve tekrar başlar. Tekrar başladıktan az sonra da halen okunan blok numarasını içeren mesaj görünür.

Her hangi bir zamanda bilgisayar kötü bir veri yakalarsa, ne tür bir problemle karşılaştığını belirten bir hata mesajı gösterir. (listesi EK 8'de) Genellikle yalnışsız yükleme yaptırıncaya kadar programı çalıştırmaktan başka, yapabileceğiniz birşey yoktur.

Bandınızı yüklediğini varsayalım. Ekrandaki komutları uygulayın *"Welcome"* bandı gerisini yapar.

2.5 Güvenli yükleme ve Hızlı yükleme:

CPC 464 kullanıcıya iki hız sunar: güvenli 1000 baud (saniyede 1000 bit) ve Hızlı yükleme. 2000 baud görüldüğü gibi Hızlı yükleme, güvenliden bir kat daha hızlı veri okuyabiliyor. Ancak, bazen ucuz teyplerde görünen tutarsızlıkları ve değişik kayıt cihazlarının teyp başlarından kaynaklanan hataları önlemek için gereken güven sınırından da fedakârlık eder.

Aynı cihazda kaydedilen ve yüklenen ve makul ölçüde kaliteli teyp kulanılan programlarda Hızlı yükleme güvenilebilir. Hızlı yükleme çoğu program bandında yalnışsız doğrudan yüklemede kullanılabilir. Ama, *AMSTRAD* bütün programların bu iki hız gücüyle doldurulmasını salık verir.

Bilgisayar otomatik olarak kendini, kayıt yapıldığı hızla okumaya ayarlar. Bir programı kayıt yapacağınız zaman bilgisayara Hızlı yükleme komutu vermeniz gerekir. Yoksa güvenli hız seviyesinde çalışır.

Program ve verilerinizi yüksek hızda saklamak için direk *MOD*'da olduğunuzu kontrol edin. (*Ready* mesajiyla görüntüdeyken)

SPEED WRITE 1

Tekrar daha yavaş hıza dönmek için ya bilgisayarı ilk konumuna getirirsiniz ya da şöyle yazabilirsiniz:

SPEED WRİTE Ø

2.6.

Programı ve verileri kasede kaydetmek, BASIC'ın kasete verileri yazdırma şekliyle ilgili bir kaç emri vardır. Burada bunlar örneklerle özetlenmiştir.

2.6.1 SAVE "Program ismi":

Programı çalıştırdıktan veya listesini aldıktan sonra gördüğünüz ***Ready*** sözcüğünden sonra kasede kaydetmenin en dolambaçsız yolu *SAVE* emrini kullanmaktır. 1. bölümde ele aldığımız görüntülenebilen karekterleri ***list*** edebilen kısa programı örnek olarak kullanalım.

Program ismi klavyenin boşluk dahil 16 karekterinin her hangi bir kombinasyonudur. 17.ci veya daha sonraki karekterler atılırlar.

Bilgisayarın hafızasında program verirken şunları yazın:

SAVE "KAREKTERLER"

Bilgisayarın cevap mesajı

Press Rec and play then any key

REC'e ve *PLAY*'e sonra da herhangi bir tuşa basın.
"Herhangi bir tuş" terimini hatırlayın, teyp başlıyacak ve bilgisayar programı KAREKTERLER ismi altında saklıyacaktır.

ÖNEMLİ
Bilgisayar doğru tuşlara basıp basmadığınızı kontrol edemez. Yani yalnızca ***PLAY*** tuşuna basarsanız teyp çalışmaya başlar, program alınıyormuş gibi görünür ama alınmaz.

DİKKAT-
Bir programı yüklerken yanlışlıkla ***[REC]*** ve ***[PLAY]***'e basarsanız kasette olan programı silersiniz. ***[ESC]*** tuşuna basarak durdurulmazsa kasette, program varsa silersiniz. Böyle yanlışlıklar yapmak istemiyorsanız arkası istenmeden silinmelere karşı kopartılmış kaset kullanın.
CPC 464 4 metodla kayıt yapar. İlk anlattığımız genel metoddu. Ama daha özel durumlar için 3 yol daha vardır.

2.6.2. SAVE "Program ismi", A

Yapılan işlemler yukarıda anlattığımızla aynıdır. Yalnız bir *A* eki vardır. Bu bilgisayara programı ve verileri her zaman yaptığı gibi kısaltılmış şekilde değil de ASCII text file şeklinde saklamasını sağlar.

Bu kaydetme metodu ***wordprocessor***'lar ve başka uygulama programları tarafından kullanılır. Bunun kullanılışı, uygulamalara geldiğimizde daha fazla açıklanacaktır.

2.6.3 SAVE "Program ismi", P

Buradaki, *P* kasetteki verileri, programı bir başkasının çalıştırarak kolayca okuyamıyacağı *[ESC]* tuşuna basıp durduramıyacağı şekilde kaydetmesini söylüyor. Bu şekilde saklanan programlar yalnız *RUN* veya *CHAİN* emirleriyle kullanılabilirler. Programı düzeltmek ya da değiştirmek ihtimaliniz varsa bir de normal bir kopyesini saklayın.

2.6.4 SAVE "Program ismi", B, (başlama adresi), (uzunluk), (çalışma adresi)

Bu seçenek ikili sistemde kaydetme olanağı sağlıyor. Bilgisayarın *RAM*'ında depolanan bütün bir veri bloku tam hafızada, olduğu şekilde kasete depolanıyor. Bu durumda program *BASIC* programı gibi çalıştırılacaksa bilgisayara hafızanın saklamak istediğiniz yerinin nerede başladığını, ne uzunlukta olduğunu ve uygulamaya başlatacak hafıza adresini vermelisiniz.

İkili sistemde kaydetme özelliği ekran kaydetmesi şeklinde ekrandan doğru kasede veri kaydetme olanağı sağlıyor. Esas kullanımlarından biri, uzun programlarda uzun yükleme işleminin tekdüzeliğini bölmek için "başlık" parçaları oluşturmaktır.

2.6.5 İsimsiz programlar ve CAT

Program ismi vermeden program kaydediyorsanız

SAVE " "

BASIC bunu isimsiz program olarak saklar. Disk sisteminde her girişin tek bir ismi olması gerekirse de kaset aynı isimden (isimsizler dahil) teype sığdırabilecek kadar çok programı ard arda saklayabilir.

Programlarınıza içeriklerini hatırlatacak birer isim vermezseniz kısa zamanda hesabını şaşırırsınız. Aynı zamanda bir de tarih koduyla program ve veri kayıtlarınızın sırasını da belirtmenizi salık veririz.

Kasetlerinizin içeriğini *CAT* emriyle katologlıyabilirsiniz. *CAT* emrini verin ve talimatı yerine getirin.

Press PLAY Then any key

PLAY ve daha sonra herhangi bir tuşa basınız.

Şimdi *BASIC*, bütün program ve veri kütüğü isimlerini BÜYUK HARFE çevirerek, size teybin içeriğinin listesini verecek, isimleri verirken kaç blok olduğunu da belirttiği gibi ardından da ne tür olduğunu tek bir karekterle anlatır. Bu karekterlerin anlamları şöyle:

$ Standart BASIC programı
% Korunmuş BASIC Programı
* ASCII text kütüğü
& ikili (binary) kütük.

Sıranın sonundaki *"OK"* programın okunabilir olduğunu istendiği zaman yüklenebileceğini belirtir. *CAT* fonksiyonunu yapmak bilgisayarın halen hafızasında olan programı etkilemez.

2.7 Yükleme Hataları:

CPC 464 bir programı ya da verileri yüklemeye çalışırken hatalı yükleme *(read error)* anlamına gelen mesajı görürseniz, teyp işlemeye devam eder ve bilgisayar hatadan sonra bulduğu blokları okumayı sürdürür. Ancak bunlar, ilk okumaya çalıştığı (başarısızlıkla) programın 1. bloku olarak tanımlanmadıkça yüklemeye girişmez.

Yani bir *(read error)* yükleme hatası mesajından sonra teyp *(STOP/EJECT)* tuşunu kullanarak durdurur, *(REW)* ile başa sarıp tekrar *(PLAY)* ile teybi çalıştırabilirsiniz.

Bilgisayar yükleme hatası bulduğu programı yeniden yüklemiye çalışacaktır. Şansınız varsa bu sefer başarırsınız.

Yükleme hatası bir kaç nedenden ortaya çıkar. En sık görüneni bandın alıcı yüzünde, kırışma, bükülme: vs. nedenlerden olmuş bir zedelenme sonucu meydana gelir. Bir de görünüşte pek masum olan, kaset *PLAY* ya da *(REC)* ve *(PLAY)* tuşları basılıyken bilgisayarı kapatma hareketi, hatalı okumaya sebep olabilir.

Bu durumda, tuşlar basılıyken kaset en başında tutulur ve kısa bir elektrik akımı teyp kafasından geçer. Band hareketsiz kalsa bile bu akımı etkin bir şekilde bandın o parçasında olan bilgiyi karıştırır ve okunmaz hale sokar. Ayrıca ***kapstan*** ve ***pinch roller*** arasında sıkışır ve uzun süre bu durumda kalırsa buruşur.

Yükleme hatası ayrıca, kayıt ya da play sırasında *(PAUSE)* basılırsa, veya olan kayıt ilk başta teyp kafası farklı olan başka bir CPC 464 de yapılmışsa, meydana gelir.

Yükleme hatası bazen de sizin dışınızdaki nedenlerden ortaya çıkabilir. Kullandığınız kaset aslında veri kaydetmeye elverişli hazırlanmamıştır ve bu eksikliği ona hata yaptırır.

Unutmayın ki başka tip bilgisayarların programlarını içeren kasetler CPC 464 de yüklenemez. Aynı gibi durabilirler hatta ses kayıt sisteminde aynı sesleri verebilirler. Ama yüklenemezler ve çalıştırılamazlar.

2.8. KASET İŞLEMLERİNDE DİKKAT EDİLECEK HUSUSLAR

Teyp her ne kadar C 90'a kadar bir tip kaseti kabul ederse de, C12 (bir tarafı 6 dakika) veya en fazla C 30 kullanmanızda fayda var. Uzun bir bandın arka kısmındaki programları bulmak için çok beklemek, program isimlerini hatırlamak gerekir. Uzun band kullanmak istiyorsanız kasetin Numaratörünü titizlikle kullanarak bir index sistemi kurmanız gerekir. Uzun bir bandta depolanmış bir programın üstüne bir şey kaydetmek istiyorsanız sileceğiniz programın başlangıç noktasını çok iyi bilmeniz gerekir, yoksa saklamak istediğiniz başka bir programın üstüne kaydedebilirsiniz.

Genelde mümkün olduğu kadar az programı bir kasette tutun. C12 ler biraz daha ucuzdur. Yanlışlıkla zedelerseniz fazla zıyanınız olmaz.

3 BASIC diline giriş

Amstrad *BASIC* dili kullanılarak yazılmış programlara kısa bir giriş yapacağız.

Bu bölümün içerdiği konular:

* Yazılım kuralları ve yazılım tanımı.
* *PRINT* emirleri kanal numaraları ve ekran düzeni.

3 BASIC dilinin temel özellikleri:

CPC 464'ün *ROM BASIC*'i ile bilgisayarın iç işlemleri arasındaki temel ilişki EK 2'de anlatılmıştır. Şimdiye kadar bilgisayar programcılığında hiç bir deneyiminiz yoksa, size yavaş yavaş yardım edeceğiz. Ama unutmayın ki yeni başlayanların aklını karıştırabilecek bazı varsayımlarda bulunmamız gerekli. Aklınız karışırsa hiç bilmiyenler için hazırlanmış, program hazırlamanın başlangıç ilkelerini içeren bir kaç kitaba göz gezdirmenizi salık veririz.

Bu bölümü, basit alıştırmaları yaparak, tam olarak ne olup bittiğini anlamanız gerekmeden, işleyebilirsiniz. Tabii ne kadar çok kural öğrenirseniz, ilerisi için o kadar iyi olacaktır. *BASIC* dili CPC 464 de hazır olan bir dildir. Cihazınızı açtığınızda

Ready

sözcüğüyle kendini belli eder.

BASIC öğrenilecek en basit dildir. Iyi belirlenmiş bir gramer ve sözcüklerden oluşur ve kuralları anladıktan sonra tam bir mantıkla çalıştığını göreceksiniz.

AMSTRAD BASIC 8. bölümde verilen emirleri uygulama gücüne sahiptir. Her emir bir ya da daha fazla komutla tanımlanır. Komutların bazılarının isteğe bağlı olan bir kaç parametresi olabilir. Genellikle her parametre sabitler veya değişkenlerle ilgili bir ifadedir. Harflerin ve sayıların kombınasyonları karekter dizisi olarak bilinir. Kasetteki kütükler sırayla ele alınırlar. İstenmeyen kütüklerin üstünden geçmeden doğrudan istenen kütüğe gelinmez. Hepsi sıra ile geçecektir.

3.2 BASIC Programının yapısı:

Program komutları *BASIC*'e satırlarla verilir. Bir satır en fazla 255 karekterlik satır boyuyla sınırlanan ve iki nokta üstüste ile ayrılan, birden fazla emir içerebilir. Bir karekter, bir harf, rakam veya boşlukdur.

Direk yazılımda satırlar klavyeden yazılır ve bir numara ile başlamamalıdır. Program yazılımında satırlar hafızadaki en son programdan okunurlar. Satırın ilk girişi olarak görünen numaraya kesin bir şekilde uyarak uygulanırlar.

AMSTRAD BASIC Program direkt *MOD*'dayken kullanıcıya satır ekleme, çıkartma ve olan satırları değiştirme imkânı sağlar. *RUN* ve *LIST* komutundan önce *BASIC* program satırlarını küçükten büyüğe doğru düzenler. Satırların giriş sıraları değil, numaraları önemlidir.

3.3 Line INPUT

BASIC, [ENTER] tuşuyla biten 255 karektere kadar olabilen satırları kabul eder. *LİNE ınput* sırasında en son yazılanı düzeltmek ve *COPY* kürsör olanağını kullanarak ekranda görünebilen karekterleri girme olanağı vardır.
(Birinci bölüm 1.2.7'ye bakınız)

Bütün komutlar bir ayırıcıyla sınırlanmaktadır.

(Bu ara , + - veya herhangi bir tanınabilen karekter olabilir). Bunun nedeni bir komut ile değişkenin karıştırılmamasından kaynaklanır. Yani bir parçası komut olan değişkenlerin kullanım olanağı vardır. Tabii ki bir *BASIC* komut bilgisayarın komutun başını sonunu tanımasını önleyecek şekilde çevrelenmedikçe değişken olarak kullanılamaz.

Tuş sözcükleri küçük harf *(SHIFT)*'sız, ya da BÜYÜK HARF *(SHIFT)*'li girilebilirler. *PRINT* soru işareti (?) olarak kısaltılabilir. Bu durumda sınırlamaya gerek yoktur. Aritmetiksel işlemciler (+ - x / *MOD*) bir komutu sınırlamak için de kullanılırlar.

Yani aşağıdaki gibi bir giriş geçerlidir. Ama pek tavsiye edilmez. Çünkü, araların gerektiği programlar da kötü alışkanlığa yol açabilir:

For n = 1 to 5Ø : ? n : next

Aynı şekilde tek tırnak *'(SHIFT)*li *7) Rem*'in yerini tutabilir.

Fazla bırakılmış boşluklar dikkate alınmaz ve program listesinde çevrim ve benzeri yerleri belirtmek için kullanılır.

3.4 TERİMLER

BASIC emirlerini ve komutlarını tanımlamak için basit terimler kullanılmalıdır. Her emir klavyeden girildiğinde görülmesi gerektiği gibi tanımlanmalıdır. Emirlerin değişkenleri ve başka isteğe bağlı kısımları bundan sonra gelen tanımlamada ayrıntılı parçalara atıf yapan parentezler içinde gösterilir. Bu parçalar değişik isimlerle belirtilirler ve köşeli parentez [ ] içine alınırlar. Örneğin: sayı gerektiren bir çok yerde şöyle belirtilir:
[sayı ifadesi] [sayısal ifade]

Köşeli parantezin içine konulmayan her şey gereklidir. Örneğin *STOP* emri şu şekildedir:
STOP
Tanımlamada isteğe bağlı bir parça olduğunda isteğe bağlı kısmı köşeli parentez içine konur. Örneğin sayısal ifade isteğe bağlı olursa şöyle görünürler.:
[*sayısal ifade*]
İsteğe bağlı kısım tekrarlanabilirse köşeli parentezden sonra yıldız konur.
< sayı> [< sayı >]*
Bu anda şu şekilde ifade kullanılabilir.
34
ya da 334
ya da 345 678 vs.
Bir çok yerde virgülle ayrılan maddeler listesi kullanılır. Aşağıda gösterilen örneğe çok sık rastlayacaksınız.
list <satır numarası> [satır numarası]*
list# < kanal no> [# < kanal no>]
3,4
veya 3,4,4.
veya 3,4,5,6,7;8 vb.

Listede tek bir madde olabilir. Liste birden fazla madde içeriyorsa bu ek maddeden önce bilgisayarın ayrı işlem yapması gereken parçaların sınırını belirttiği için virgül konulmalıdır.
Sayılar birkaç şekilde ifade edilebilirler.
a < üssüz sayılar>
b < üslü sayılar>
Örnek
2 E4
Üs kısmı negatif ya da pozitif olabilir.
C. birde 16'lı ya da ikili sistemde yazılmış olan sayılar vardır.
onluk taban. *100*
16'lık taban *& 64 veya & H 64* (H şart değildir)
ikili taban. *& × 1100100*

3.5 PRINT KOMUTU

Bu terimlerin nasıl kullanıldığını göstermek için işte bazı *BASIC* örnekleri:
PRINT komutu doğrudan giriş moduyla ya da program moduyla bilgisayarı çalıştırabilir. Bir fonksiyon onu harekete geçirecek bir girdi gerektirir.
Örnek:

PRINT FRE (" ")

CPC 464'ün size bir sorunun cevabını vermesini sağlamak için ona üç şey söylemeniz gerekir.

1- Cevabın nereye çıkmasını istediğiniz (ekran, yazıcı, ya da başka bir yer)
2- Bilgisayara kullanacağı verileri vermelisiniz.
3- Bilgisayara, verilenlerle nasıl bir işlem yapması gerektiğini belirtmelisiniz.
PRINT komutu bilgisayara yazacağı sonucu belli bir kanal kullanarak yazmasını söyler. Bu kanalın numarası 0 ile 9 arasında değişir.
0.....7, daha önce *WINDOW* emriyle kullanılmış metin "pencere"lerini belirten kanallardır.
8. kanal yazıcı içindir. Ancak doğru bir biçimde bağlanmışsa kullanılabilir.
9. Programın başında doğru bir biçimde açılmış olması gereken kaset kanalıdır.
PRINT emrinin başka şekli *(PRINT USING'* kullanılmadan) şöyledir:
PRINT [# < kanal numarası>] [< print maddeleri >].... şimdilik. Köşeli parentezler ne kanal numarası, ne de *PRİNT* emrine yazacak başka bir şey vermeniz gerekmediğini anlatıyor. (bu durumda bilgisayar boş bir satırla karşılık verir. deneyin) Çıkışı özel bir kanala yöneltmediğiniz takdirde CPC 464 0 kanal numarasını varsayar. Aşağıdaki satırı deneyin. Bilgisayara verdiğiniz komutu işleme sokmasını belirtmek için satır sonunda *[ENTER]*'a basmayı unutmayın.

PRINT "MERHABA"
Bilgisayarın cevabı şu şekilde olur.
MERHABA
Çifte tırnak işareti *BASIC*'de prınt maddesinin nerede başlayıp nerede bittiğini kesin belirtmek için kullanılır.
Şimdi şunları yazalım.
PRINT # 0, "MERHABA"
.... sonuç aynıdır.
Ama şöyle yazarsanız:
PRINT # 4, "MERHABA"

Bilgisayar sonucu ekranın sol üst tarafına koydu. Çünkü, daha önce *WINDOW* emriyle belirlenmediği takdirde bütün metin alanını kaplayan kanal numarası 4'ün ekrana ilk girişi budur. Bir ekran kanalı için metin başlama durumu sol üst köşedir. Kanal 4 henüz kullanılmamıştır. Başlangıç mesajı için kanal 0 kullanılır. Bu yüzden metin burada görüntülenen karekterlerden sonra ortaya çıkan kanala gönderildi. Düzenli yerleştirilmek ekran görüntüsünü kolaylaştırmaya katkıda bulunan basit emir ve tanımlamaları kullanarak, güç ekran görüntüsü sağladığı için *AMSTRAD BASIC*'ın bu özelliği çok önemlidir.
PRINT komutlu çifte tırnak işareti koyduğunuz herhangi birşeyi yazar
PRINT " 4 * 4 "
Bilgisayarın cevabı şudur:
4 * 4
BASIC dilinde direk modda bir çarpma işlemi yapmak için şunları yazmak gerekir.
PRINT 4 * 4
16
Dikkat ederseniz sayı sol sınırdan bir sütun ileri yazıldı. Çünkü *BASIC* bu sütunu eski bir sonucu tanımlayan (-) işareti için saklar.
PRINT emrinde < Prınt maddeleri > yazılacak maddelerin listesinden atıf yapar. Bu bir sayı, değişken, bir karekter dizi değişkeni *(ör MERHABA $)* ya da çifte tırnak içine konulmuş her hangi bir şey olabilir.
PRIINT USING sütunların aynı sıralarda olabilmeleri için ve istenmeyen parçaların atılması için belli bir düzene uygun olarak sayıları derli toplu yapar.

3.6 PRINT USİNG

Açılışta *BASIC* enini 13 sütun genişliğinde konuma ayarlar. Prınt komutunda bir virgül olduğunda bir sonraki parça bir sonraki sütun konumuna yazılır. Bir sütunda *ZONE* emrinde belirtilenden daha az kolon varsa o zaman *BASIC* ondan sonra yazılacak maddeye yeni bir satır başlatır. Satırın kenarına *USING* emri belirlenmedikçe *BASIC*, pozitif sayıları bir boşluktan sonra negatif sayıları da - işaretinden sonra yazar. Bütün sayılardan sonra ara vardır. Ondalık nokta yoksa, yazılanda ondalık nokta da konulmaz.

AMSTRAD BASIC (TAB) tuşunu sütun ileri almak için kullanmaz. *TAB* tuşuna basmakla, sağa doğru ok işareti (→) yazılır. *(CTRL* ve *I* tuşuna birlikte basıldığında olduğu gibi) ve bunun *AMSTRAD BASIC*'de başka amacı yoktur.

3.7 PRINT TAB (değeri tam sayı olan bir girdi) (prınt maddeleri)

Bunun etkisi en iyi bir örnekle gösterilebilir..

```
 5 MODE 2: INK 1,Ø : INK Ø,9
1Ø FOR N=1 105
2Ø ZONE 40
3Ø PRİNT
4Ø NEXT TAB (N*4); "OK" N
```

Bu program, hem sütun ayarlaması ile birlikte, hem de *TAB* ( ) işlevlerini çalışır halde gösterir. Satır 10'u aşağıdaki gibi değiştirerek yeniden çalıştırın
1Ø FOR N = 5 to 5
Tab komutu kursörü tab komutunun girdisi kadar ileri götürür. *(ZONE* 1...255 arası moda ayarlanabilir. Bölüm 8'e bakın).
PRINT USİNG sonuçların ondalık kısmı olsun olmasın aynı düzende çıkmasını sağlar. Örnek olarak şunu verebiliriz.

PRINT [# (kanal numarası),] [Prınt maddeleri] [(kullanma cümleciği)] [(ayırıcı)]
Bu haklı olarak kullanıcıya zor gelebilir Özellikle kullanma cümleceği tekrar ikiye ayrılır:
USING (karakter dizisi); [(using maddeleri)]
Using listesi de ikiye ayrılır:
İfade [(ayırıcı) (ifade)]*
Aşağıdakini deneyin:
PRINT 123.456, USING " # # # . # # ; 4567.896
Sonuç:
123.456 % 4567.90

Bu bizi biraz aydınlatıyor. *USING*'den önce yazılan *USING*'den etkilenmez. İkinci olarak *USING* ardından gelen (bu bir değişken de olabilir) mümkün olan çıkışın konum sayısının yerini ayırı ve bu parça ayrılan yerde ondalık noktanın soluna doğru geçerse yine de görüntülenir ama bir aşma meydana geldiğini belirtmek için % işareti kullanılır. Bundan başka *123.456* dan sonraki virgül ondan sonra gelen sayının bir sonraki prınt zone'a yazılmasına sebep oldu. Eğer bu bir noktalı virgül olsaydı sayı *123.456* nın yanına yazılırdı. Aynı satırdaki sayılar her zaman bir ara ile ayrılırlar. Sayıların yuvarlanmış olduğuna ve düzenin Usıng ile belirtilene uygun olduğunu farketmişsinizdir.

Şunu deneyin:
PRINT 123.456, USING " # # # # # # # # # -"; 4567.899
....Ve işaret ssayının dışınnda beliriyor. Negatif sayının başına eksi işareti kendiliğinden gelir.

PRINT USING bu çeşit düzenli sonuçlar çıkartmak için çok yararlıdır. Belirlenmiş düzen fazla kısıtlıysa her zaman sizi uyarır (% kullanarak)

4

DEĞİŞKENLER, İŞLEMCİLER ve VERİLER

Bu bölümün içerdiği konular:

* Tanışmak
* Değişken türleri: gerçek, tam sayı, karekter dizileri
* İşlemciler, mantık ifadeler
* Matrisler
* Veriler.

4.1

Dikkat edin ki (şimdiye kadar farketmişsinizdir zaten) *AMSTRAD BASIC*'de emirler ve komutlar ya aralarla ya noktalama işaretleriyle vs sınırlandırılmışlardır (sınırlar belirlenmiştir) Programların okunması ve düzeltme daha kolay olur. Çünkü, komutları büyük harf ve küçük harfle girebilseniz liste istendiğinde hepsi zaten büyük harfe çevriliyorlar. Büyük harfe çevrilmezlerse yazılışlarında bir hata var demektir. Ve program çalışmaz.
AMSTRAD BASIC değişken isimlerin içinde komutlarında bulunmasına izin verir
AMSTRAD BASIC'de değişken bir harf kadar basit olabilir. (değişkenler mutlaka bir harfle başlamalıdır, bir sayıyla değil)
Uzun programlarda yapılan işi yansıtan bir değişken kullanırsanız, okuması daha kolay olur:

SONUÇ = 4 * 4 : PRINT SONUÇ

AMSTRAD BASIC'de değişkenler 40 karektere kadar olabilirler (ilk karakter harf olmalıdır) ve bu karekter belirleyicidir. Değişkenlerde hiç ara olmamalıdır. Olursa *BASIC* ilk araya kadar olan harfleri (veya harf ve sayıları) okur ve şu cevabı verir:

Syntax error.

Bu da kural dışı yazılım girildiğini belirtir. (EK 8'e bakın) Kelimelerin ayrıldığı cümlecikleri kullanmak isterseniz normal ara koyacağınız yerlere nokta koyun. Aşağıda yazılı bütün değişken biçimleri olasıdır. Yalnız *DIM* komutunun gereğini aklınızdan çıkartmayın. (Matris kullanımında)

4.2

Her zaman *PRINT* yazmak sıkıcı olabilir. Onun yerine (?) kullanabilirsiniz ve *BASIC* onun *PRINT* anlamına geldiğini anlar. (Tabii bir cümlecikle onu iki tırnak " " içinde kullanma-

dığınız sürece) Unutmayın ki(?) ***PRINT*** yazısında olduğu gibi ara ile sınırlandırılması gereği göstermez. Direkt emir yerine program olarak bir satır yazarsanız:

10 ? 4 * 4

RUN

Yanıt hala aynıdır ama bu tek satırlık programı ***LIST*** edin

LIST

10 PRINT 4 * 4

BASIC soru işaretini ***PRINT*** yazısına çevirmeden ara da koyar. Tırnak işareti kullanan ***PRINT*** bildirilende:

10 ? "MERHABA"

Dalgınlık yapıp sondaki tırnak işaretlerini de koymayabilirsiniz, ***[CTRL] [ENTER]*** tuşlarına basım işleminden sonra ekrana ***"RUN"*** yazılmasının da kanıtladığı gibi. Aynı şey program satırları için de geçerlidir ama, edinilecek pek iyi bir alışkanlık değil, çünkü sonradan dönüp satıra eklerseniz tırnakları kapatmayı yine unutabilirsiniz.

4.3 Çok bildirili satırlar ve karışık hesaplamalar

Tek bir ***BASIC*** satırında bir çok işlem yapabilirsiniz. Daha doğrusu en fazla uzunluk olan 255 karekteri geçmeden istediğiniz sayıda işlem koyabilirsiniz. Her zamanki gibi bildirilen iki nokta üstüyle ayrılmalıdır:

? 2 * 8/5 + 5 - 4 * 777 E 9/3

..... sonucu

- 1.0 36 E + 12

BASIC'in matematik işlemcileri tanıyış sırasını anlamak çok önemlidir. Yoksa çok temelden yanlışlar yaparsınız. Bu sıra şöyledir: üs işlemleri, bir sayıyı belirli bir üsse çıkartma.

- sayıyı negatif olarak belirtmek için gerekli - işaret.
/ bölme
Bölme: sonuç sonucun tam sayı bölümünü versin diye kısaltma (ondalık kısmı BASIC tarafından atılır)
* Çarpma
+ Toplama
- Çıkartma.

Parantez içindeki işlemler önce yapılır. Parantezin içinde karışık hesap varsa, bunlar da yukarıda belirtilen sırayla ele alınır. Parentez içinde parentez varsa yine öncelik kazanır. Böyle bir ifadede ne kadar parantez açma işareti varsa o kadar da parentez kapama işareti olmalıdır. Aksi takdirde ***syntax error*** yazısı ile karşılaşırız.

*3.5 de **PRINT*** bildirisini tanıttığımızda ri epey yol katettik. Yapmacık hesap durumlarından bilgisayar programlamaya geçecek kadar ***AMSTRAD BASIC*** temel kuralları kapmış olmanız gerekir. ***BASIC*** komutları gerektikçe tanıtılacaktır.

Direk ***MOD***, birden fazla emirleri iki nokta üstüste: ile ayırarak girmenize izin verir. Ancak bir kere ***[ENTER]*** tuşuna basarak. Satır bilgisayara girdikten sonra komutlar işlem görür ve satır kaybolur. Her zaman ***Copy*** kürsör özelliğini kullanarak, satır hala ekrandaysa, tekrar o satırı kullanabilirsiniz.

4.4. İLERİYE DOĞRU ADIMLAR

Paragraf 3.5'de tanıştırdığımız PRİNT komutundan beri bir hayli yol aldık. Şimdiye kadar öğrendikerimizden AMSTRAD BASİC Dilinin ana kurallarının hesabın dışında olanaklarda sağlanmış olduğunu gördünüz. Bundan sonra Basic komutları gerektiğinde kullanılacak

ve anlamadığınız bir komut olursa 8.Bölümden alfebetik sıraya göre bakabilirsiniz. Birçok BASİC komutu İngilizce lisanında kullanıldığı gibidir. Örneğin; GO TO 50, programda 50. satıra git demektir. END (bitti) programın sonunu belirtir. Program yapmadan direkt komutları makine kabul eder fakat aynı işlemi yapmak için bir daha yazmak gerekir. Ancak komutunuz halen ekranda ise "COPY" kursörü ile bu kopyalıyabilirsiniz.

4.5 Şart ve Mantık Deyimleri:

BASIC bilgisayarın basit ve tekrarlanan görevleri hiç sıkılmadan - yapabilme gücünden fazlasıyla yararlanır. Bu işlemleri meydana getirmek için bir kaç tane programlama emiri sağlanmıştır. Daha önce saptanan bir takım şartlar yerine getirildiğinde, çevrim başlatan, sürdüren ve ne zaman biteceğini bildiren emirlerdir..

Bu çevrim kontrolu elemanlarının sonuncusu şart deyimi ile ilgilidir. Yani bir veri parçasının bir başka veri parçasıyla ilişki kurma biçimi bir karşılaştırıcı tarafından saptanır. Bir değişken bir başka veri ile, ya da bir değişkeni bir sabitle karşılaştırabilirsiniz.

Karşılaştırıcılar şunlardır:

< Küçüktür.
< = Küçük veya eşit
= Eşit
> Büyüktür
> = Büyük veya eşit
< > Farklı

İşte size bu karşılaştırıcıları, bize çok açık olarak, gösterecek bir kısa program. Halen ekranda açılış mesajını görmüyorsanız

Amstrad 64 k Microcomputer (v1)
c 1984 Amstrad Consumer Electronics pk
and Locomotive Software Ltd
BASIC 1.0
Ready

o halde *(CTRL) (SHIFT)* ve *(ESC)* tuşlarına basın.

Bilgisayarda yukarıdaki mesajı göreceksiniz.

Şimdi devam edin ve aşağıdakileri yazın. (1.27 de verilen komutları değiştirerek)

```
10 İNPUT "MAAŞINIZ NE KADAR"; MAAŞ
20 İF MAAŞ > 100000 THEN GOTO 30 ELSE 40
30 PRİNT "KENDİNİZE BİR RENKLİ MONİTÖR ALIN": END
40 PRİNT "AYLIĞINIZA ZAM İSTEYİN"
```

RUN

ve tabii şart olan *[ENTER]*

MAAŞINIZ NE KADAR?

(Bilgisayar sizden verileri istediği zaman soru işaretini otamatik olarak koyuyor.) Soruyu yalnızca sayı kullanarak yanıtlayın. - harf, para işaretleri veya virgül kullanmayın, cevabını *[ENTER]*'e basarak girin.

Ready sözcüğü tekrar çıktığında yukarıdaki işlemlerin altına satır besi ekleyin.

```
5 CLS
```

Ekranı silmek için tekrar *RUN* yazın. İlk yanıtınız 100000 den az idiyse bunu 100000 den çok yapıp farkı görün. Şimdi bu ilk programı, alta 35 ve 36 ilave ederek genişletin.

```
 5 CLS
10 İNPUT "MAAŞINIZ NE KADAR" ; MAAŞ
20 İF MAAŞ > 100000 THEN GO TO 30 ELSE 40
30 PRİNT "KENDİNİZE BİR RENKLİ MONİTÖR ALIN"
35 İF MAAŞ > 300000 THEN PRİNT "BİRDE YAZICI SATIN ALABİLİR-
SİNİZ"
36 END
40 PRİNT "AYLIĞINIZA ZAM İSTEYİN"
```

36. satırdaki *END* komutu eğer maaşınız 100000 TL den fazla ise programın durmasını sağlıyor. Böylece ekranda birde AYLIĞINIZA ZAM İSTEYİN cümlesi gözükmüyor. Şimdi programı aşağıdaki gibi değiştirin.

```
 5 CLS
10 İNPUT "MAAŞINIZ NE KADAR" ; MAAŞ
20 IF MAAŞ > 100000 THEN GO TO 30 ELSE 40
30 PRİNT "KENDİNİZE BİR RENKLİ MONİTÖR ALIN"
34 İF MAAŞ < 300000 THEN PRİNT "PARA BİRİKTİRİN": GO TO 36
35 PRİNT "BİRDE YAZICI SATIN ALABİLİRSİNİZ"
36 END
40 PRİNT "AYLIĞINIZA ZAM İSTEYİN"
```

Sonucun farklılığını göreceksiniz.

Bu noktada, şimdiye kadar işlenen konuları kullanarak kendi programınızı kurun. Bu programın nasıl "büyütüldüğüne" dikkat edin. Çoğu programlar bu şekilde gelişir. Bu da, programı yapmakta *BASIC*'in ideal düzenleme yeteneğini kullanan en önemli sayılabilecek kavramı tanıtıyor.

4.6 PROGRAM OLUŞUMU

Program ilerledikçe (büyütülebilme) özelliği *BASIC*'in en kullanışlı özelliğidir. Bazıları bu kullanışlılığın, yani yeni fikirler geldikçe programa ekleyiverme olanağının, düzensiz ve "bilgisiz" program tekniklerine yol açacağını savunabilirler; gerçekciler ise bunun, ilerlemelerini kolay evrelerde kontrol etme olanağı bulan öğrecinin ilgisini uyanık tutmanın en iyi yolu olduğunu düşünebilirler.

Örnek programımızı yeniden ele alalım ve ekran cevabını okuyacak zaman bırakan duralamadan sonra programı başa dönmeden biraz bekletin 30 ve 36. satırı ekleyelim.

```
36 for n = 1 to 900 : next : go to 5
50 for n = 1 to 900 : next : go to 5
```

Bu yeni satırın ve eklenen öteki satırlarının *AMSTRAD BASIC*'in değişken bir isimle komut arasındaki farkı anladığını size hatırlatmak için küçük harf ile girildiğine dikkat edin.

Bu programı kesmek için *[ESC]* tuşuna iki kez basın ve *LIST* isteyin. Bilgisayarı, komutları BÜYÜK HARFE çevirdiğini ama, *n* değişkenini küçük harf bıraktığını görüyorsunuz.

50. satır bir geciktirme çevrimini tanıtıyor. Bilgisayar 1 den 900 e kadar saydıktan sonra

GO TO 5 komutunu yerine getiriyor. Böylece program bitmeden kendini yeniden çalıştırıyor. Bunu kesmenin tek yolu ***[ESC]*** tuşunu kullanmaktır. Bu tuşa bir kez basmak, programı durdurur, bir kez daha basmak, programın içeriğini hafızada kaybetmeden, bilgisayarı direkt emir moduna geçirir.

Aslında, bilgisayar, giriş beklerken hiç bir şey yerine getiriyor olmadığı için 50. satırda daha geciktirme çevrimlemesi içindeyden ***[ESC]*** tuşuna basmanız, program hemen durur. Kaldığı satır durduktan sonra bilgisayar, ekranında çıkar. Giriş beklerken ***[ESC]***'e basarsaanız şu mesajı alırsınız.

Break in 10

Geciktirme çevriminde yakalarsanız *(ESC)* tuşuna 2. basışınızdan sonraki mesaj şudur:

Break in 50

50. satırda çevrim sırasında ***[ESC]*** tuşuna basarsanız işlemi bir süre durdurursunuz ve herhangi bir tuşa basarak kaldığınız yerden devam edebilirsiniz.
Programı durdurursanız, yine durduğunuz yerden şöyle girerek devam edebilirsiniz.

CONT

ve program işlemlerin durduğu yerden devam eder.

[ESC] tuşuyla ne yaparsanız yapın, en son hafızada olan programı kaybetmezsiniz. Hafızadaki programı, özellikle temizleme emri olan ***NEW*** emrini veya bilgisayarı ilk konumuna getiren ***[CTRL] [SHIFT]*** ve ***[ESC]*** tuşlarına basmazsanız kaybetmezssiniz.

Bu yüzden "dalgınlıkla" programı silmemek için özel bir koruma tedbirine gerek yoktur. Hafızadaki programı silmek işi gayet bilerek yapılan bir şeydir ve dönüşü yoktur, kalıcıdır.

Tekrar kullanma ihtimaliniz olan programları korumak için kasete çekin.

4.7 BAŞKA DEĞİŞKENLER VE KAREKTER DİZİLERİ

Bilgisayar programcılığının özü değişkenlerdir.. Bilgisayar yalnızca belirli olanaklarla iş yaparsa o zaman elektronik bir yazma aracından başka bir şey olamaz. Unutmayın ki matematiksel ifadelerin bir parçası değişken ise sonuç da mutlaka değişken olmalıdır. Değişkenlerin üç özelliği ya da karekteristliği vardır: bir ismi, bir tipi ve birde yapısı isimlerinin üstünde daha önce duruldu. (4.1) tipler isteğe bağlıdır. Bu yüzden bir değişkeni (3.4) deki kurallara göre şöyle tanımlayabiliriz:

(isim) [(cinsini gösteren sembol)]

Cins gösteren semboller şunlardır:

%; ondalık noktanın sağındaki şeylerin atıldığı sayılar Tam sayı, değişkenleri hafızada daha az yer tutarlar, dolayısıyla ondalık çarpma gerektirmeyen programlar, değişken ***(DEFİNT)*** tam sayı olarak tanımlanırsa, daha hızlı çalıştırılabilirler.. ***DEFİNT*** emri değişkeni tam sayı olarak tanımlar değerleri -32768 den + 32767 arasında sıralamadır. !.; Bir değişkenin gerçek olduğunu bildirir. Gerçek sayı tam kısım ve ondalık kısım olabilen sayı demektir. Değişkenler bilgisayar açıldığında, gerçek kabul edilirler, bu yüzden daha önce ***DEFİNT*** emrini kullandınızsa yalnızca gerçek değişkenleri tanımlamanız gerekir. Gerçek değişkenler 2.9E - 39 dan 1.7E + 38'e kadar bir değer alabilirler.

$ karakter dizi değişkeni belirtir. Yani değişkenin içeriği harflerle, rakamlar vs olabilir

Bir başka deyişle iki tırnak içine alınacak istenilen karakterler dizisi.. Örneğin:

İsim $ = "AHMET SÖKMEN"

Bunu evrimli örneğimizde kullanarak 6. satırı ekleyin ve 40. satırı yeniden düzenleyin.

6. İNPUT "İSMİNİZ LÜTFEN"; İSİM $
40 PRİNT İSİM $; "AYLIĞINIZA ZAM İSTEYİN"

PRINT ve *INPUT* komutlarından sonraki; noktalı virgül bilgisayarın, aksi söylenmedikçe, her yazılacak için yeni bir satır başlama isteğini bastırır.

Tam sayılar konusunda 1. satırı ekliyerek de çalışabiliriz. Siz yalnızca yazın, bilgisayar programı düzenler:

11 GÜN. KAZ = MAAŞ/30: PRİNT "GÜNLÜK KAZANCINIZ" ; GÜN. KAZ.
36 FOR I = TO 5000 : NEXT : GOTO 5
50 FOR J = TO 5000 : NEXT : GOTO 5

Dikkat ederseniz 50. satırda değişiklik yaptık. Çünkü ekranda okuyacaklarınız daha fazla bu yüzden bilgisayar daha çok beklemeli.

Bir de şunu ekleyelim.

12 TAM.SA % = GÜN. KAZ: PRİNT "EĞER KURUŞLARLA İLGİLENMİYORSANIZ" ; TAM.SA %

Şimdi yine çalıştırın *[RUN]*

Dikkat: Cins gösteren semboller, %'yi kullanmaya devam etmeniz gerekiyor. Çünkü, tam sayı değişkenler aynı zamanda bir gerçek değişken olabilir ve bunların farkı % işareti ile anlaşılır. Program satırı çok uzun olduğundan devamı bir alttaki satıra taşır. Uzun programlar yazmak için *MOD 2*'yi kullanın, çünkü satırların sonundan taşmıyan programları okumak daha kolay olur.

MOD 2'yi kullanmak için yazın:

MODE 2

ve CTM 640 da okuması daha kolay olan beyaz üstüne siyah görüntü elde etmek için aşağıdaki direkt emirleri yazın:

INK 1 0
INK 0,13
Border 13

Şimdi yine liste isteyin.

4.8 GÖRÜNTÜ DÜZENİ

Programınızın gelişiminin bir parçası da zaman zaman onu tertiplemektir. Yapabileceğiniz ilk şey programı bütün satırları 10 çarpımlarıyla, *RENUM* emri kullanarak, yeniden numaralamaktır.
Ready sözcüğü gelince şu emri verin.

RENUM
VE YİNE *LİSTE* isteyin.

10 CLS
20 INPUT "İSMİNİZ LÜTFEN" ; isim $

```
30 INPUT "MAAŞINIZ NE KADAR" ; MAAŞ
40 GÜN.KAZ = MAAŞ/30 : PRINT "GÜNLÜK KAZANCINIZ"; GUN.KAZ.
50 TAM.SA %% = GÜN..KAZ: PRINT "EĞER KURUŞLARLA İLGİLENMİ-
YORSANIZ" ; TAM.SA%
60 İF maaş 100000 THEN GOTO 70 ELSE 110
70 PRINT "KENDİNİZE BİR RENKLİ MONİTÖR ALIN"
80 İF maaş 300000 THEN PRINT "PARA BİRİKTİRİN" : GOTO 100
90 PRINT "BİRDE YAZICI SATIN ALABİLİRSİNİZ"
100 FOR n= 1 TO 5000: NEXT: GOTO 10
110 PRINT isim $ ; "aylığınıza zam isteyin"
120 FOR 1 = 1 TO 5000 : NEXT : GO TO 10
```

Bütün satırlar, programın bütünü içinde atıf yapılanlar dahil, toparlanmış. Zaten *BASIC* bütün numaralarının hesabını tutmayıp hep en son şekline göre sıralamasaydı pek işe yaramazdı.

Şimdi ekrandaki görüntüyü biraz tertipleyelim. Bunun için önce 110. satırdaki çevrimi, bir emirden, *(REM)* komutuyla bir uyarı satırına çevirin.

```
10 CLS
20 INPUT "İSMİNİZ LÜTFEN" ; isim $
30 INPUT "MAAŞINIZ NE KADAR" ; MAAŞ
40 GUN. KAZ = MAAŞ/30 : PRINT "GÜNLÜK KAZANCINIZ" ; GUN.KAZ.
50 TAM. SA% = GUN.KAZ: PRINT "EĞER KURUŞLARLA İLGİLENMİYOR-
SANIZ" ; TAM.SA%
60 İF maaş > 100000 THEN GO TO 70 ELSE 110
70 PRINT "KENDİNİZE BİR RENKLİ MONİTÖR ALIN"
80 İF maaş > 300000 THEN PRINT "PARA BİRİKTİRİN": go to 100
90 PRINT "BİRDE YAZICI SATIN ALABİLİRSİNİZ"
100 FOR n = 0 to 5000: NEXT: GOTO 10
110 PRINT isim $ ; "aylığınıza zam isteyin"
120 REM: FOR 1 = 1 TO 5000 : NEXT: GOTO 10
```

REM KOMUTU:

Satırın başına *REM* koymak bilgisayarın, satırın gerisini dikkate almadan geçmesini sağlıyor. Yani o zaman program bitiyor ve *Ready* sözcüğü geliyor. Şunları yazalım:

```
15 mode 1
```

15. satır önce görüntü modunu ayarlar. Öyle ki program çalışırken *[RUN]* hangi yürürlükte olursa olsun, 15. satırı önce *MOD 1*'e çevirin. *MODE* emri otomatik olarak *CLS* yapar. Bu yüzden 10 satır artık fazladan duruyor, ama yinede yerinde bırakalım.
Şimdi programı çalıştırın ve cevaplayın.

```
İSMİNİZ LÜTFEN? ALİ
MAAŞINIZ NE KADAR? 80000
GÜNLÜK KAZANCINIZ 2666.66667
EĞER KURUŞLARLA İLGİLENMİYORSANIZ 2667
ALİ AYLIĞINIZA ZAM İSTEYİN.
```

eğer ekrandaki görüntü hoşunuza gitmediyse aşağıdakileri ekleyin.

```
25 PRİNT
35 PRİNT
45 PRİNT
```

55 PRİNT
75 PRİNT
85 PRİNT
aradaki farka görmüşünüzdür. Hiç birşey yazılmaksızın *PRİNT* sadece satır atlatır.

4.9 LOCATE KOMUTU :

Buraya kadar *BASIC* komutları, BASIC dilini içeren bilgisayarların çoğunun anlıyabileceği *BASIC* yazılımı kullanılarak yazıldı. *LOCATE* kürsörü ekranın istediğiniz yerine koymanızı sağlayan *AMSTRAD BASIC*'in bir özelliğidir.

LOCATE 10, 4

Bu komut kürsörü,10. sütuna ve yukarıdan 4. satıra getirir. Bunu direkt komut kullanarak yapmayı denerseniz, kürsör yerine gelir ama, ***Ready*** sözcüğü yazılır ve kürsör yine ekranın soluna gelir, yeni satıra hazırlık yapar. 16. satırı ekleyin:

16 LOCATE 10, 4

run

Görüyorsunuz ki ***input*** komutunda kürsör yer değiştirdi. Bir sonraki satır eskisi gibi sürüyor. 25. satır bir kaç boş satır ekliyor. *[ESC]* tuşuna iki kez basın ve 26. satıra şunları ekleyin:

26 LOCATE 10, 4

Programı çalıştırın. İkinci soru mesajının yazısı birincinin üstüne yazılır. Şimdi artık metni istediğiniz yere yerleştirerek devam edebilirsiniz. EK 6'da ekranın tüm bir planı koordinatlarıyla beraber verilmiştir. Bütün soruların ve yazıların aynı satırda görünmesini istiyorsanız, bu *LOCATE* komutundan önce *CLS* yazınız.

LOCATE koordinatları programdan kaynaklanan değişkenler olabilir. Maaş programını artık eskittik. Onun için bundan sonraki örneği başka bir komuttan alalım. Buraya kadar yaptıklarınızı kaydetmek istiyorsanız 2. Bölümdeki emirleri kullanarak kasete alıp saklayın. Belki de bu programı sevmiye başlamışsınızdır ve ona ileride tekrar bakmak için saklamak istersiniz. Birazdan tanıtacağımız emirlerle daha da geliştirmek istiyebilirsiniz.

4.10 IF.....THEN

Bu komutun kullanışı ingilizce kelime anlamına uygundur.

[İF (EĞER)...*THEN* (O HALDE)] *İF* (mantıksal işlemci)
THEN GOTO (satır numarası) Bu, emrin bir kaç kullanılışından biridir.
IF (mantıksal işlemci) ın sonuçlarının doğru olup olmadığını kontrol eder. Doğruysa seçenek yerine getirilir. *IF* operasyonu tekrarlanan çevrimler olarak kullanılabilir.

[CTRL] [SHIFT] ve *[ESC]* tuşlarıyla bilgisayarı ilk konumuna getirin ve aşağıdakileri yazın:

```
1 MODE 1.
2 AMSTRAD = 0
10 PRİNT "AMSTRAD CPC 464"
20 AMSTRAD = AMSTRAD + 1
30 İF AMSTRAD < 10 GOTO 10
```

Bunu çalıştırın ve 10. satırdaki print komutunun 30. satırdaki koşul yerine getirilinceye kadar tekrarlandığını görün. Böylece 30.ıncı satır programı 10.cı satıra geri gönderiyor. Değişken terimini, *AMSTRAD*ın programı her çevrimlemesinde değerinin değişmesindeki önemini görün.

Bu programda *AMSTRAD* değişkenine ne olduğunu görmek istiyorsanız 25 nolu bir satır daha ekleyebilirsiniz.

25 LOCATE 1,20: PRINT AMSTRAD : LOCATE 1, AMSTRAD
run

Bu çok hızlı geldiyse geciktirme çevrimiyle yavaşlatın:
26 for n = 1 to 500 : next
Şimdi (eğer CTM 640 seçeneğiniz varsa) biraz renk ilave edin şöyle:

24 BORDER AMSTRAD

Bu satır border rengini 20. satır ile *AMSTRAD*'ın değerine ayarlıyor. Yine programı ***list***'den isteyin.

1 MODE 1
2 AMSTRAD = 0
10 PRINT "AMSTRAD CPC 464"
20 AMSTRAD = AMSTRAD + 1

24 BORDER AMSTRAD
25 LOCATE 1,20 : PRİNT AMSTRAD : LOCATE 1, AMSTRAD
26 FOR n = 1 TO 500 : NEXT
30 İF AMSTRAD < 10 GOTO 10

Kuşkusuz hepiniz CPC 464 in renk olanaklarını görmek ister. O halde 40.satırı şöyle değiştirin:

30 IF AMSTRAD > 26 GOTO 10

Programı çalıştırın. Bütün renk olanaklarını en koyudan başlayıp en parlak beyaza kadar göreceksiniz. 25.satıra *BORDER* sözcüğünü ekliyerek Border renk numara mesajını daha yararlı yapabilirsiniz.

25 LOCATE 1, 20: PRINT "Border" ; AMSTRAD: LOCATE 1, AMSTRAD

run

Hazır *Border* konusundayken, program bitip de *Ready*'e döndüğünde şunları yazın:
BORDER 14,6
Göreceksiniz ki border 14 ve 6 renkleri arasında değişiyor. Grafik renkler üstüne daha fazla bilgi için gelecek bölümü beklemeniz gerek. Bu yan bölümü toparlamak için, şunları yazalım.
ınk 1, 18, 16
sonra....
speed ınk 1,5
Şimdi bilgisayarı ilk durumuna getirin. Programı arkadaşlarınıza gösteriş yapmak için saklamak istiyorsanız, kasede kaydedin.

4.11 MATRİSLER

Bazı programlar verileri depolamak için çok değişken gerektirir. İyi hoş da çoğu kez hangi değişkenin, hangi veri için kullanıldığının hesabını tutmak güçtür. Neyse ki *BASIC* bu durumun da veri matrisleriyle üstesinden geliyor.

Matris nedir? diye sorabilirsiniz. Esasında hepsi aynı adla çağrılabilen bir grup değişkendir.

En iyi anlatma yolu örneklemektir. Bir iskambil oyununu yansıtan ve seçilmiş kartların rastgele dağıtılmasını söz konusu olan bir programı düşünün. Besbelli ki aynı kartı iki kez ağıtmak olmaz ve bu kartın hesabı tutulmalıdır. Bunu yapmanın en kolay yolu her kart için bir değişken tayin etmek ve o değişkeni kartın yerine göre ayarlamaktır. Örneğin 1 dağıtıldığını 0 da destede olduğunu belirtsin. Belli ki bu 52 ayrı değişken söz konusu demektir. Ve hangi değişkenin hangi karta karşı düştüğünü hatırlamanız gerekir. İşte burada matrisler çok işe yarar.

Önce bir matris ismi vermeniz gerek, örneğin *DESTE.* Şimdi, bir matrisdeki belli bir değişken ya da elemana ulaşmak için ona yalnızca bir numara veririz. Dolayısıyla ilk 13 eleman kupaları tanımlamak için kulanılıyorsa o zaman altılı *DESTE(6)* onlu *DESTE (10)* la papaz da *DESTE (13)* le gösterilir. Biraz daha anlaşılır oldu mu?

Maalesef sonsuza dek, bilgisayara hafızada yer ayırması için uyarmadan, elemanlara atıf yapmayı sürdüremezsiniz. *DIM* emri bunu sağlamak için kullanılır.

DIM emri matris'in yerini ayarlar. Örneğin, bir deste kart örneğinde elliiki elemanın yeri ayrılması gerekiyordu.

Gereken *BASIC* komut aşağıdaki gibi yazılır.

DIM DESTE (52)

Bu hafızada elliikilik yer ayırmasını söyler. (Aslında elliüç. Çünkü, sıfır elemanınada ulaşılabilir).

Şimdi kart dağıtma alt programı olarak şunları yazabiliriz.

```
10 DIM DESTE (52)
20 FOR X = TO 52
30 LET DESTE (X) = 0
40 NEXT X
1000 KART = RND (52) + 1
1010 IF DESTE (KART) = 1 GOTO 1000
1020 DESTE (KART) = 1
1030 RETURN
```

DİM komutunun programın ilk satırında olduğuna dikkat edin. Bunun nedeni bir matrisin yalnızca bir kez boyutlarının saptanabilmesidir. Program ilerledikçe yeniden saptanamaz. 20. ve 40. satırlar, matrisin elemanlarını sıfıra ayarlıyor. Alt program 1000. satırda başlıyor, sonra rastgele bir kart seçiyor ve henüz dağıtılmış olup olmadığını kontrol ediyor. Seçilen kart dağıtılmışsa, dağıtılmamış bir tanesi bulununcaya kadar aranıyor. Program kartın şimdi dağıtıldığını anlatmak için matrisde uygun elemanın değerini değiştirir ve tekrar alt programdan döner.

Matris kullanmak tek bir boyutla sınırlanmamıştır, istenilen boyutlara genişletilebilir. Bu değişkenlerin sayılarına daha fazla referans eklemekle yapılabilir. Örneğin 10 x 10 x 10 luk bir matris aşağıdaki emirle ayarlanabilir:

DIM A(10, 10, 10)

Bu teknik, büyük bir grupdaki verileri daha küçük alt gruplara ayırmakta yararlıdır. Örneğimizde desteyi onüçlük dört iskambil rengine bölebiliriz. Bunlar o zaman şu şekli kullanarak ulaşılabilirler.

DIM DESTE (4, 13)

Şimdi karo dörtlüsünü bulmak istersek, bu ilk matrisimizde 43. eleman olabilirdi, oysa şimdi sadece (2,4) elemanını incelemeniz gerekiyor. (Karoların matrisimizde ikinci sırada olduğunu varsayarak)

Matrislerin yalnızca sayısal verileri depolamak için kullanılmaları gerekmez, karekter dizileri de kullanabilirler. Bunun bir uygulaması. Bir tiyatroya, ya da uçakda yer ayıran insanların isim kayıtları olabilir.

4.12 DATA komutu

Bu emir, *READ* emriyle bağlantılı olarak, programa veri girişi için kullanılabilir. Gereken veriler bir satıra, bu veriyi bir virgülle ayırarak ve bütün listenin başına *DATA* emri konularak sıralanabilir. Verilere artık *READ* emriyle daha sonra ulaşılabilir.

ÖRNEK:

```
10 READ X, Y, Z
20 PRINT X ; " + " ; Y ; " + " ; Z ; " = " ; X + Y + Z
30 DATA 1, 3, 5
```

Veriler sayısal ya da karekter dizisi veya ikisinin karışımı olabilir. Bütün verileriniz bir satıra sığmıyorsa üzülmeyin, başında *DATA* emriyle yeni bir satıra başlayın. Bilgisayar bir *READ* emriyle karşılaşınca programda, nerede olursa olsun takib eden veri parçasını arar. Bütün *READ*'leriniz için yeterli verileriniz olmasına dikkat edin, aksi halde hata mesajına yol açabilirsiniz.

Bu birbirini takib eden verileri bölmenin tek yolu *RESTORE* emrini kullanmaktır. Bu veri göstericisini tekrar programın başına alır ve aynı verinin gerekirse bir kaç kez okunmasına izin verir. Aşağıdaki program *DATA*, *READ* ve *RESTORE* emirlerini gösteriyor.

```
10 FOR C = 1 TO 4
20 READ A$
30 PRINT A$ ; " " ;
40 NEXT C
50 RESTORE
60 GOTO 10
70 DATA "BASIC", "DİLİ", "HOŞUNUZA", "GİTTİ Mİ"
```

Bu programı durdurmak için *[ESC]* tuşuna iki kez basın.
Dikkat edin ki *DATA* örnekde programın sonuna konulmuşsa da aslında müsait olan herhangi bir yere konulabilirdi.

DATA emri yalnızca okunduğu zaman yazılması gereken bilgileri içermesi için kullanılmaz ; bir *DATA* satırında sayısal değişkenler de, örneğin, *SOUND* emrinde okunabilir.

```
10 FOR n = 1 TO 30
20 READ S
30 SOUND 1,S,40,5
40 NEXT n
50 DATA 100, 90, 100, 110, 120, 110, 100, 0
60 DATA 130, 120, 110, 0, 120, 110, 100, 0
70 DATA 100, 90, 100, 110, 120, 110, 100, 0
80 DATA 130, 0, 100, 0, 120, 150
```

Bir şey duymuyorsanız bilgisayarın sağındaki düğmesini ayarlayın.

BASIC'de bu kısa girişi bitirirken, işte size CPC464 ile yirmibir oynama olanağı veren bir program. Program *BASIC*'in bir çok özelliğinin kullanımını gösteriyor ve değişken isimlerin uygun kullanılmasından ötürü zevkle anlaşılacaktır. Bu programı grafikle güzelleştirebiliriz, sesle heyecan katabilir ve genel olarak bütün *BASIC* programlarının sağladığı imkânla basit bir başlangıçtan en güzel şekle getirebilirsiniz.

```
  1 REM YİRMİBİR
 10 REM BAŞLAMA
 20 YC = 2 : CC = 2
 30 AS = 0
 40 FAS = 0
 50 S = 0
 60 T = 0
 70 DIM TAKIM $ (4)
 80   AKIM $ (1) = "SİNEK"
 90 TAKIM $ (2) = "KUPA"
100 TAKIM $ (3) = "MAÇA"
110 TAKIM $ (4) = "KARO"
120 CLS
130 DIM DESTE (52)
140 FOR X = 1 TO 52
150 DESTE (X) = 0
160 NEXT X
170 REM HER OYUNCUYA İKİ KART DAĞITMA
180 LOCATE 10,3
190 PRINT "SİZ"; SPC (15) "KASA"
200 LOCATE 3,5
210 GOSUB 740
220 S = S + F
230 IF F = 11 THEN AS = AS + 1
240 LOCATE 3,6
250 GOSUB 740
260 S = S + F
270 IF F = 11 THEN AS = AS + 1
280 LOCATE 24,5
290 GOSUB 740
300 T = T + F
310 IF F = 11 THEN FAS = FAS + 1
320 LOCATE 24,6
330 GOSUB 740
340 T = T + F
350 IF F = 11 THEN FAS = FAS + 1
360 REM .....
370 X$ = INKEY$ : IF XS <> "S" AND XS <> "T" THEN 370
380 IF X$ = "S" THEN 560
390 LOCATE 3, YC+5
400 YC = YC + 1
410 GOSUB 740
420 S = S + F
430 IF F = 11 THEN AS = AS + 1
```

```
440 REM
450 IF S<22 THEN 370
460 IF AS = 0 THEN 500
470 AS = AS - 1
480 S = S - 10
490 GOTO 450
500 LOCATE 12,19
510 PRINT "KAYBETTİNİZ"
520 PRINT : PRINT "YENİ OYUN (E/H)"
430 X$ = INKEY$ : IF <>"E" AND X$<>"H" THEN 350
540 IF X$ = "E" THEN RUN
550 END
560 IF T >16 THEN GOTO 700
570 CC = CC+1
580 LOCATE 24, CC+4
590 GOSUB 740
600 T = T + F
610 IF F = 11 THEN FAS = FAS - 1
620 IF T<21 THEN 560
630 IF FAS = 0 THEN 670
640 FAS = FAS - 1
650 T = T - 10
660 GOTO 620
670 LOCATE 12,19
680 PRINT "KAZANDINIZ"
690 GOTO 520
700 LOCATE 12,19
710 IF T < S THEN 680
720 PRINT "KASAYI KAZANDINIZ"
730 GOTO 520
740 REM KART DAĞITMA
750 LET KART = INT (RND (1)*52+1)
760 IF DESTE (KART) = 1 THEN GOTO 750
770 DESTE (KART) = 1
780 F = KART - 13* INT (KART/13)
790 IF F = 0 THEN F = 13
800 IF F = 1 OR F >10 THEN GOTO 850
810 PRINT F ; "KAPALI";
820 IF F > 10 THEN F = 10
830 PRINT TAKIM$ (INT ((KART-1)/13)+1
840 RETURN
850 IF F = 11 THEN PRINT "VALE";
860 IF F = 12 THEN PRINT "DAM";
870 IF F = 13 THEN PRINT "PAPAZ";
880 IF F<>1 THEN GOTO 820
890 F = 11
900 PRINT "AS"
910 GOTO 830
```

Oyunun amacı elinizdeki kartları toplıyarak suratları da ekliyerek 21'i geçmeden 21'e en

yakın sayıyı tutturmaktır. Kasa da sizin sayınızı tutturmaya ya da 21'e sizden daha yakın gelmeye çalışır. Birinci satırı yazdıktan sonra *AUTO* emrini satır numaralarını otomatik olarak çıkartmak için kullanılabilir.

Yeni kart için *T*, yatmak için *S* girin

Bu oyun CPC464 ile oynanabilecek iskambil oyunlarından birisi. Şimdilik bunu daha sonra *GRAFİK* ve *SES* ile besliyeceğiniz bir iskelet olarak kullandık.

4.13 MANTIK İFADELERİ

Bir hesap makinesiyle bir bilgisayarın en büyük farklarından biri, bilgisayarın şartlı *İF...THEN* zincirlemesini uygulamasında olduğu gibi, mantık ifadeleri kullanma yeteneğidir.

Mantık işlemcinin iki girdisi vardır.

< 1. girdi > [< Mantık işlemci > <2. girdi >]

Mantıksal işlemcide girdilerin ikisi de tam sayı olarak gösterilmelidir. Ve bir girdi tam sayı sınırları içinde değil ise işlem *Error 6* ile sonuçlanır.

Öncelik (önce gelme) sırasıyla ***Sayısal Operatörler*** şunlardır.
AND her iki girdinin ilgili biti 1 olmadıkça sonuç 0 dır.
OR her iki girdinin ilgili biti 0 olmadıkça sonuç 1 dir.
XOR her iki girdinin ilgili biti aynı olmadıkça sonuç 1 dir.
AND en çok kullanılan mantıksal işlemcidir.

PRINT 10 AND 10

10 yanıtıyla sonuçlanır.

PRINT 10 AND 12

Sonuç 8

PRINT 10 AND 1000

Sonuç yine 8

Bunun nedeni 10 ve 1000 sayılarının ikili tabana çevrilmeleridir.

1010 = 10 (onluk sistemde)
1111101000 = 1000 (ondalık sistem)

AND işlemi her biti tek tek kontrol eder, ve üstteki sıradaki bit ile alt sıradaki bit 1 olduğunda yanıt 1 dir:

0000001000

.....bu yeniden ondalık sisteme çevrildiğinde 8 olur. Bütün bunlar *AND* işlemcisinin iki satır ardarda mevcut olduğunu saptamak için kullanıldığı anlamına gelir.

İşte bize çok şey öğretecek bir uygulama:

```
10 INPUT "BUGÜN AYIN KAÇI" ; GÜN
20 INPUT "HANGİ AYDAYIZ" ; AY
30 İF GÜN = 1 AND AY = 1 THEN GOTO 50
40 GOTO 10
50 PRINT "YENİ YILINIZ KUTLU OLSUN"
```

OR'da bit'ler üstünde çalışır. Girdilerin ilgili her iki biti 0 olmadıkça sonuç 1'dir. *AND* örneğinde kullandığınız sayıları kullanarak

PRINT 1000 OR 10
1002
bit gösterici olarak
1010
1111101000
Şu sonucu alırız:
1111101010
Ve aşağıdaki program örneğinde:

```
10 CLS
20 INPUT "KAÇINCI AYDAYIZ" ; AY
30, IF AY = 12 OR AY = 2 GOTO 50
40 CLS : GOTO 10
50 PRINT "HAVALAR SOĞUK OLMALI"
```

```
10 CLS
20 INPUT "KAÇINCI AYDAYIZ" : AY
30 IF NOT (AY = OR AY = 7 OR AY = 8) GOTO 50
40 CLS : GOTO 10
50 PRINT DENİZE GİRİLMEZ"
```

Ele alınacak son önemli özellik de, bir koşulun içindeki daha başka koşullarıda (en fazla satır dolana kadar) bir kaç tane mantık ifadeyi biraraya getirebilme olayıdır.

```
10 CLS : INPUT "BUGÜN AYIN KAÇI" ; GÜN
20 INPUT "KAÇINCI AYDAYIZ" ; AY
30 IF NOT (AY = 12 OR AY = 1) AND GÜN = 29 GOTO 50
40 CLS : GOTO 10
50 PRINT "BU AY OCAK veya ARALIK değil"
55 PRINT "AMA ŞUBAT 29 çekebilir"
```

Bir mantık işleminin sonucu ya -1, ya da 0 dır. Bir mantık işleminin sonucu doğru için -1, yanlış için 0 olur.
Bunu, yukarıdaki programa 60. satırı ekliyerek kontrol edin.

```
60 PRINT NOT (AY = 12 OR AY = 1)
70 PRINT (AY = 12 OR AY = 1)
```

Ve program çalıştırıldığında gün için 29, ay içinde örneğin 5 girilerek, 50. satırdaki cevap meydana gelecektir ve 60. ve 70. satırlardaki mantık ifadelerden dönen esas değerler altta görüntülenecektir.

XOR, her iki girdinin ilgili bitleri farklı olduğu sürece doğru (1) sonuç meydana getirir. Aşağıdaki tablo mantık tablosu olarak bilinen bütün bu özellikleri özetler.

A nın ilgili biti B nin ilgili biti	1 0	0 1	1 1	0 0
AND işlemi	0	0	1	0
OR işlemi	1	1	1	0
XOR işlemi	1	1	0	0

5.GRAFİK ÖZELLİKLERİ

CPC464'ün renk ve grafik olanakları:

Bu bölümün içerdiği konular.

* **Ekran MOD'ları ve PIXEL'ler**
* **Renkler**
* **INK, PAPER ve PEN**
* **Çizgi çizmek**
* **WINDOW**

5.1 Cihaza özgü özellikler:

Şimdiye kadar yapılan tanımlamalar ve *AMSTRAD BASIC* uygulamaları daha çok Standart Basıc'in özelliklerine dayanıyordu. Sırf aritmetik operasyonlarının çoğu çok küçük ayarlamalarla *BASIC*'in bir türünden öbür türüne uygulanabilir. Ancak, *BASIC*'in grafik kontrol emirleri CPC 464'ün donanımının ekran görüntüsünü kontrol ediş biçimine çok daha belirgin bir şekilde sadıktır ve bilgisayarın en iyi biçimde yararlanmanız için çok dikkatle anlaşılması gerekir.

Bu komutlarla ilgili ileri örnekler, kısa ve özlü tanımı, kullanımı 8.bölüme bakarak bulunabilir.

5.1.1 Renk Seçenekleri:

'Siyah' aşağıdaki renklerin ve onlarla ilgili değişik emirleri tanımlayan, tanımlamaların amaçlarına hizmet etmek için kullanılan renk olarak kabul edilir.

BORDER, ekran *MOD*'u ne olursa olsun herhangi bir çift rengede ayarlanabilir ve *MOD* emri verildiğinde ilk konumuna dönmez. Değişen iki rengide ayarlanabilir. Birbiri ardından görüntülenebilen *INK* olanaklarının sayısı seçilen ekran *MOD*'una bağlıdır. Her *INK* yanan sönen iki renge ya da tek sabit bir renge ayarlanabilir. Daha önce de tanımlandığı gibi, herhangi bir zamanda kullanılabilir. *INK* sayısı ekran *MOD*'una bağlıdır.

Kullanılan *MOD*'a göre *PAPER* ve *PEN* komutlarında mümkün olan renkler seçilir.

5.1.2 SAYDAMLIK PEN, INK ve PAPER ilişkisi:

İki değişik rengin belirtildiği yanar söner koşul haricinde, ekrana yazı yazarken iki *INK* kullanılır: biri *PEN* rengini saptayan *INK*'dir, öteki de *PAPER* rengini saptar.

NOT:
PAPER emriyle ilgili girdi 0 sayısı için belirtilen *INK*'dir. (mürekkep rengi) ve EK VI da sıralanan renk numarası ***DEĞİLDİR.*** Aynı şekilde *PEN* emriyle ilgili girdi bu sayıyla belirtilen *INK*'dir. EK VI da verilen renk numarası değildir.

PAPER belirtilmediği zaman kendiliğinden 0 olur. *PEN* belirtilmediğinde 1 olur. *PAPER* No 0 için *INK*'i renk No 9 olan yeşile ayarlamak için şöyle yazarsınız.

INK 0, 9

Aynı şekilde *PEN* No için *INK*'i renk No 0 olan siyaha ayarlamak için, şunları yazın:

INK 1, 0

PAPER'ı *PEN* ile aynı *INK*'e ayarlamak bütün ekran görüntüsünü siler.

Yazılar, saydam olarak (kürsör herzaman saydama ayarlıdır) olarak ayarlanabilir. Bunun için başlıca *BASIC* grafik emirlerine yararlı ekler sağlayan bir seri karekter kontrolundan birini kulanırsınız. Saydam seçeneklerde kâğıt rengini yok sayıp grafiklerin üstüne yazabilirsiniz, ya da fonun üstüne yazabilirsiniz.

Bu kısa program eldeki olanakları gösteriyor:

```
10 MODE 1
20 INK 2,19
30 DRAW 200,200,2
40 LOCATE 1,21
50 PRİNT "1 NORMAL"
60 PRİNT CHRS (22)+CHRS (1)
70 ORİGİN 0,0
80 DRAW 500,200,2
90 LOCATE 12,18
100 PRİNT "2 SAYDAM"
110 PRİNT CHR$ (22)+CHR$(0)
120 LOCATE 22,15
130 PRİNT "3 NORMAL"
```

Satır 30 daki ilk *DRAW* emri saydam *MOD* ayarlanmadan önce yerine getirilmişti, ama ikincisi *CHR$ (22) + CHR$ (1)* emriyle 60.satırda saydam *MOD* verildikten sonra çizildi.

Üst üste gelen noktaların nasıl renk değiştirdiğine ve saydam *MOD*'da karekterin bulunduğu yerin rengindeki değişikliğe dikkat edin. Programda satır 60 daki saydamlığın başlama yeriyle satır 110 daki saydamlığın bitmesi emirlerinin yerlerini değiştirin ve bunun görüntüdeki etkisini görün. Bu ek emirlerin tamamının listesi EK VI da verilmiştir.

5.2 Ekran MOD'ları:

Ekranın (text ve grafik operasyonları) fonksiyonlarını yerine getirdiği üç *MOD* vardır.

a) Normal
MOD 1: 40 sütun x 25 satır, 4 *INK* 320 x 200 *Pixel* ve tek tek çağrılabilen 4 renk.

b) Çok renkli *MOD*
20 sütun x 25 satır, 16 *İNK*
160 x 200 *pixel* ve tek tek çağırılabilen 16 renk

c) yüksek rezolusyonlu *MOD*
80 sütun x 25 satır 2 *İNK*
640 x 200 *pixel* tek tek çağırılabilen 2 renk.

Gördüğünüz gibi farklılık görüntünün tek tek yatay elemanlarının sayısındadır. Bunlar TV tüpünün yüzündeki küçük eğrilerle karıştırılmamalıdır. Bu çizgiler TV monitör donanımının ayrı bir özelliğidir.

Bu üç değişik görüntünün her birine ekran ya da görüntü *MOD*'u terimi kullanılır. Ve *BASIC*'de bir seferde yalnız bir *MOD* kullanılabilir. *MOD* değiştirmek text ve grafik pencereler *(window)* dahil ekranı tamamiyle temizler, (*CLS* ve *CLG* emirleriyle aynı etkiyi yapar) ama, program hafızasının içeriğini etkilemez. *MOD* değişiklikleri *BASIC* programdan, ya da direkt emirler kullanılarak yapılır.

5.2.1 MOD 0 çok renkli grafik görüntüsüdür.

Eldeki 27 renkten 16 sı aynı anda görüntülenebilir,ve görüntünün her ayrı elemanı ayrı ayrı renk için programlanabilir. Görüntü, her yatay sıra için 160 ve her dikey sıra için 200 pixel'den oluşur. Bu şekilde bir ekran planı EK VI da yer alır.
MOD 0 da 25 satır 20 kolon vardır.

5.2.2. MOD 1 Standart MOD'dur.

MOD 1 CPC464 açıldığında önceden ayarlanmıştır. 27 renkten 4'ü aynı anda görüntülenebilir. Bütün 27 rengin birinden öbürüne hızla geçebilirsiniz.
Görüntü 320 pixel eninde ve 200 pixel boyundadır.
Bu şekilde bir ekran planı da EK VI da bulunur.
MOD 1 de 25 satırla 40 kolon vardır.

5.2.3 MOD 2 yüksek rezolüsyon MOD'udur.

MOD 2 aynı anda 2 renk kullanımına izin verir ve en çok, bir satırda 80 karekter kullanabilmesi yeteneğinden dolayı kullanılır.
Bir bakışda programın daha büyük bir kısmı görülebildiği için bu *MOD* program okumayı daha kolaylaştırır.
MOD 2 her yatay sırada 640 ve her dikey sütunda 200 pixel sağlar.

5.2.4. DENEME

CPC464 'u *[CTRL] [SHIFT]* ve *[ESC]* tuşlarıyla ilk konumuna getirdikten sonra bu programı yazın.

```
  5 REM GRAFİK PROGRAMI
 10 MODE 1
 15 İNK 2,0
 16 İNK 3,6
 17 BORDER 1 : REM KOYU MAVİ
 20 GLG  : REM GRAFİK EKRANI TEMİZLEME
 30 b% = RND*5+1 : REM RASTGELE SAYI
 40 c% = RND*5+1
 50 ORİGİN 320,200 : REM GRAFİK kürsör için ilk konum
 60 FOR a=0 TO 1000 STEP Pi/30
 70 X% = 100*cos(a)
 80 MONE X%, X%: REM grafik kürsörün hareketi
 90 DRAW 200* cos(a/b%), 200*SİN(a/c%),3: REM çizgi çizimi
 91 İF İNKEY$ <>" " THEN 20
100 NEXT: REM 60 a DONUS
100 GOTO 20
```

Şimdi programı çalıştırın. Başka bir örnek elde etmek için herhangi bir tuşa basın. Bu CPC 464 donanımının ve yazılımının önemli bir kaç özelliğini gösteriyor: CPC464 ekrana son

derece net yazar, yazılanı en az çaba ile son derece ince ve güç etkileri başarmayı sağlıyan emirleri içerir. *REM* komutu ise yalnızca işinizi kolaylaştırmak için bulunur ve bunları çalışılacak programa dahil etmeniz şart değildir. Yalnızca size, (özellikle başta programı kendisi yazmamış olanlara) ne olup bittiğini anlatmaya yardımcı olur.

Dikkat ederseniz satır numaraları birkaçı sonradan düşünülmüş girişleri belirtiyor ve eğer ilk numaralamayı bırakırsak, *RENUM* emri vererek bir yandan listeyi tertipler, bir yandan da programların yapılarından ne şekilde gelişip, büyüdüğünü izleyebilirsiniz.

Bu programı kasede kaydedin. Örneğin:

SAVE "GRAFİK 5.5.84"

Şimdi şu programı inceleyelim ve değişik grafik olanaklarını görelim.

```
10 a$ = İNKEY$:REM yeni bir şekil için herhangi bir tuşa basın
20 İF a$ = "" THEN 10
30 CLS
40 m = İNT (RND*3) : REM 0 ila 2 arasında rastgele bir sayı seçimi
60 MODE m
70 i1 = RND*26 : REM rastgele ink rengi seçimi
80 i2 = RND*26
90 İF ABS (i1-i2)>5 THEN 70
100 İNK 0, i1 : İNK 1,i2
110 S = RND*5+3
120 ORİGİN 320, -10
130 FOR X = 1000 TO 0 STEPS
140 MOVE 0,0
150 DRAW X,300 : DRAW 0,600
160 MOVE 0,0
170 DRAW -X,300 : DRAW 0,600
180 a$ = İNKEY$
190 İF a$ < > "" THEN 30 : HERHANGİ BİR TUŞA BASARAK CEVRİMDEN
    ÇIKIŞ
200 NEXT X
210 GOTO 10
```

Bu ve bir önceki program renk dolu ve görsel bir şekilde basit matematik kavramları gösteriyor. Bunların her ikisi de temelde her örneğin bir şekilde farklı olmasını sağlamak için rastgele sayılar üstüne toplamalar yapıyor ve sonuçları rastgele satır olarak sergiliyorlar.

Elinizdeki CPC 464 mükemmel bir grafik kâğıdıdır ve sinüs eğriside en klasik örneklerden birisidir.

```
10 REM SİNÜS EĞRİSİ ÇİZİMİ
20 MODE 2
30 İNK 1,21
40 İNK 0,0
50 CLS
60 DEG
70 ORİGİN 0,200
80 FOR N = 0 TO 720
90 Y = SİN (N)
```

```
100 PLOT n*640/720,198*y,1
110 NEXT
```

100. satırdaki *PLOT* komutu, programın eğriyi çizen kısmıdır. *FOR NEXT* çevriminde (satır 80-110) yapılan her hesap için ekranda bir pixel oluşturuluyor ve sonuç ekranınızda görüntüleniyor.

CPC 464'ün bir çok basit ve güçlü emri vardır. Yukardaki programın etkisine yalnızca şöyle yazarak katkıda bulunabilirsiniz.

```
15 BORDER 6,9
```

Tekrar çalıştırın, *BORDER* şimdi 6 ve 9 numaralı renkler arasında değişiyor. Yanıp sönme hızı ilk değerlerce ayarlanmıştır. *[ESC]* tuşuna basıncaya kadar sürekli programın çevrimlenmesi için şunları ekleyin:

```
120 GOTO 50
```

Program durduğunda yanıp sönen border durmadı. Çünkü, *BORDER*'in kontrolü programınkinden ayrıdır. *BORDER*'in yanıp sönmesini durdurmak ve parlak maviye ayarlamak için *[ESC]* tuşuna iki kez ve 15. satırı şöyle değiştirin.

```
15 BORDER 2
```

Programı tekrar çalıştırın, yanıp sönme durur.
Eğrinin ve fonun rengini değiştirmek için 30. ve 40. satırlardaki INK'in rengini değiştirmeniz gerekir.
Programın listesini istediğinizde şöyle görünmesi gerekir.

```
 10 REM SİNÜS EĞRİSİ ÇİZİMİ
 15 BORDER 2
 20 MODE 2
 30 İNK 1,2
 40 İNK 0,20
 50 CLS
 60 DEG
 70 ORİGİN 0,200
 80 FOR N = 0 TO 720
 90 Y = SİN (N)
100 PLOT n*640/720,198* y,1
110 NEXT
120 GOTO 50
```

100. satırda *PLOT* bildiriminin sonundaki 1 sayısı bilgisayara eğriyi 30. satırda *INK 1* emriyle belirlenen renkte çizmesini söyler.

8. bölümdeki komut listesinde *PLOT* komutunun tanımlamasına bakın ve bu bildirimin değişik bölümlerinin tam olarak normal işlediğini görün.

Eğrinin ekranda çizilişine dikkatle bakarsanız tek bir çizgi olmayıp bir çok kırık küçük ince parçalardan oluştuğunu görürsünüz. En küçük tek parça daha önce de tanımlanan pixel'dir.

5.2.5. Grafik kürsörü ve çizgi çizmek:

Programları grafik görüntülere çevirmenin bir takım yollarını deneyiniz ve program emirlerinden ve kavramlarından birkaçına gösteri yapma şansı verildi. Ekrana çizgi çizerken karışıklığı önlemek için dikkat edilecek birkaç nokta vardır.

Dikkat edilecek ilk nokta hafızadaki programın son şeklidir. Program hafızasını temizlemek için verilen *NEW* komutundan sonra bile bilgisayar renkleri hatırlar. Her şeyi ilk konumuna getirmek için aynı anda *[CTRL] [CTRL]* ve *[ESC]* tuşlarını kullanmanız gerekir.

(Bunu yapmadan önce gerek duyduğunuz şeyi kaydedin) Bunu katınlamak için şunları yazalım.

NEW : CLS

Bir önceki programdan çıktıktan sonra şunları yazın.

DRAW 100, 100

DRAW komutu *GRAFİK* kürsörünün son yerinde belirtilen (100, 100) x,y koordinatına düz bir çizgi çizer. *GRAFİK* kürsörü, bir sonraki grafik işleminin olacağı noktayı belirten, görünmeyen bir kavramdır.Yerini bulmak için *XPOS* ve *YPOS* fonksiyonlarını kullanmanız gerekir.

PRINT XPOS yazdığımızda

Yanıt şöyledir:

100
(bu da şu anda *YPOS* içinde aynıdır.)

Dikkat edin, metin ekranın dibine gidip görüntünün yukarı gelmesine sebep olursa grafik görüntü de yukarı kayar ama grafik kürsörün koruma eskisi gibi kalır. (↓) kürsör tuşuna ekran yukardan temizleninceye kadar basın ve tekrar *XPOS* ve *YPOS* isteyin. Grafik kürsörünün değeri hala aynı olarak hafızadadır.

Çizilen çizgiyi renk belirtmek için *DRAW* emrinin sonuna renkle ilgili sayıyı ekleyin (bir önceki sayfadadır. Programdan sonraki *PLOT* emrinin tanımlanmasına bakın)
Önce *İNK*'i belirtmiş olmanız gerekir.Ve unutmayın ki *INK* yalnızca kullandığınız ekran *MOD*'unda izin verilen renk ve *INK* numaralarını kullanabilirsiniz.

Bunu görmek için aşağıdaki kısa programı yazınız.

```
10 MODE 1
20 İNK 0,10
30 ORİGİN 0,0
40 İNK 1,20
50 İNK 2,0
60 DRAW 320,400,1
70 DRAW 640,0,2
```

İşte size şimdiye kadar sözü edilen birimleri kullanan ve birkaç tane de daha yararlı kavram tanıtan bir program örneği. Beklenen sonuçları alabilmek ve CPC 464'ün hafızasında her ne varsa yeniden ayarlamak için ilk satırın (satır 10) nasıl renk ve *INK* koşullarını ayarladığına dikkat edin.

```
10 İNK 0,0:İNK 1,26:İNK 2,6:İNK 3,18: BORDER 0
20 REM GRAFİK
30 mode 1:DEG
40 PRINT "3,4 veya 6 yüzlü?";
50 LINE INPUT p$
60 IF p$="3" THEN sa=120:GOTO 100
70 IF p$="4" THEN sa=135:GOTO 100
```

```
 80 IF p$="6" THEN sa=150:GOTO 100
 90 GOTO 50
100 PRINT:PRINT "Hesaplyorum";
105 IF p$="3" THEN ORIGIN 0,-50,0,640,0,400
    ELSE ORIGIN 0,0,0,640,0,400
110 DIM cx(5), cy(5), r(5), lc(5)
120 DIM np(5)
130 DIM px%(5,81),py%(5,81)
140 st=1
150 cx(1)=320:cy(1)=200:r(1)=80
160 FOR st=1 TO 4
170 r(st+1)=r(st)/2
180 NEXT st
190 FOR st=1 TO 5
200 lc(st)=0:np(st)=0
210 np(st)=np(st)+1
220 px%(st,np(st))=r(st)*SIN(lc(st))
230 py%(st,np(st))=r(st)*COS (lc(st))
240 lc(st)=lc(st)+360/r(st)
245 IF (lc(st) MOD 60)=0 THEN PRINT".";
250 IF lc(st) < 360 THEN 210
252 px%(st,np(st)+1)=px%(st,1)
254 py%(st,np(st)+1)=py%(st,1)
260 NEXT st
265 CLS:ink 1,2
270 st=1
280 GOSUB 340
290 LOCATE 1,1
300 EVERY 25,1 GOSUB 510
310 EVERY 15,2 GOSUB 550
320 EVERY 5,3 GOSUB 590
330 GOTO 330
340 REM çember çizme
350 cx%=cx(st):cy%=cy(st):lc(st)=0
360 FOR x%=1 TO np(st)
370 MOVE cx%,cy%
380 DRAW cx%+px%(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)*COS(sa+lc(st))
450 st=st+1
460 GOSUB 340
470 st=st-1
480 lc(st)=lc(st)+2*sa
490 IF (lc(st) MOD 360)<>0 THEN 430
500 RETURN
510 ik(1)=1+RND*25
```

```
520 IF ik(1)=ik(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(?)=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
```

Bu programı çalıştırdığınızda size bir soru soracaktır (satır 40) en hızlı sonuç için 3 olarak yanıtlayın.

Program daha sonra kendi kendine düşündüğü ve programın çalışıyor olduğunu belirtmek için bir kaç saniyede bir Hesaplıyorum mesajını ve bir nokta görüntüleyecektir. (satır 245) 300-320 satırlarda çağırılan alt programlar, *EVERY* emriyle saptanan hızda değişik renk *INK* yanıp sönüyor. Bu yanıp sönmeyi yavaşlatmak için 300-320 satırları şöyle düzeltin.

```
300 EVERY 250, 1 GOSUB 510
310 EVERY 150, 2 GOSUB 550
320 EVERY 50, 3 GOSUB 590
```

Ne yaptığınızı görmek için 8. Bölümde *EVERY* emrinin anlamına bakın. *AMSTRAD BASIC*in en yararlı özelliklerinden biridir. *EVERY* emrinin ilginç bir etkisi de, program *[ESC]* tuşuna bir kez basılarak bölündüğünde, bir şey yapmak için istemleri biriktirmesidir. *[ESC]* tuşuna bir kez basarak işlemleri biraz duraklatın ve herhangi bir tuşa basarak yeniden başlatın. Görüntü çılgın bir hızla yanıp sönecektir. Çünkü, kuyrukta bekleyen zamanlama komutları yetişmek için acele eder. Kuyrukta yalnızca belirli miktarda yer vardır. Bu yüzden bir süre sonra, kuyruktakiler işlemini yapıp yer açıncaya kadar *EVERY* emri çıkartılır.

5.3 WINDOW (pencere) özelliği

Kullanıcı metnin yazılacağı ve grafik çiziminin yapılacağı pencereyi 8 adede kadar seçebilir.
Ekran *MOD*'u seçildiğinde pencerelerde kendiliğinden ilk değerlerine yeniden ayarlanır.

NOT:
Bir sonraki pencere *(window)* tüm ekrana eşitse, o zaman hızlı bir geçiş bilgisayar tarafından yapılır. Bir sonraki pencere *(window)* eldeki ekrandan küçükse geçiş program tarafından yapılır. Bu da daha yavaştır.

Window emri belirlenmiş ekran kanalından sağ-sol, alt-üst karekter hücrelerini belirler. Pencereler birbirlerinin üstüne binebilir ve içi dolu kutuları hızlı çizme imkânı sağlar. Bunları keşfetmeden önce şunları yazın:

```
KEY 139, "Mode2: paper 0 : ink 1,0 : ink 0 , 9: List " + CHR$ (13)
```

Bu, nümerik klavyede *[ENTER]* tuşunu *PEN* ve *PAPER*'in bazı görünmeyen renk karışımlarında yolunuzu kaybettiğiniz durumlarda, görünen renkleri geri getirmeyi ayarlar. Aşağıdaki programı ekrana bir seri pencere çizer ve iki önemli nokta gösterir.

```
 5 MODE 0
10 FOR N = 0 TO 7
20 WINDOW#n,n+1, n+6, n+1, n+6
30 PAPER#4, n+4
40 CLS#N
50 FOR C=1 TO 200 : NEXT: NEXT
```

İlk nokta yeni ekranın bir öncekinin üstüne oluşmasıdır. İkincisi de, kanal 0 da bu zaman belirginleştirme mesajının çıkmasıdır. Başka bir şey yapmadan önce, programın listesini isteyin.

Ve program kanal 0'ın içinde sıkışacak. Aşağıdakini deneyin:

LIST # 5
Sonra da:
CLS # 6

.... En son çağrılan ekran başka bir pencerenin üstüne yazılır ve liste kanal 5'ten gönderilse de ***Ready*** mesajı kanal 0 da görünür.

WINDOW SWAP emrini kullanarak 55. satırı ekleyin:

55 IF n= 3 THEN WINDOW SWAP 7,0

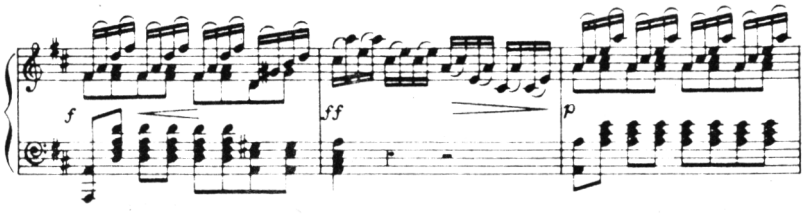
Bunun ***Ready*** mesajını program yerine getirme sırasında kanal 7'nin sonuna yönlendireceğini düşünebilirsiniz. Çalıştırın ve görün. Bu basit programı geliştirerek *WINDOW*'ların nasıl işlediğini ve birbirlerini nasıl etkilediğini iyi anlarsınız.

6. SES KOMUTLARINA GİRİŞ:

Ses bilgisayarın kendi içindeki oparlörden kaynaklanır. Ml 1 modülatör/güç kaynağı ve televizyon kullanıyorsanız TV'nin ses kontrolünü en aza indirin.

Sesin seviyesi bilgisayarın sağ tarafındaki ***VOLUME*** kontrolu kullanılarak ayarlanabilir. Ses aynı zamanda, bilgisayarın arka yüzündeki (1/0) kapısı kullanılarak stereo setinizin input girişinden verilebilir. Bu bilgisayardan çıkan sesleri stereo olarak hi-fi hoparlörden ya da kulaklıklardan dinlemenizi sağlar.

CPC 464'ün sesi görüntüleri kadar güzeldir. [Sesi işlemden geçiren yaratıcı program yapısından ***(software)*** en fazla yararlanabilmek için zamanlama (azami) yapılarının arkasındaki felsefeyi anlamanız gerekir.]



Bu bölümün içerdiği konular:

* Ton periodları
* Sound komutu
* Envelop yapabilme
* Sıralar ve senkronizasyon

Tek istediğiniz bilgisayara biip sesi çıkarttırmaksa:

PRINT CHR $ (7) yazmak kafidir.

Ve daha ileri gitmezseniz CPC 464'ün en enteresan özelliklerinden bazılarını kavramamış olursunuz. Bu bölümde konunun genel oluşumunu veren bir giriş ve komutlar üzerine ayrıntılı bir bakış ve onları belirleme yolları izler. Programın yapısına nota gamını nasıl konacağının temelini, değişik tip müzik aleti yaratma yollarını ve bunları kullanarak ezgi oluşturma bilgilerini verir.

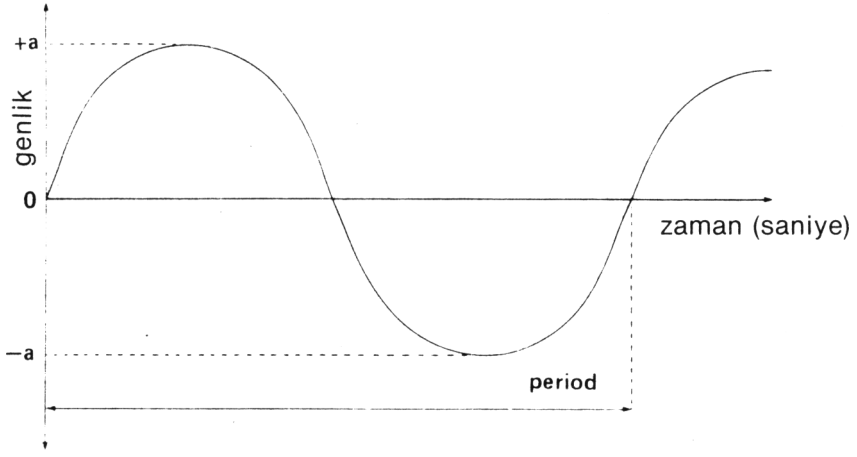
6.1 SES'in esasları.

Standart bir ezginin notasından çıkan bir sesi duyduğunuzda üstünde durulacak bir kaç özellik vardır.

1- Perde ve nota süresinde perde değişiklikleri.
2- Ses düzeyi ve nota süresinde bu düzeyde değişiklikler.
3- Notanın uzunluğu.

6.2 PERDE

Bir müzik notasında en önemli özellik perdedir. Bir müzik notası titreşim olarak tanımlanabilir. Bütün titreşimlerin frekansı, periodu ve genliği vardır. (bkz. Resim 1)



Resim 1 : Bir müzik notasının özellikleri.

Frekans bir saniyedeki titreşim sayısıdır. Period ise bu titreşimin süresidir. Genlik ses düzeyinin bir özelliğidir ve burada notanın perdesiyle bir ilgisi yoktur.

Frekans ve period birbirine basit bir formülle bağlıdır.
Frekans *(Hertz*'le ifade edilir) = 1/Period *(saniye* ile ifade edilir)
Ses *(SOUND)* emrindeki frekans ve ton periodu arasındaki ilişki:
Frekans 1000 lik bir ton periodu, 125 Hzlık bir ses ile ve 125 lik bir ton periodu, 1000 Hz. lık bir notayla sonuçlanır.

Bunlardan herhangi birinden perdeyi ayarlamak için yararlanılabilir. Ton periodunun değerinin çıkmasıyla perdenin düşmesi olayı sizi şaşırtmasın. Perde ***sound*** komutunun tanımlamasında (ton periodu) olarak görünür. Eldeki Ton periodları sıralaması geniştir.. (0.....4095) ve bir tam sayı olarak ifade edilmelidir. Bu, müzik notalarından bir gamı kurarken bazı yaklaşıklardan dolayı yanlışlara yolaçabilir. Ama, fazlasıyla hassas bir kulaktan başkasını rahatsız etmez. (EK VII ye bakın)

Bir nota gerçek bir müzik aletinde çalındığında perdesi değişebilir, bazen özellikle bir nota ile değiştirilir. ***ENT (Envelope Tone)*** emriyle, bir nota süresindeki period değişikliği yapısı için form çizebiliriz ve bu *SOUND* emriyle istenebilir. Bu özellik ***(tone envelope)*** olarak bilinir.

6.3. Ses Düzeyi:

Bu sesin yüksekliğinin ölçüsüdür. CPC 464'de bir sesin *(SOUND)* ilk seviyesini ayarla-

mak için doğrudan bir derecelendirme vardır. Ses yüksekliği bir tam sayıyla ses seviyesindeki yükseklik 0.....15 arasında ayarlanır.

Bu değer *SOUND* emriyle de ayarlanabilir.

Bazı basit noktalar denedinizse, işlemden geçmemiş, sıradan bir notanın nisbeten olağan bir şekilde tekrarlandığını farketmişsinizdir. Bunun nedeni, geleneksel müzik aletlerinde bir nota çalınırken, bu notanın başında bir ses artışı - buna atak denilir. Ve her notanın sonunda bir ses azalması vardır, buna düşüş denilir.

Her müzik aletinin değişik tip atak ve düşüşü bilgisayarla taklit edilebilir. Bunun için kullanacağınız komut *ENV* olacaktır.

6.4 Notanın Uzunluğu:

Notanın uzunluğu herhangi bir müzik oluşumunun en temel özelliğidir. Müzik notasının birimi çeyrek notadır ve yarım notalar (seri 1/16 lık notalar) vs. de vardır. Bunlar temel ünitenin çarpımları veya kesirleridir. Ancak ezgiler değişen hızda çalınabilir. Ve standart çeyrek nota için temel zaman uzunluğu veya periodu saptanmalıdır.

Bunu yapmak için *SOUND* (Ses) emrinde *(pine)* olarak bilinen bir ifade vardır. Bu da, 1 birim = 1/100 lük saniye (veya 100 = 1 saniye) olduğu bir tam sayıdır.

Dikkat edin: Bu ses düzeyi envelop kullanıldığında karışıklığı önleyen envelop değeriyle de saptanabilir.

6.5 Başka ses yaratma:

Rastgele gürültü (beyaz gürültü) bilgisayardan gelen ama müzikal nota olmayan sesdir. Bu ezgilerin fonuna ilginç varyasyonlar yaratmak için eklenebilir, ya da yalnızca kendi özel etkileri için kullanılabilir. Örneğin patlama temelde ses düzeyi (envelopu) kontroluyla yapılmış, beyaz gürültüdür. Bu gürültü, *SOUND* emri parametresi, gürültü periodu ile ayarlanan bir period etrafından frekansı rastgele değiştirir. Önemli bir özellik de, 3 ton period ayarlayabilen üç ayrı kanal olanağı olmasına karşın yalnızca bir gürültü period'un gereken kanal kombinasyonunda görünebilmek üzere ayarlanabilmesidir.

6.6 Birden fazla Ses ve Kanal:

Çoğu müzik parçaları en az iki anahtarla yazılmıştır, bas ve tiz. Bu yaklaşımı CPC 464'de mümkün kılmak için üç ses kanalı sağlanmıştır, A, B, ve C. Bunların hepsi tek başına çalabilir ya da gerektiğinde çalışmak üzere zamanlanabilir. Kanal seçimi *SOUND* emri parometresi kanal durumu, ile yapılır.

6.7 Kanal Sıraları:

Her ses kanalının bir ses sırası vardır. Bu sırada beş değişik *SOUND* emri için yer vardır: biri aktifdir ve 4'ü bekler. CPC 464'ün işletim istemi bir ses çalarkan başka görevleri de sürdürür ve sadece gerektiğinde daha başka *SOUND* emirleri almak için programa döner.

6.8. Kanal Durumu:

SQ istediğiniz kanalın durumunu saptamada kullanılır ve sıradaki boş yerler hakkında bilgiyle döner. *ON SQ GOSUB*, bilgisayarın dikkatini programın ses oluşturma bölümüne çevirmek için kullanılan bir sıraya sürme emridir.

Randevular (Ses kanalları için)

6.9 RANDEVU

Kanalları senkron yapmaya zorlamak için bir *"RANDEVU"* olanağı vardır. Bu, *SOUND* emirlerini aynı an la harekete geçiren iki ya da, daha fazla kanala bir işaretleyici konulmasıdır. Bunlar, her iki kanalda aynı anda meydana gelmeyen kesiklik ve başa dönüşleri içeren bir ezginin sürekli notalarının zamanlaması ve başa dönüş işlemi için çok yararlıdır.

DURDURMA:
[Ses *(SOUND)* kanal durumuna bakın.] Kanalların genel senkronun bir gecikmeyle yapılması gereken durumlarda ve sıraların hazır tutulması gereğinde önemlidir. Böyle durumlarda *DURDURMA* ve *BIRAKMA* sırasıyla yapılır.

6.10 Ses oluşturma Sıralaması:

SOUND G, H, I, J, K, L, M
G: Kanal durumu
H: Tone periodu
I: Süre
J : Ses düzeyi
K: Ses düzeyi envelopu
L : Tone Envelopu
M: Gürültü Periodu.

SOUND, Parametreleri *G*'den *M*'ye kadar hepsi tamsayı olan ve bunların yalnız ilk ikisi kesin gerekli olan bir emirdir.. Geri kalan parometreler isteğe bağlıdır, ama ilk değerleri vardır. Bunları parametre tanımlamasında inceliyeceğiz.

6.10.1 Parametre tanımlamaları:

G- Kanal Durumu.
Değer sıralaması 1.....255
Bir değer olmazsa hata verir.

CPC 464'de bir anda üç taneye kadar değişik ses çalmak olasıdır. Bu üç ses kanal ile mümkün olur, daha önce A, B ve C olarak söz etmiştik.
Girişte bir tamsayı verilir. Bu tamsayı bir byte'a karşılık düşmektedir ve anlamı aşağıdaki gibidir.

On tabanlı	Bit	Açıklama
1	0	sesi kanal A ya gönder
2	1	sesi kanal B ye gönder
4	2	sesi kanal C ye gönder
8	3	kanal A ile randevu
16	4	kanal B ile randevu
32	5	kanal C ile randevu
64	6	tut
128	7	bırak

Giriş Örnekleri:

2 = bundan sonra gelen *SES:* çıkış kanalı, B'ye gönder.
5 = bundan sonra gelen *SES:* çıkış kanalı A ve C'ye gönder.
98 = 64 + 32 + 2

= bundan sonra gelen *SES*'i çıkış kanalı B'ye gönder. Kanal C ile randevu ve hold *(tut)*. Unutmayın ki, iki (ya da daha fazla) kanal arasında randevu ayarlanacaksa bu kanalın c kanal durumuna uygun zamanda işaret vermek gereklidir.

Herhangi bir kanal kombinasyonunda ***hold*** (tutmak), ***RELEASE*** (bırakmak) emriyle serbest bırakılıncaya kadar (ya da daha sonraki bir ***SOUND*** emriyle koyuverilinceye kadar), bu emrin işlemden geçmesini dondurur. Bir kanal hareket biti 1 yapıldığında, geçirildiğinde SES *(SOUND)* hemen yerine getirilir, bu da sırayı boş ve kanalın başını karektersiz bırakır. Son duyulan seslerin hepsi susar.

H.Tone Period

Değer sıralaması 0.....4095
Çıkartılırsa, değer almazsa hata verir.
Tone period çalınacak *SES*'in frekansını ayarlayan periodu girer.
(Notanın perdesi) Frekans bu formülden değerlendirilebilir.
Frekans = 125000 / period
Ø kullanılarak, frekansız ayar yapılır, bu daha çok, yalnızca bir gürültü gerektiğinde yararlıdır.

I Süre

-32768.....+ 32767 arasındaki sıralamada herhangi bir değer.
İlk değer : 20
Sıfırdan büyük değerler için girilen değer saniyenin 1/100'ünü temsil eder. Sıfıra eşit olduğunda süre, belirtilen ses düzeyi envelopu'nun boyu tarafından idare edilir.
Süre sıfırdan az olduğu zaman, bu sayının pozitif değeri, belirtilmiş olan ses düzeyi envelopunun kaç kez tekrarlanması gerektiğini verir.

J. Ses Düzeyi

0.....15 arasındaki sıralamada herhangi bir tam sayı ses düzeyi envelopu belirtilmediği zaman 0.....7
İlk değer: 12 ses düzeyi envelope belirtilmediği zaman 4
Bu başlangıç ses düzeyidir. Ses düzeyi envelopu belirtilmişse, onunla değiştirilebilir. Sıfır sesin kapalı olması anlamına gelir.

K.Ses Düzeyi: Envelopu:

0.....15 arasında bir tam sayı olmalıdır. İlk değer: Ø
Kullanılan değer önceden tanımlanmış bir envelope belirtir. ***ENVELOPE'u*** tanımlamak için ***ENV*** emrini kullanın.
Eğer ses düzeyi envelop numarasını Ø yaparsanız, bu artık kalıcı olur ve ***ENV.*** komutu ile değiştirilemez ve ses düzeyi seviyesinde 2 saniyeye ayarlanır.

L.Tone Envelope:

0.....15 arasında bir tam sayı olmalıdır.
İlk değer: Ø dır.

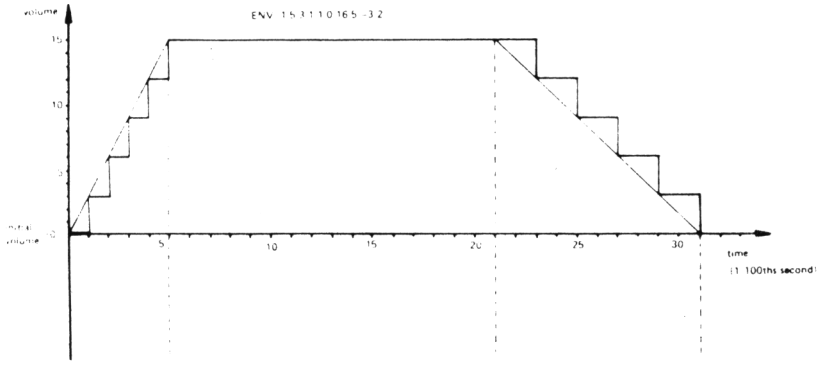
Kullanılan değer, önceden tanımlanmış bir ***envelope***'u belirtir. Bir ***envelop***'u tanımlamak için ***ENT*** emrini kullanın. Kalıcı bir tanımlama ton "***envelop*** sayısı" 0 dır, bu ***ENT***emriyle değiştirilemez ve ton sabit bir şekilde ayarlıdır.

M. Gürültü Periodu:

0.....15 arası bir tam sayı.
İlk değer ◊ dır.
Sese eklenecek gürültü efektini belirtir. İlk değer veya sıfır kullanılırsa hiç gürültü eklenmez. Unutmayın ki bir seferde yalnız bir gürültü periodu ayarlanabilir. Bu demektir ki gürültü için belirtilen bütün kanallar aynı gürültüyü alır.

6.11 ENV emri ve ses düzeyi envelopu

Atak ve düşüş bir noktaya boyut kazandırmak ve ona canlılık vermek için gereklidir. Bu emir oluşturulacak notanın biçimini tanımlar. Bilgisayara bu biçimi doğal sayısal komutlarla vermeye kalkışmadan önce gereksinmelerinizin taslağını bir kağıda yapmanız iyi olur. Aşağıdaki örneğe bakın.



Resim 3. Ses düzeyi envelope örneği w

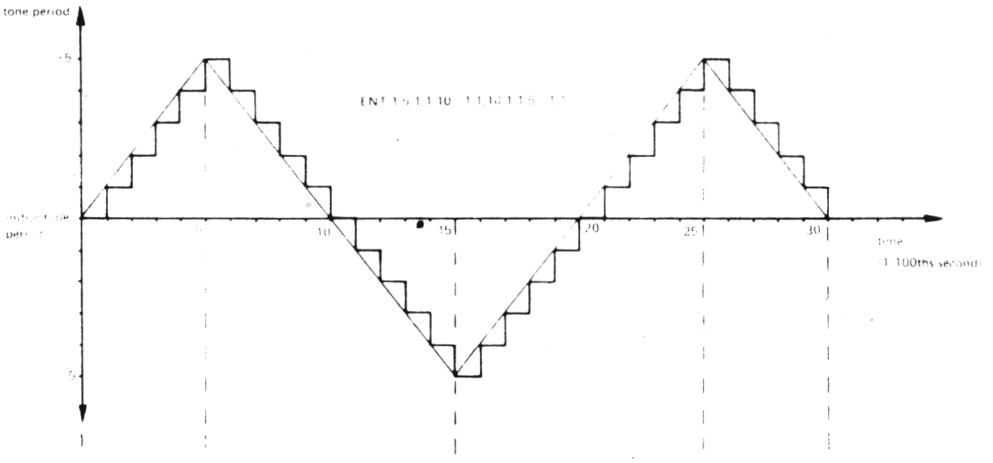
Bu komutun formüle edilebilmesi için biçimin sayılarla planlanması gerekir. Bunu yapmak için biçimi alt bölümlere ayırın. Beş bölüme kadar yapabilirsiniz. Ancak her bölümde biçim bir çizgi olmalıdır. Şimdi bu bölümü basamaklara ayırın, seçtiğiniz basamak sayısı, basamak sayımı, olarak bilinecektir ve bu basamaklara duraklama zamanı, denilen bir zaman süresi verilecektir. (1 = 1/100 saniye)

Basamakların binde, basamak boyu olarak bilinen ses çıkış ve inişleri vardır. Bir bölümde hiç basamak belirtilmezse yalnızca bir sonraki bölüme taşınan bir ses ayarı vardır. Bir müzik aletini taklit etmeye girişildiğinde çoğu kez bu notanın başlangıç ve bitiş ses düzeyi sıfır olacaktır. Bu yüzden ***SOUND*** emrinde ses düzeyini sıfır yapmalı ve bütün ses düzeyleri envelop ile ayarlanmalıdır.

Emrin Şekli:
ENV, N, Pl, Ql, R1, P2, Q2, R2, P3, Q3, R3, P4, Q4, R4, P5, Q5, R5
N: Envelope numarası (sayısı)
Pl....5. Basamak sayısı (1....5)
Ql....5. Basamak boyu (1....5)
Rl....5. Duraklama süresi (1.....5)
Envelope numarası mecburidir. Bütün bir bölüm zorunludur, ama bölümlerin sayısı isteğe bağlıdır. Örneğin 4 ve 5 bölümleri çıkartmak yalnızca 3 bölüm varsaydırır.

G. 12 ENT emri ve tone envelopu

Bunların kuruluşları ses düzeyi enveloplarınkiyle aynıdır, yalnız bu özellik notanın frekansında küçük değişiklikler yaratır, yani, bir çeşit vibrato. Bir notanın frekansının biçimine karar verdikten sonra daha önce yaptığınız gibi taslağını çıkartın ve bölümlere, basamaklara ayırın. Aşağıdaki örneğe bakın:



Resim 4. bir tone envelopu'nun karekteristikleri ses düzeyi envelope'ları ile Tone envelope'larının, daha önce ***SOUND*** emrinde ya da bir volume envelope'da belirlenmiş bir notanın süresi üstünde etkisinin olmayışıdır. Tone envelope notadan önce biterse ***SOUND*** (ses) emrinde (çağrılan) sabit tone period, ayrılan zamanı doldurur. Ancak eğer notanın perdesini aşarsa o zaman, geri kalan basamakların bölümleri terk edilir.. Bir tone envelope sayısı kullanın ***(SOUND)*** emrinde negatif olarak çağırılamaz.

Emrin biçimi:

ENT S, Tl, Vl, Wl, T2, V2, W2, T3, V3, W3, T4, V4, W4, T5, V5, W5
S: Envelope sayısı değer sıralaması 1.....5 (tekrar için-)
Tl.....5: Basamak sayımı (böl.1....5) değer sıralaması 0...239
Vl.....5: Basamak Boyu (böl.1...5) değer sıralamı 128 + 127
Wl.....5: Duraklamak zamanı (böl...5) değer sıralaması 0.....255(1/100 saniye)
değer sıralaması zorunludur, bir bölümün tamamlanması da zorunludur.

Bir envelope sayısı tanımlandıktan sonra daha önce yapılan ayarlanmalar geçersizdir. Envelope'lardan birini ya da ötekini bölüme vermeden belirtmek o envelope sayı ayarını sıfıra getirir.

6.13 İlgili başka fonksiyon ve emirler:

SQ (X)

X kanal numarası (1,2 veya 4)
Bu değerler sırasıyla A,B ve C kanallarını temsil ederler. (Ses emrinin bir parametresi olan kanal durumunun aşağıdaki tabloda gösterildiği gibi)
Bu aday gösterilen ses kanalının durumunun bir soruşturmasıdır.
Bu, 0.....255 arasında bir tam sayı olan cevap ile geri dönen bir fonksiyondur. Bu sayı-

dan bilgi çıkartmak için aşağıdaki tabloyu inceleyelim.
(bit) 0.....2 (bakılan) sıradaki boş yer sayısı (değer 0....4)
(bit) 3 başta görünen Kanal A ile randevu
(bit) 4 başta görünen Kanal B ile randevu
(bit) 5 başta görünen Kanal C ile randevu
(bit) 6 sıra başında durdurma
(bit) 7 Halen çalan kanal
3 den 6 ya kadar olan bitler.

Kanal durumu bölümünde tarif edilmiştir. Bu testin bir başka olanağı da, bundan sonraki paragrafta anlatılan ***ON SQ GOSUB*** emrini etkisiz hale getirmektir. ***(disable)***

ON SQ GOSUB

ON SQ (y) GOSUB (satır numarası)

Bu bir interrupt emridir. Aday gösterilen ***(SOUND)*** Ses sırasında bir ara olduğu zaman bunu bulur ve ***(SOUND)*** ses alt programına gönderir. ***SQ*** ve ***SOUND*** komutlarının ikiside interruptı durdurur. Alt program bir ***RETURN***'la normal şekilde biter. Bütün kanalların aynı önceliği vardır ve aday kanalda boş yer olduğunda emir devreye girer, böylece aynı zamanda silahsızlandırır. Böylece alt programın ***interrupt*** tekrâr gerekmesi durumunda yeniden kurması gerekir.

RELEASE

RELEASE (Bırakma)

RELEASE z

Z kanal numaralarını belirler, 1....7 arasında bir tamsayıdır.

SOUND emri parametresi kanal durumunda anlatıldığı gibi, bir kanaldır, işaret vermek ve durdurma özel bir ***SOUND*** emriyle mümkündür. ***RELEASE*** emri bu durdurmaları kaldırmak içindir.

Kanal girişi bitler seviyesindedir. Ve çözülünce serbest kullanılacak kanal kombinasyonları verir.

Durdurulma altında olmayan bir kanalın bırakılmasının etkisi yoktur.

Bit 0 : Kanal A
Bit 1 : Kanal B
Bit 2 : Kanal C

7 YAZICILAR VE OYUN ÇUBUĞU

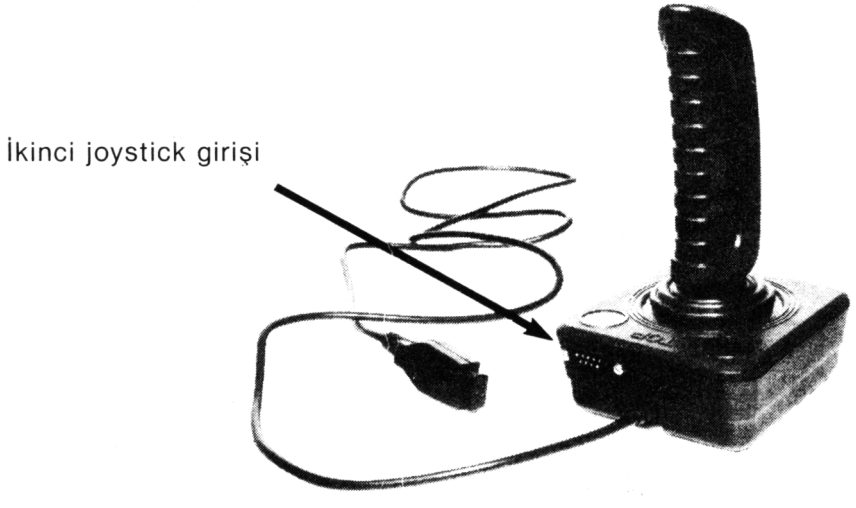
CPC 464'ün oyun çubuğu kontrolu ve ***Centronics*** çıkışlı yazıcı için ek arabirime ihtiyacı yoktur.

Bu bölümün içerdiği konular:

* Oyun çubukları
* Paralel çıkışlı yazıcılar.
* Arabirimleri

CPC 464 bilgisayarını oyun çubuğu olanağı içeren oyunlar için kullanıyorsanız, o halde ***AMSOFT*** oyun çubuğu modeli ***J41*** almak isteyebileceğiniz ek bir parçadır. ***J41*** bilgisayarınızın arkasında ***USER PORTS*** (1/0) yazan 9. yollu giriş fişine takılabilir. Amstrad CPC 464 bilgisayarı iki oyun çubuğu ile kullanılabilir. İkinci oyun çubuğu birinci oyun çubuğunun üstündeki giriş fişine takılmalıdır.

Oyun çubuğunu CPC 464'e bağlarken düşünülecek özel noktalar yoktur. ***AMSOFT J 41*** bilgisayarın arkasındaki ***USER PORTS*** (1/0) yazılı 9 yollu giriş fişine doğrudan doğruya takılır. Gerekirse ikinci oyun çubuğu da ilk oyun çubuğunun tabanındaki fiş girişine yerleştirilir. Bu aletin bağlantıları bütün öteki 1/0 fonksiyon ve bağlayıcılar ile birlikte, bu kitabın sonundaki EK LL de verilmiştir.



7.1 Oyun-Çubuğu

AMSTRAD CPC 464 program yapısı bir ya da iki oyun çubuğuna dayanır. Ve bunlar klavyenin bir parçası gibi muamele görür ve ***INKEY*** emirleriyle klavye tuşlarıymış gibi dikkate

alınabilirler. Dikkat edin, eğer oyun çubuğunuzda yalnızca bir ateşleme düğmesi varsa *AMSTRAD* CPC 464 terminolojisinde büyük bir ihtimalle "ateş 2" (bire 2) olacaktır.

Oyun çubuğunu doğrudan yoklamak için özel bir fonksiyon da vardır. Bu birinci oyun çubuğu için ***JOY*** (Ø) ikincisi için ***JOY*** (1) dir. Fonksiyon son klavye taramasında oyun çubuğu anahtarlarının durumunu gösteren bitler seviyesinde bir sonuçtur. Bir saniyede 50 klavye taraması olduğuna göre sonuç gerçekten oyun çubuğu anahtarlarını anında durdurur

Birinci oyun çubuğu	JOY (0)	Tuş	İkinci oyun çubuğu	JOY (1)	Tuş karşılığı	
Yukarı	Bit 0	72	Yukarı	Bit 0	48	6
Aşağı	Bit 1	73	Aşağı	Bit 1	49	5
Sol	Bit 2	74	Sol	Bit 2	50	R
Sağ	Bit 3	75	Sağ	Bit 3	51	T
Ateş 2	Bit 4	77	Ateş 2	Bit 4	52	G
Ateş 1	Bit 5	77	Ateş 1	Bit 5	53	F

Yukarıdaki tabloda tuş basılı kabul edilen kod numarasıdır. ***KARŞILIĞI*** ile kastedilen ise hangi tuşa basıldığında aynı sonucun alınacağıdır.

Dikkat edin. İkinci oyun çubuğunu kontrol edildiği zaman CPC 464 oyun çubuğu ile klavye tuşu arasındaki farkı ayıramaz. Pratikde yarım çalışması olması pek olası değildir. Aslında klavye ikinci oyun çubuğunun yerini tutmak üzere kullanılabilir.

AMSOFT J41'i kullanırken ikinci oyun çubuğu birinciyle eştir ve birinci oyun çubuğunu yan tarafındaki fiş girişine takılır. İkinci oyun çubuğunu kullanmak için özel kablolar gerekmez.

USER PORTS (1/ 0) yazılı 9 yolla giriş fişi başka bilgisayarla kullanılan standart oyun çubuklarını kabul eder, ancak bunlar ikinci oyun çubuğunu özel bir adaptör kullanılmadan takmaya müsait değildir ve böyle bir oyun çubuğunu ikinci oyun çubuğu olarak ***AMSOFT JYİ*** oyun çubuğunun yanına takılamaz.

Programcılar, programlarının başına kullanıcıya oyun çubuğu ya da kürsör tuşu çalıştırma imkanı sağlamayı düşünebilirler. (Bu durumda ***[COPY]*** tuşu veya başka belirtilmiş tuşlar ateşleme düğmesi olarak kullanılabilirler.

7.2 Yazıcı Kullanımı:

AMSTRAD CPC 464 standart ***Centronics*** arabirimli bir yazıcıya direk olarak bağlanabilir.

Yazıcı kablosu yazıcı ile bilgisayar arasında direk bir bağlantıdır.

Dikkat edin, bilgisayarın baskılı devresinde yazıcının bağlantısında olduğundan iki tane daha az yol vardır. Bu standart yazıcı, baskılı devresi kenar bağlayacısı kullanmaya elverişlidir.

Esas arabirim ayrıntıları ***EKV*** de gösterilmiştir. Kabloyu bilgisayarın üstündeki yol 1 yazıcıdaki yol 1'e bilgisayardaki yol 19 yazıcıdaki yol 19 vs. bağlanacak şekilde yapın. Yazıcıdaki yol 18 ve 36 bilgisayara bağlanmış olmayacaktır.

Özellikle bilgisayardaki alt sıra yollar 19 dan başlayarak numaralandırılmış, (üst sırada 17

yol olduğu için 18 ile başlaması beklenebilirdi) kullanılan bu telin bilgisayar kenar bağlantılı yollarıyla yazıcı bağlantı yollarının tam aynı numaralara bağlanması içindir.

Bilgisayar meşgül sinyaline (yol 11) yazıcı ile sevk sağlamak için kullanılır ve yazıcının *OFF LİNE* olmasını bekler.

Kullanıcı için *(setup)* emirleri gerekli değildir, ve çıkış kanal 8 belirtilerek printere yönlendirilir.

LIST # 8

Bu hafızadaki *BASIC* programını, list yazılmaya uygunsa, yani korunmamışsa list edilmesini sağlar.

Programların içinde yazıcıya seslenmek, hitab etmek (adress) Şu basit forma kullanarak mümkündür.

PRINT = 8, "Bu yazıcıya gönderilmiştir"

Birçok yazıcılar çıkış satırın sonuna varırsa, satır sonlarına otomatik olarak bir alt satıra atlar. (Yazıcı, el kitabına bakın) *AMSTRAD BASIC de WIDTH* emrinde söylendiği gibi çıkışı ayarlar. Yazıcının ilk değeri 132 genişlikte gereken değere ayarlanabilir.

Örneğin *WIDTH 80*

Özel olarak 255 değerine ayarlanırsa *AMSTRAD BASIC* çıkışı sarılmaz ve tamamen yazıcının satır sonlarını kontrol etmesine güvenir. *BASIC* yazıcının konumunun bir sayacında hazır tutar. Bu da *POS* fonksiyonuyla öğrenilebilir.

IF POS (# 8)> 50 THEN GOTO 1000

CPC 464 satırın sonunda bir 'line feed' *CHR$* (10) ve satır başı *CHR$* (13) gönderir. Yazıcıda da genellikle uygun çıkış formunu seçmek için önceden ayarlanmış bir anahtar vardır. Ve çıkış almaya başladığında ilk değerin ne olduğunu derhal açık seçik belli olacaktır.

7.3 YAZICI ile GRAFİK

Yazıcınızla birlikte verilen elkitabında kontrol kodları belirtilmiştir.

Bunlar genellikle şu formdadır.

PRINT CHR$ (n)

Bazı yazıcıların EK 111 de listelenen *AMSTRAD* grafik karekterlerinin çoğuna benzer karakterleri olabilir. Ama, karekter numaralarının tam olarak birbirini tutma ihtimali çok azdır. Bu yüzden kendi printerinize uyacak bir değiştirme tablosunu kendiniz yapmalısınız. Yazıcı arabirimi ucuz *"DOT MATRİX"* yazıcılarla kullanılması gözönünde olarak yapılmışsa da uygun bir arabirim ile *"PAPATYA ÇARKLI"* yazıcılarla, grafik çizicilerle ve renkli yazılarla kullanılabilir. Bağdaşma standart paralel arabirim kullanıldığı için çok kolaydır.

8. AMSTRAD BASIC'e kısa ve özlü referans rehberi

Bu bölümün içerdiği konular:

* Yazılış
* Özel karakterler ve anlamları
* Alfabetik sırayla bütün AMSTRAD BASIC komutları.

Bu bölüm CPC 464 ile ROM'daki BASİC fonksiyonların özlü bir özetini içerir. Sağlanan özelliklerin kapsamı, CPC 464'in özelliklerini tamamlayan uzantılarla birlikte, dilin endüstri standardı kullanımını temsil eder.

8.1 Yazı Şekli

Özel karakterler.

&	ya da H Hexadecimal (16 lık taban)
&X	Bınary (2'li taban) sayı için öntakı
?	Aynı satıra yazılan bildirimleri ayırır.
#	Kanal gösterimi için öntakı.

Veri Tipleri:

Karakter dizileri 0 dan 255 karaktere kadar uzunlukta olabilir.

Sayısal veriler tamsayıya gerçek olabilir. Tam sayı veriler -32768.....32767 arasında bir sıralamada tutulürlar ve gerçek veriler 9 dizilik bir kesinliğin biraz üstünde + 1.7 E + 38 lik sıralamada sıfırın üstünde en küçük değerin aşağı yukarı 2.9 E-39 da tutulurlar. % Tam sayı! gerçek ,$ karakter dizisi anlamına gelir. < sayısal ifade > sayısal değerle sonuçlanan herhangi bir ifadedir: yalnızca sayı olabilir ya da sayısal bir değişken ya da değişkenlerle işletilen sayılar olabilir. Yani bir karakter dizisi olmayan aşağı yukarı herşey. Bir karakter dizisi metnin belirli bir kanaldan gönderilmesi gerektiğinde, ekranı, kaseti ya da printeri tanıyan sayısal ifadeye referans yapar

Bir geçersiz girdi bildirimi, sayısal değer sınırı olarak tanımlanan alanın dışında kalan bir değerle döndüğü ya da bir emir parametresinin bir şekilde geçersiz olduğu ve BASIC'in girişi geçerli kabul etmediği anlamına gelir.

KOMUTLAR

Amstrad komutları burada şu formu kullanarak sıralanmıştır. söz dizimi/fonksiyon
Örnek

tanımlama
Bağlantılı komutlar, fonksiyonlar

ÖNEMLİ
KOMUTLAR: doğrudan yapılan işlemler.
FONKSİYONLAR: bir ifadede girdi olarak istenen işlemler.

Parentezler:

( ) bunlar bir emrin ya da fonksiyonun parçası olarak gereklidirler. KOMUTLARIN tanımında kullanılan başka tip parentezler, tanımlama amacı içindir ve satırın bir parçası olarak yazılmamalıdır.

[ ] Seçime bağlı parçaları içine alır.

< > sonra gelen tanımlamada tanımlanan değişik ifadeleri içine alır.

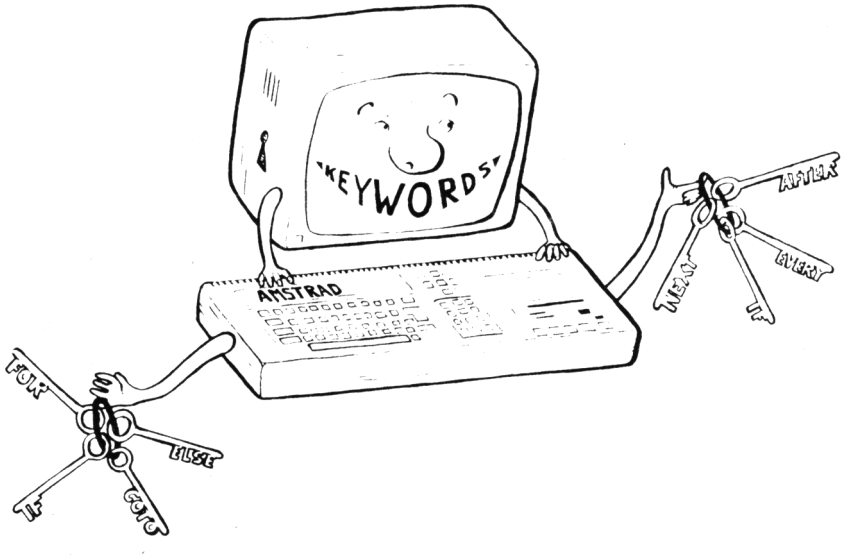
Tırnak işaretleri:

Yalnız " " bu tipleri esas *BASIC* yapısının bir parçasını oluşturur. Bu tipleri tanımlamanın bazı taraflarını belirginleştirmek için kullanılır bunlar tarakter dizilerinin ya da metnin dışında girişlerde kullanılmamalıdır.

Giriş

BASIC küçük harf girilen bütün komutları *LIST* ederken büyük harfe çevirir. Program *LIST* edildiğinde büyük harfe çevrildiği için buradaki bütün örnekler büyük harfle yazılmıştır. Siz küçük harf yazarsanız yazma yanlışlarını daha çabuk fark edersiniz. Çünkü *LIST* edildiğinde doğru yazılanlar büyük harfe çevrilirken, yanlış yazılanlar küçük harf kalır.

Komutlar boşluklarla sınırlandırılırlar.Çünkü, *AMSTRAD BASIC* tuş sözcüklerini değişken isimlerin içinde bulundurmanıza izin verir. Örneğin *AMSTRAD BASIC* de değişken olarak kabul edilebilir.



ABS

ABS **(sayısal ifade)**
PRINT ABS (-67.98)
67.98

Fonksiyon: verilen ifadenin mutlak değerini geri verir. Kısa olarak bu, negatif sayıların pozitife döndüğü anlamına gelir.

Bağlantılı fonksiyon: *SGN*

AFTER:

AFTER (tam sayılı girdi) [(tam sayılı girdi)]

GOSUB (satır numarası)

AFTER 200, 2 GOSUB 320

KOMUT: verilen bir zaman dönemi geçtikten sonra bir alt program çağırmak. İlk (tam sayı) 0.02 saniye birimleriyle gecikme süresini belirtir. İkinci (tam sayı) da (0....3 arasında olmalı) eldeki dört geciktirme zamanlamasından hangisinin kullanılması gerektiğini belirtir.

Bağlantılı komutlar: *EVERY, REMAİN*

ASC

ASC (karakter dizisi)

PRİNT ASC ("X")
88

FONKSİYON: ASC11 karakterleri kullanıldığı takdirde bir karakter dizinin ilk karakterinin ASC11 kod numarasını verir.

Bağlantılı fonksiyon: *CHR$*

ATN

ATN (sayısal ifade)

PRINT ATN (1)
0.785398163

FONKSİYON: Belirtilen değerin arktanjantını hesaplar Sayısal ifade radianları-PI/2 den PI/2 arasında olan bir reel sayıya dönüştürür.

Bağlantılı fonksiyonlar: *SIN, COS, TAN, DEG. RAD*

AUTO

AUTO [(satır numarası)] [(artış)]

AUTO 10,50

KOMUT: Satır numarasını otomatik olarak çıkartmak içindir. Satır numarası, var olan bir programın sonuna ekleme yapmak istediğiniz durumlarda, çıkacak ilk satırı ayarlar. Satır sayıları arasındaki artış, değeri ve çıkacak ilk satırın numarası, belirtilmediği takdirde, ilk değer kendiliğinden 10 dur. Varolan bir satır üstüne yazılma tehlikesinde olursa *BASIC* uyarı olarak çıkan satır numarasından sonra bir yıldız* koyar.

BIN $

BIN $ (işaretsiz tam sayı) [(tam sayı)]

PRINT BIN $ (64,8)
01000000

FONKSİYON: İşaretsiz tam sayı değerini temsil eden bir binary karakter dizisini öndeki sıfırları ikinci tam sayılı ifade ile komutlanan hane sayısıyla doldurarak meydana getirir.

Bağlantılı fonksiyonlar : *HEX$, STR$*

BORDER

BORDER (renk) [(renk)]

BORDER 3,2

KOMUT: Ekranda border (sınırın) rengini değiştirmek içindir. İki renk belirtilirse, border rengi, bu iki renk arasında değişir. Bu değişme hızı ***SPEED INK*** emriyle verilirse renkler verilen hızda değişir. Border renkleri 0.....26 arasındadır.

Bağlantılı komut: *SPEED INK.*

CALL

CALL (adres) [(parametre) listesi]
CALL &BD 19

KOMUT: Makina dilinde geliştirilmiş bir alt programı *BASIC*'den istenmesini sağlar. Dikkatle kullanılması gerekir.

Deneyimsizlerin üstünde denemeler yapacağı bir fonksiyon değildir. Yukardaki *CALL* nisbeten zararsızdır. Çünkü, bir sonraki ***Frame fly back*** bekliyoruz. ***Frame fly back*** özellikle animasyon etkileri yapılırken karakterlerin ekranda hareketini toparlamakta yararlıdır.

Bağlantılı KOMUT: *UNT*

CAT

CAT
CAT

KOMUT: BASIC'ın kaseti okumaya başlamasına ve bulunan bütün program isimlerinin görüntülenmesine neden olur. Bu hafızadaki en son programı etkilemez ve hafızadaki programı değiştirmeden önce yeni kaydedilen programı kontrol etmek için kullanılabilir. Emir kaseti çalıştırmanızı ister ve bir program bulunca yanıt verir.

FILENAME blok numarası sembol ok.

Semboller yapılan kayıt tipini belirtir:

$ bir *BASIC* programı
% korunmuş *BASIC* program
* bir *ASCII text* kütüğü
& bir *Binary* kütük

KÜTÜK *BASIC* tarafından yapılmışsa başka karakterler de olabilir. Bağlantılı komutlar: *LOAD, RUN, SAVE*

CHAIN
CHAIN MERGE

CHAİN (kütük ismi) [(satır numarası)]
CHAIN MERGE (kütük ismi) [(satır numarası)]
[*DELETE*(satır numarası sıralaması)]

CHAIN "TEST", 350

KOMUT: CHAIN, hafızada bulunan programın yerine kasetten bir programı yükler *LOAD. CHAİN MERGE* kasetten bir programla hafızadaki en son programa katar. Kütügün içeriğini hafızada en son bulunan programa ekler. Satır numarası, yeni programı ***chain merge*** olduktan sonra, emrin kaçıncı satır numarasından başlayarak yerine getireleceğini belirtir. *BASIC* eldeki son satır numarasını başlangıç kabul eter. Kütük ismi bildirilmezse *BASIC* teypde rastladığı ilk geçerli programı okur. Kütük isminin ilk karakteri! ise, bu kütük isminden çıkartılır ve kaset okuma süresinde kasetten çıkan olağan sesleri bastırır.

Tanımlanabilir fonksiyonlar. *(USER FUNCTIONS)* ve açık kütükler atılsa *CHAIN MERGE* en son değişkenlerin hepsini tutar. *ON ERROR GOTO* kapatılır, *bir RESTORE* kullanılır ve *DEFINT, DEFREAL DEFSTR* yeniden oluşturulur ve bütün aktif *FOR WHİLE VE COSUB* emirleri unutulur. Korunmuş kütükler merge olmaz.

Bağlantı komutlar: *LOAD, MERGE*

CHR$

CHR $ (sayısal ifade)
PRINT CHR$ (100)
d

FONKSİYON: bir sayısal değeri karşılığındaki karaktere çevirir. (AMSTRAD CPC 464 karakter setini kullanarak. EK 111)

Bağlantılı fonksiyonlar: *ASC, LEFT$, RİGHT $, MID $, STR $*

CINT

CINT (sayısal ifade)

10 N = 578.76543
20 PRINT CINT (n)
RUN
579

FONKSİYON: Verilen değeri - 32768.... 32767 arasında yuvarlayarak bir tam sayıya çevirir.

Bağlantılı fonksiyonlar : *CREAL, INT, FIX, ROUND, UNT*

CLEAR

CLEAR
CLEAR
KOMUT: Bütün değişkenleri ve kütükleri temizler.

CLG

CLG [(yeni ink rengi)]
CLG
KOMUT: Ekrandaki grafikleri temizlemek içindir.
Bağlantılı komutlar: *CLS, ORIGIN*

CLOSEIN

CLOSEIN
CLOSEIN
KOMUT: Kaset input kütüğünü kapat. *NEW* ve *CHAİN MERGE* gibi emirler açık bir kütüğü terkeder.
Bağlantılı komutlar: *OPEN İN, CLOESOUT*

CLOSEOUT

CLOSEOUT
CLOSEOUT
KOMUT: output kaset kütüğünü kapat.
Bağlantılı komutlar: *OPENOUT, CLOSEIN*

CLS

CLS [# (Kanal numarası)]
CLS
KOMUT: Verilen ekran penceresini (window) kendi kağıt renginde temizlemek içindir.

CONT

CONT
CONT

KOMUT: * *BREAK* *, *STOP* ya da *END*den sonra programı yerine getirmeyi (***execute***), program değişikliği uğratılmadığı sürece, sürdürülür. Direkt emirler girilebilir.

COS

Cos [(sayısal ifade)]

? COS (34)
-0.848570274

ve

deg: ? COS (34)
0.829037573

FONKSİYON : Verilen değerin kosinüsünü hesaplar. *DEG* emriyle özellikle tersi komutlanmadıkça bu fonksiyon girdisini radyan kabul eder. Dikkat edin: yukardaki örnekte *PRINT* in kısa formu? kullanılmıştır ve fonksiyonlar küçük harf yazılmıştır. *AMSTRAD*

BASIC'de bağdaşan özellikli takımlar.

Bağlantılı Fonksiyonlar: *SIN, TAN, ANT, DEG, RAD.*

CREAL

CREAL [(sayısal)]

```
5 DEFINT N
10 n= 75.765
20 d= n/34.6
30 PRINT d
40 PRINT CREAL (n)
50 PRINT n/55.4
run
2 19653179
76
1.37184116
Ready
```

FONKSİYON: Bir değeri reel sayıya çevirir.
Bağlantılı Fonksiyonlar. *CINT*

DATA

DATA *(veri listesi)*

```
5 DİM A$ (2), B$(2).
10 FOR I=1 TO 2.
20 READ A$ (I), B$ (I).
30 NEXT I
40 FOR I=1 TO 2: FOR J=1 TO 2.
50 PRİNT A$ (I); " ";B$ (J).
60 NEXT : NEXT
70 DATA "ALI", "OKULA", "AYŞE", "EVE"
```

KOMUT: Program içinde kullanım için sabit veriler bulundurur. *BASIC*'ın en çok kullanılan özelliklerinden biridir. Sabit verileri gerektiğinde ortaya çıkarmak üzere *DATA* bildirimlerinde biriktirir. Veri tipi onu isteyen değişkene uygun olmalıdır. Bir ***DATA bildirimi programın herhangi bir yerinde görünebilir.***

Bağlantılı komutlar: READ, RESTORE.

DEF FN

DEF FN (isim)'[(fonksiyon girdileri)] = (genel ifade)

```
10 DEF FN sec (x)= 1/cos (x)
20 İNPUT X.
30 PRİNT. cos (x), FN sec (x)
```

KOMUT: Kullanıcının kendine has fonksiyonları kullanmasına ve tanımlanmasına izin verir. *DEF FN* bu mekanizmanın tanımlanma kısmıdır ve programa özgü fonksiyon yaratır.

bu fonksiyon aynı *COS*'un *BASIC* fonksiyonu olarak çalışması gibi program içinde çalışır.

DEF FN program boyunca istenebilir. Değişken tipine uygun olmalıdır ve *DEF FN* fonksiyon emri ana programın dışında yazılmalıdır.

Bağlantılı komutlar:

DEFINT
DEFSTR
DEFREAL

Deftipi (harf sıralaması)

DEFINT I-N
DEFSTR A, W—Z
DEFREAZ

KOMUT: İlk değişken tiplerini, 'tip' tam sayı gerçek sayıya da karakter dizisi olduğundan, tanımlar. Değişken değişkenin ilk harfine göre ayarlanır. Bu büyük harf ya da küçük harf olabilir.

Bağlantılı komutlar: *LOAD, RUN, CHAIN, NEW, CLEAR*

DEG

DEG
DEG

FONKSİYON: Derece *MOD*'unu ayarlar. İlk koşulu *SIN* ve *COS* gibidir. Yani sayısal veriler için radian ölçü varsayar. Emir, *CLEAR* ya da *RAD* ile başka türlü tanımlamaya kadar ya da yeni bir program yükleninceye kadar derece moduna ayarlar.

Bağlantılı fonksiyonlar: *RAD*

DELETE

DELETE (satır numarası sıralaması)

DELETE 100—200

KOMUT: En son programın "satır numarası sıralamasında" ifadesinin tanımladığı parçasını ortadan kaldırır. Yanlışlıkla çıkartılırsa, geri alınamaz. Bu yüzden dikkatle kullanın ve girmeden önce yazı hatalarını kontrol edin.

Bağlantılı komut: *NEW*

DI

DI

```
10 CLS
20 TAG
30 EVERY 10 GOSUB 100
40 X 1= RND * 320: X2=RND*320
50 Y= 200+RND*200
```

```
60 FOR X= 320-X1 TO 320+X2 STEP 2
70 DI: PLOT 320, 0, 1: MOVE x-2,
     Y: PRINT " "; :MOVE X, Y: PRINT "#";:EI
80 NEXT
90 GOTO 40
100 MOVE 320,0
110 DRAW x+8, Y-6,0
120 RETURN
```

*KOMUT: İnterrupt'ları (*Break* * dışındaki interrupları açık bir şekilde *EI* ya da *GOSUB* kesme programının sonunda *RETURN* ile yeniden harekete geçirilinceye kadar etkisiz kılar.

Program kelimenin tam anlamıyla kesintisiz devam etmek istediği zaman kullanılır. Örneğin: kaynakların kullanımı için iki altprogram yarış halindeyden yukarıdaki örnekte esas program kesme alt programı ile grafik görüntü için yarışıyorlar.
Bağlantılı komutlar : *EI*

DIM

DIM (değişken listesi)

```
10 DİM A (20)
20 FOR I= I TO 20 : İNPUT A(I): NEXT
30 FOR I= 20 TO 1 STEP-1 ; PRİNT a (I): NEXT
```

KOMUT: Matrislere yer ayırmak için kullanılır. BASIC'e bir matris için ayırılacak yer hakkında bilgi vermelidir. Yoksa ilk değer olarak 10 verir. Bir kez açık ya da kapalı olarak ayarlandıktan sonra değiştirilemez, yoksa hata ile sonuçlanır.

Bir matris değişkeni aynı değişken isminin boyut listesi içeren tam sayılar listesinde bildirilen bir seri değeri alabildiği durumdaki değişkendir. Boyut listesindeki ilk sayı çok katlı bir araba parkının katları olarak düşünülebilir ve geri kalan sayılarda araba park yerleri vs. İleri *BASIC* programcılığının başlıca elemanı matrislerin çok iyi anlaşılmasıdır. Bir matrisin boyu yalnızca eldeki hafıza ve programcının boyutları listesindeki girişlerin hesabını tutabilme yeteneğiyle sınırlıdır.
Bağlantılı komut: *ERASE*

DRAW

DRAW (x koordinatı), (y koordinatı) , [(ink rengi)]

```
DRAW 200, 200, 13
```

KOMUT: Grafik kürsörünün son konumundan mutlak bir konuma çizgi çizer Koordinat konumları 3 değişik mod da da ayrı kalır. Daha fazla örnek 5. bölümde vardır.
Bağlantılı komutlar: *DRAWR, PLOT, PLOTR, MOVE, MOVER, TEST, TESTR, XPOS, YPOS, ORIGIN*

DRAWR

DRAWR (x uzaklığı), (y uzaklığı) [(ink rengi)]

```
DRAWR 200, 200, 13
```

EMİR: Grafik kürsörünün son konumundan buna göre bir konuma çizgi çizmeli.
Bağlantılı komutlar: *DRAW, PLOT, PLOTR, MOVE, MOVER, TEST, TESTR, XPOS, YPOS, ORIGIN*

EDIT

EDİT (satır numarası)

```
EDIT 110
```

KOMUT: Programda, belirli bir satır numarası isteyerek, düzeltme yapar, Bölüm 1.4'e bakın.
Bağlantılı komut: LIST

EI

EI
EI
KOMUT: DI emriyle kaldırılan interrupt 'ı yeniden geçer kılmak.
Bağlantılı komutlar : *DI*

END

END
END
KOMUT: Program sonucunu bildirir. *AMSTRAD BASIC*'de bir program en son komut satırını geçtiğinden bir *END* gereklidir. *END* bütün kaset kütüklerini kapatır ve direkt moda döner. Ses sıraları boşalıncaya kadar sürer.
Bağlantılı komut: *STOP*

ENT

ENT
ENT (envelope sayısı) (envelope bölümleri)

```
10 ENT 1, 100, 2, 20
20 SOUND 1, 100, 1000, 4, 0,1
```

KOMUT: Ses çıkartılırken tonunu değişikliklere uğratmak mümkündür. Ton envelopu tonda nasıl değişiklikler olacağını tanımlar. Bölüm 6 ve EK V11'ye tam tanımlama ve bunun işletilmesi için bakın. Envelope 1.....15 arasında olması gereken bir tam sayıdır. Bunu da ayarlanacak ton envelop'u belirtir. Envelope sayısı negatif o zaman envelope tekrarlanır.

Beşe kadar envelope bölümleri sağlanabilir ve her biri şu formlardan birini alabilir: (basamak sayısı), (basamak boyu), (duraklama zamanı) ya da = ton periodu, (duraklama zamanı)

Birinci form, en son ton dönemine göre bir artma değişikliği belirtir. İkinci fazın ton dönemine mutlak bir ayar belirtir. Burada basamak sayımı, bölümdeki basamak adedini verir 0.....239 arasında bir tam sayı.

(basamak boyu) envelope'deki her basamakta ton dönemini değiştirecek miktarı verir. -128.....127 de eğri veren bir tam sayı.

(duraklama zamanı) basamaklar arasında bekleme zamanını verir. Zamanı 0.01 ünitelerinde veren bir tam sayıdır. Bu sayı 0.....255 (burada 0 256 olarak muamele görür) arasında bir değer olmalıdır.

(ton dönemi) döneme yeni bir ayar verir. 0....4095 arasında bir değer veren bir tam sayıdır.

SOUND emri ilk ton periodunun ayarlar ve onbeş ton envelope'undan birini belirtebilir. Eğer hiç envelope, ya da daha önce ayarlanmış bir envelope belirtilmezse o zaman ton ses boyunca sabit kalır. Ton envelop'unun sesin süresinde hiç bir etkisi yoktur. Ses *(SOUND)* bittiğinde ton envelope da basamaklar kalmışsa, bunlar terkedilir. Tekrar eden bir ton envelope, ses bitinceye kadar her bittiğinde yeniden başlar.

Ton envelope'undaki ifadeler, emir yerine getirildiğinde ve sonuçlar daha ilerki kullanımlar için saklandığında değerlendirilir. Ton envelope kullanmak emrin yeniden yerine getirilmesine neden olmaz. Verilen ton envelope'unun bu yeni ayarlanışında bir öncekinin değeri kaybolur. Bir envelope'u onu kullanan ses aktifken ya da olmak üzereyken değiştirmek, geçici (ama enteresan) etkiler meydana getirir.

Bölümleri olmayan bir envelope belirtmek daha önceki yapılmış ayarları iptal eder. Envelope'un daha ileri kullanımları yok sayılır. Ve bunların yerine ilk değerler kullanılır.
Bağlantılı komutlar: *ENV, SOUND*

ENV

ENV (envelope sayısı) [(envelope bölümleri)]
10 ENV 1,00, 2,20
20 SOUND 1, 100, 1000, 4,1

EMİR: Bir ses çıkartılırken, onun ses düzeyini değişikliğe uğratmak, bu emri kullanarak mümkündür. Bir ses düzeyi envelope şunlardan oluşur:
(basamak sayısı), (basamak boyu) (duraklama zamanı)
Ve sesin düzeyinin nasıl değişeceğini, 0.....127 arasında basamak sayısı, -128.....127 arasında basamak boyu ve 0.01 saniye aralarla, 1.....256 arasında duraklama zamanı belirterek tanımlar.
Envelope sayısı tam bir sayıdır. 1.....15 arasında ayarlanarak envelope'u belirten bir değer verir.
Beşe kadar envelope bölümleri verilebilir. Her biri şu formlardan birini alabilir.
(basamak sayımı) , (basamak boyu) , (duraklama zamanı)
ya da: = (donanım envelope) (envelope periodu)
Birinci form program kontrolu altında bir envelope belirler. Burada parametreler şunlardır:

(basamak sayısı) bölümdeki basamak adedidini verir- 0.....127 arasında bir tam sayıdır.
(basamak boyu) envelope'daki bu basamağın genliğinin de değiştirileceği miktarı verir. -128.....+ 127 arasında bir tam sayıdır.

(duraklama zamanı) basamaklar arasında bekleme zamanını verir. Zamanı 1/100 saniyede belirten bir tam sayıdır. Sayıya 0.....255 (burada 0.256 olarak muamele görür) arasında değişen bir değer vermelidir.

İkinci fazın ses tarafından yerine getirilmesi gereken bölümü belirtir. Burada: donanım envelope, (elektronik envelope) envelope registerina yerleştirilecek değerdir (register, 15 oktal) ayarlanacak. (envelope periodu) envelop period registerlarına verilecek değildir. (register 13 14 oktal)

Donanım envelope ayarlarının bağlantılı duraklama zamanı yoktur. Bu yüzden envelope'un bir sonraki bölümü hemen yerine getirilir. Ve bir sonraki basamağın uygun duraklaması olması salık verilir. Eğer bir sonra basamak yoksa 2 saniyelik bir duraklama olur.

İlk ses düzeyini ***SOUND*** emri ayarlar ve onbeş volume envelepe'undan birini belirtebilir.

Eğer bu envelope ya da daha önce ayarlanmış bir envelope belirtilmezse, o zaman ses boyunca yükseklik sabit kalır. Sıfırlık bir (basamak boyu)nu sıfır olmayan bir (basamak sayımına) ayarlamak son yükseklik ayarının kalmasıyla sonuçlanır.

Genlik envelope'un daki ifadeler emir yerine getirildiği ve sonuçlar daha sonraki kullanım için depolandığı zaman değerlendirilir. Genlik envelope'u kullanmak emrin yeniden yerine getirilmesine neden olmaz. Her yeni genlik envelope'u ayarladığında bir önceki kaybo-

lur. Envelope'u onu kullanan ses *(SOUND)* aktifken ya da olmak üzereyken değiştirmek geçici (ama güzel) etkiler meydana getirir.

Bölümleri olmayan bir envelope belirtmek daha önceki ayarlamaları iptal eder. Envelope'un daha ileri kullanımı yok sayılır. Ve bunun yerine ilk değerler kullanılır.

Bağlantılı komutlar: *ENT, SOUND.*

EOF

EOF
PRINT EOF
-1

FONKSİYON: Kaset input'unun kütüğün sonunda olup olmadığını test eder. Sonda olduğu doğruysa -I yoksa Ø verir.

Bağlantılı komut: *OPENİN*

ERASE

ERASE (değişken isimlerin listesi)
ERASE A, B$

KOMUT : Bir matris artık gerekmiyorsa *ERASE* ile silinebilir ve hafıza başka kullanım için hazır hale getirilir.

Bağlantılı komut: *DIM*

ERR
ERL

ERR
ERL
10 CLS
20 ON ERROR GOTO 1000
30 READ A
40 DATA 3,5 7,9
50 GOTO 30
1000 IF ERR=4 THEN PRİNT "DATA okuma hatası"
101/ Ø END

DEĞİŞKEN: Bu değişken hatalarla ilgili alt programlarda, hatanın numarasını ve olduğu satırı bulmak için kullanılır. EK VIII de hata mesajı listesine bakın.
Bağlantılı komutlar: *ON ERROR, ERROR*

ERROR

ERROR (tam sayı ifadesi)
ERROR 17

KOMUT: Verilen bir hata numarasıyla hata tedbiri alır. Hata *BASIC* tarafından fark edilmiş ve kullanılmış olabilir (EK VIII-bu durumda alınan tedbir, *BASIC* Hatayı tesbit etseydi alınan tedbirle aynı olurdu. *BASIC* tarafından yanlış sayılardan yüksek olanlar program tarafından kendi hatalarını haber vermek için kullanılabilir.
Bağlantılı Komutlar: *ON ERROR, ERR, ERL*

EVERY

EVERY (tam sayılı ifade) [(tam sayılı ifade)]

GOSUB (Satır numarası)
KOMUT: CPC 464 gerçek zaman saati sürdürür. *EVERY* emri bir *BASIC* programına bir alt programın düzenli aralıklarla çağırılmasına imkan verir. Dört erteleme zamanlandırması vardır. Bundan ilkini tam sayı ile 0.....3 arasında belirtebilir ve bağlantılı alt prorgamları olabilir.
Bağlantılı komutlar: *AFTER, REMAIN*

EXP

EXP (sayısal ifade)
PRINT EXP (6.876)
968.743625
FONKSİYON: "e" nin verilen üssünü hesaplar. Burada "e" yaklaşık 27182818 dir ki buda doğal logarıtması 1 olan bir sayıdır.
Bağlantılı fonksiyon: LOG

FIX

FIX (sayısal ifade)
PRINT FIX (9.99999)
9
FONKSİYON: CINT'ın tersine *FIX* sayısal ifadenin ondalık noktanın sağındaki parçasını kaldırır sıfıra toparlıyarak bırakır. Bir tam sayı sonuç bırakır.
Bağlantılı tuş sözcükleri: *CINT, INT, ROUND*

FOR

FOR (basit değişken) = (başlangıç değeri) *TO* (son değer) [adım]
FOR GUN = 1 to 5 STEP 2
KOMUT: Bir programın bir bölümünü, başlama ve bitiş değeri arasına bir kontrol değişkeni belirli adımlarla koyarak, belirlendiği kadar tekrarlanır,*STEP* belirtilmezse adım 1 olur.
Bağlantılı komutlar: *NEXT, WHILE*

FRE

FRE (sayısal ifade)
FRE (karakter dizisi)
PRİNT FRE (Ø)
PRINT FRE ("" "")
FONKSİYON: BASIC tarafından kullanılmamış ne kadar hafıza kaldığını meydana çıkartan *FRE* ("" "") eldeki boşyerin cevabını getirmeden önce bir hafıza toplaması yaptırır.

GOSUB

GOSUB (satır numarası)
GOSUB 210
KOMUT: Belirtilen satır numarasına atlıyarak bu *BASIC* alt programı çağırır.
Bağlantılı komut *RETURN*

GOTO

GOTO (satır numarası)
GOTO 90
Belirlenmiş bir satır numarasına atlar.

HEX$

HEX$ (işaretsiz tam sayı) [tam sayı]
PRINT HEX$ (65534)
FFFE
FONKSİYON: verilen sayıyı hexadıcimal (16 lı taban) forma çevirir.
(EK 11 ye bakın)
Bağlantılı fonksiyonlar: ***BIN $, STR $***

HIMEM

HIMEM

? HIMEM
43903
DEĞİŞKEN : BASIC tarafından en yüksek byte'in adresini verir ve olağan şekilde sayısal ifadelerle kullanılabilir.
MEMORY emri kullanılarak eldeki hafızanın en yüksek byte'ini tekrar ayarlamadan önce *mm = HIMEM* emri çıkartmak akıllıca olur. Bundan sonra önceki hafıza kapasitesine ***MEMORY mm*** emriyle dönebilirsiniz.
Bağlantılı değişkenler: ***FRE, MEMORY***

IF

IF
IF (mantık ifade) *THEN* (seçenek kısım) *[ELSE* (seçenek kısım)]
IF A> B THEN A=C ELSE A=D
IF A> B GOTO 1000 ELSE 300
KOMUT: Bu çok fazla kullanılan emir, bir programın atlama noktalarını saptamak için şartlı kullanılır. Mantık kısmı değerlendirilir ve doğru olduğu takdirde ***THEN*** ya da ***GOTO*** kısımları yerine getirilir. Yanlış ise program ***ELSE*** kısmına atlar, ya da doğruca bir sonraki satıra geçer. Bildirimler herhangi bir uzunlukta olabilir. Bu satır uzunluğuyla sınırlıdır. ***IF*** satır sonuyla biter. Aynı satıra ***IF***'den bağımsız başka bildirimler konulamaz.
Bağlantılı KOMUTLAR: ***THEN, ELSE, GOTO, OR, AND, WHİLE***

INK

INK (mürekkep), (renk), [(renk)]
KOMUT: Kullanılan ekran ***MOD***'una göre (bölüm 5) birtakım renk olanakları vardır. Kullanılan renk ya da renkler ***INK*** emriyle değiştirilebilir. Bu EK VI daki renk değerleri tablosuna göre yapılır.
Bağlantılı komutlar: ***PEN, PAPER***

INKEY

INKEY (tam sayı)

```
10 CLS: IF İNKEY (55) = 32 THEN 30 ELSE 20
20 GOTO 10
30 PRİNT "SHİFT ve V basılı"
40 FOR Z=1 TO 1000 : NEXT: GOTO 10
```

FONKSİYON: Bu fonksiyon klavyeyi, hangi tuşların bastırılmakta olduğunu inceler. Klavye 0.02 saniyede taranır. Bu fonksiyon özellikle E/H cevaplarını farketmek için yararlıdır. Çünkü yorum seçeneklerinden birine göre shift'in durumu gerekli değildir. Yukarıdaki ör-

nek *[SHIFT]* ve *V* beraber basıldığı zaman, bunu saptar. Shift ve kontrol aşağıdakilere göre tanınır:

	[SHIFT]	[CTRL]	[KEY(tuş)]
-1	?	?	Yukarı
0	YUKARI	YUKARI	AŞAĞI
32	AŞAĞI	YUKARI	AŞAĞI
128	YUKARI	AŞAĞI	AŞAĞI
160	AŞAĞI	AŞAĞI	AŞAĞI

Bağlantılı komutlar: INPUT, INKEY $

INKEY $

```
INKEY $
10 CLS
20 PRINT "SİZ PROGRAMLAMA BİLİYOR MUSUNUZ (E-H)"
30 a$= INKEY $ : IF a$= "" GOTO 30
40 IF a$ = "E" OR a$= "e" THEN PRINT " O ZAMAN AMSTRAD'ı ÇOK SEVECEKSİNİZ"
150 IF a$= "n" OR a$= "N" THEN PRINT " hemen öğrenin"
60 GOTO 20
```

FONKSİYON: Kullanıcıya bu yanıttan sonra *ENTER* tuşuna basmadan karşılıklı ilişki sağlamak için klavyeden bir tuş okur. Bir tuşa basılmışsa fonksiyon cevap verir. Basılan bir tuş yoksa, işlem için gerekli bir giriş farkedilinceye kadar çevrimleme için kullanılan 30. satıra dönmeyi sürdürür.

Bağlantılı fonksiyonlar: *INPUT, INKEY*

INP

INP (kapı numarası)
PRINT INP (&FF 77)
FONKSİYON: Adreste belirtilen 1/0 kapısından giriş değeri getirilen bir fonksiyondur.
Bağlantılı fonksiyonlar: *OUT, Wait*

INPUT

Input [# (kanal numarası),] [;] [(çift tırnakta yazılacak olan ifade);] ([değişken] listesi)
or INPUT [= (kanal numarası),][;] [(çift tırnakta yazılacak ifade),]

```
10 İNPUT "İKİ SAYI VERİN"; A, B
20 C= (A+B+ABS(A-B)/2
30 PRİNT C; "büyüktür"
```

KOMUT: Bildirilen kanaldan verileri okur. *INPUT* dan sonraki noktalı virgül satırdan sonra yazılanın, satır başının girilmesini önler. Çift tırnak içindeki ifadeden sonraki noktalı virgül bir soru işareti görüntülenmesine neden olur. Virgül soru işaretini önler. Eğer yanlış tip bir giriş yapılmışsa (örneğin O harfi) (sayısal ifade 0'ın yerine) o zaman *BASIC* şöyle uyarır:

? Redo from start

...ve girdiğiniz ilk metin.

Bütün cevapların *[ENTER]* ile bitmesi gerekir. (kanal numarasından) sonra noktalı virgül satır sonunda satır başını önler, böylece kürsörü son girilen satırın sonunda bırakır. Kaset kanalı gösterildiği yerde hiçbir işaret çıkmaz. Eğer herhangi birşey belirtilirse kaset yazılımı tarafından yok sayılır. Böylece aynı program kanalların birinden ya da ötekinden okunabilir.

Verilen listede her değişken için klavyeden bir parça okunur. *INPUT* emrinde belirtilen tiple uyumlu olmalıdır. Bu da bir sayısal değişkendir. Virgül,satırbaşı, boşluk *MOD*'u kütük sonu ile biter. Arkadan gelen boşluklardan sonra virgüller ve *[ENTER]* yok sayılacaktır. Karakter dizileri, çift tırnak işaretiyle bitirilinceye kadar kelimesi kelimesine okunur. Bunlardan sonraki girişler sayısal değerlerde olduğu gibi yok sayılırlar.

Bağlantılı komutlar: *LINE, INPUT, READ, INKEY $*

INSTR

INSTR [(bir tam sayı,)] (karakter dizisi), (karakter dizisi))
PRINT INSTR (2, "BANANA", "AN")
FONKSİYON: Birinci karakter dizisinin içinde, ikinci karakter dizisi varsa, bunun yerine belirtir. Burada baştaki seçenek sayısı aramanın nereden başlayacağını gösterir, yoksa arama birinci karakter dizisinin ilk karakterinden başlar.
Bağlantılı fonksiyonlar: *MID$, LEFT$, RIGHT$*

INT

INT (sayısal ifade)
PRINT INT (-1.995)
-2
FONKSİYON: Sayıyı kesirli kısımlarını kaldırarak en yakın daha küçük tam sayıya yuvarlar. Pozitif sayılarda *FIX* gibidir. Ama tam sayı olmayan negatif sayılarda *FIX*'den bir eksik sonuç alınır.
Bağlantılı fonksiyonlar: *CINT, FIX, ROUND*

JOY

JOY[(tamsayı ifadesi)]
10 IF JOY (0) = 8 THEN GOTO 100
FONKSİYON: Belirtilen (tamsayı ifadesinde) [0 ya da 1] joystick'den bitler, seviyesinde sonuç okur.

Bit	Ondalık
0: Yukarı	1
1: Aşağı	2
2: Sol	4
3: Sağ	8
4: Ateş 2	16
5: Ateş 1	32

Bağlantılı fonksiyon: *INKEY*

KEY

KEY (tamsayı ifadesi), *[CHR$ (n) +] (karakter dizisi) [+ CHR$ (n)]*
KEY 140, "RUN" + CHRS (13)
KOMUT: Yeni bir fonksiyon tuşu tanımlaması ayarlar. 32 tane klavye 'genişleme' ka-

rakteri vardır. EK 111 de 128-159 luk bir sıralamayla listesi verilmiştir. Bu karakterlerden biri okunduğunda, onunla bağlantılı 32 karaktere kadar bir karakter dizisine genişletilir. Ancak genişleme karakterinin toplamı 120'yi geçemez. *KEY* emri verilen bir genişleme karakteriyle, bir karakter dizimi arasında bağlantı kurar.

Bağlantılı komutlar: KEY DEF

KEY DEF

KEY DEF (tuş nosu),(tekrar), [(normal) [, (shift hali) [, (kontrol)]

KEY DEF 46,1 63

EMİR: Klavyedeki herhangi bir tuşdan çıkan değeri EK 111 de tanımlandığı gibi değiştirir. Yukardaki örnek *N* tuşunu soru işareti yazmaya değiştiriyor. (Karakter No 63)
Tuşu normal fonksiyonuna geri getirmek için:

KEY DEF 46, 1, 110

Burada 110 numaralı karakter küçük ***n***'dır.
Bağlantılı komutlar: *KEY*

LEFT $

LEFT $ (karakter dizisi ifadesi), (tamsayı ifadesi)

```
10 CLS
20 A$= "AMSTRAD"
30 B$= LEFT $ (A$,3)
40 PRINT B$
```

EKRAN TEMİZLENİR

AMS
Ready

FONKSİYON: Verilen karakter dizisi ifadesinden solundaki karakter, tam sayı ifadesindeki belirtilen konum dahil, çıkartılır. Eğer karakter dizisi gereken boydan daha kısa ise, o halde karakter dizinin tümü geri verilir.
Bağlantılı fonksiyonlar: ***Mid$, RİGHT $***

LEN

LEN [(karakter dizisi ifadesi)]

A$ = "AMSTRAD" : PRINT LEN (A$)

FONKSİYON: Karakter dizi ifadesindeki bütün karakterlerin, aralar dahil, numaralarını karşılayan bir sayı (numara) geri verir. Karakter dizinin uzunluğunu hesaplar.

LET

LET (mantiki ifade)

LET X = 1000

KOMUT: İlk *BASIC*'lerden bir kalıntı. Eski *BASIC* yetiştirme el kitaplarıyla uyum sağlamanın dışında bir yarar yoktur. Yukarıdaki örnek yalnızca şöyle de yazılabilirdi:

X = 100

LINE INPUT

LINE INPUT [(# (kanal ifadesi),] [;] [tırnak içindeki ifade;]
(karakter dizi değişkeni)
LINE INPUT [(# kanal ifadesi),] [;] [tırnak içindeki ifade]
(karakter dizi değişkeni)

LINE INPUT A$
LINE INPUT "NAME"; N$

KOMUT: Belirtilen kanaldan bütün bir satırı okur. *LİNE INPUT* dan sonraki noktalı virgül satır başını önler. Kanal ifadesinin ilk durumu, her zamanki gibi, = *0*
bağlantılı komutlar: *READ, INPUT, INKEY $, INPUT$*

LIST

LIST [(satır numarası sıralaması] [, # kanal ifadesi)]
LIST 10Ø - 1ØØØ, # 1
EMİR: Program satırlarını verilen bir kanala list eder. Ø ilk değerdir ve ekrandır, 8 yazıcıdır. *LIST, [ESC]* tuşuna bir kez basarak yeniden başlatılır. *[ESC]* iki kez basarak *BASIC*'ı direkt *MOD*'a geri getirir.Programlar önceden tanımlarlar. Pencerelere *(windows)* liste alınabilir . Böylece bütün ekran alanına yayılmadan düzeltme yapılabilir. *LİST* programın başından verilen bir noktaya kadar ya da belirtilen satır numarasından programın sonuna kadar yapılabilir. Bunu yapmak için (satır numarası sıralamasında) birinci ya da sonuncu numaralar yazılmaz. Örneğin:

LIST- 2ØØ ya da LIST 3Ø-

LOAD

LOAD (kütük ismi) [, adres ifadesi)]
LOAD "PROG"
KOMUT: Kasetteki bir *BASIC* programını hafızaya okumak. Hafızada program varsa okunan son terimi alır. Ya da seçenekli adres ifadesi kullanılıyorsa, Binary bir kütüğü hafızaya yükler. (2. bölüme bakın)

LOCATE

LOCATE [# (kanal ifadesi),] (x koord.), (y koordinat)

```
1Ø MODE 1
2Ø LOCATE 2Ø, 12
3Ø PRINT CHR$ (249)
```

KOMUT: Metin kürsörünü ilgili pencerede belirtilen koordinata taşır. Pencerede sol üst köşenin koordinatları 1.1 şeklindedir.
Bağlantılı komutlar: *WINDOW*

LOG

LOG [(sayısal ifade)]
? LOG (9999)
9. 21Ø24Ø97
FONKSİYON: Sayısal ifadenin doğal logarıtmasını hesaplar ve sonuç tam sayı olsa dahi, sonucu reel sayı olarak verir.
Bağlantılı fonksiyonlar: *EXP, LOG 10*

LOG 1Ø

LOG 1Ø [(sayısal ifade)]
? LOG 1Ø (9999)
3.99995657

FONKSİYON: (sayısal ifade)nin 1Ø tabanında logarıtmasını hesaplar ve (sayısal ifade) tam sayı verebilse de sonuç reel bir sayıdır.
Bağlantılı tuş sözcükleri: *EXP, LOG*

LOWER $

LOWER $ (karakter dizisi ifadesi)

A$ = "AMSTRAD" : PRINT LOWER $ (A$)
amstrad
FONKSİYON; Karakter dizisi ifadesinde büyük harfleri küçük harfe çevirir. Bir girişi işlemden geçirirken cevapların büyük harf, küçük harf karışık geldiği yanıtlarda yararlıdır.
Bağlantılı fonksiyon: *UPPER $*

MAX

MAX [(sayısal ifade) listesi]
10 n = 66
20 PRINT MAX (1, n, 3, 6, 4, 3)
FONKSİYON: Sayısal ifade, listesindeki en büyük değeri bulur.
Bağlantılı fonksiyon : *MIN*

MEMORY

MEMORY (adres ifadesi)
MEMORY &2ØAA
KOMUT: BASIC hafıza parametlerini, en yüksek byt adresine ayarlayarak eldeki hafıza miktarını değiştirmek için yeniden ayarlamakta kullanılır. *HIMEM* komutunun tanımlanmasına bakın. Hafızayı yoklamak için *FRE* emrini kullanın.
Bağlantılı komutlar: *HIMEM, FRE*

MERGE

MERGE [(kütük ismi)]
MERGE "PLAN"
KOMUT: Kasetteki bir programı hafızadaki programa ekler. Bir programın içeriğini hafızadaki son programa ekler. Eğer program ismi bildirilmezse *BASIC* teyp de rastladığı ilk geçerli programı birleştirmeye kalkışır. Eğer program isminin ilk karekteri ise okuma işleminde kasetten çıkan olağan mesajları önler.

Bir programı hafızaya, programın üstüne yazmadan birleştirmek istiyorsanız, o halde en son programın satırlarını *RENUM* ile gelen programın satır numaralarının üstünde bir sıralamada yeniden numaralandırmalısınız.

Bütün değişkenler, kullanma forksiyonları ve açık kütükler çıkartılır..... *ON ERROR GOTO* kapatılır ve *DEFINT, DEFREAL DEFSTR* ayarları yeniden ayarlanır. Korunmuş kütükler birleşmez.

Bağlantılı KOMUTLAR: *LOAD, CHAIN, CHAIN MERGE*

MID $

MID $ [(karekter dizisi), (tam sayı ifadesi) [, (tam sayı, ifadesi)]

A$ = "AMSTRAD": PRİNT MID$ (A$, 1,3)
AMS.
PRİNT MİD$ (A$, 3,2)
ST.

FONKSİYON: MID$ bir karekter dizisinin bir parçasını (bir alt dizi) belirtir. Birinci (tamsayı ifadesi) alt karekter dizisindeki ilk karekterin konumunu belirtir.

İkinci (tamsayı ifadesi) alınacak alt karekter dizisinin boyunu belirtir. Eğer konulmazsa, bu esas karekter dizisinin sonuna kadar uzar.

Bağlantılı fonksiyonlar: *LEFT $, RIGHT $*

MIN

MIN (sayısal ifade listesi)
PRINT MIN (3, 6, 2. 999, 8, 9)
2.999
FONKSİYON : MAX'ın tersi
Bağlantılı fonksiyon: *MAX*

MODE

MODE (tamsayı ifadesi)
MODE 1
KOMUT: Ekran *MOD*'unu değiştirmek (Ø, 1 ya da 2) ve ekranı *INK Ø*'a değiştirmek. Bütün grafik *WINDOW* pencereleri ilk konumun gelir ve grafik kürsörleri kendi yoluna döndürür.
Bağlantılı komutlar: *WINDOW, ORIGIN.*

MOVE

MOVE (X koordinatı), (y koordinatı)
MOVE 34, 34
KOMUT: Grafik körsörü mutlak koordinatlarca belirten bir konuma getirmek için kullanılır. *XPOŞ* ve*YPOS* grafik kürörünün son konumu saptamak için kullanılan fonksiyonudur. dur.
Bağlantılı komutlar: *MOVE, PLOT, PLOTR, DRAW, DRAWR, TEST, TESTR, XPOS, YPOS*

NEW

NEW

KOMUT: En son programı ve değişkenleri silmek için kullanılır. *KEY* (Tuş) tanımlamaları kaybolmaz ve görüntü temizlenmez. İlk konumuna getirmeden ve cihazı kapatmadan sonra en geri dönüşü olmayan durumdur. Dikkatle kullanın.

NEXT

NEXT [(değişken) listesi]
FOR n = 1 TO 1000 : NEXT
KOMUT: For çevriminin sonunu sınırlar. *NEXT* emri isimsiz olabilir, ya da uygun *FOR*'a atif yapar. Bu da yukardaki örnekte şöyle olurdu:
NEXT n
Bağlantılı komutlar: *FOR*

ON GOSUB
ON GOTO

On (tamsayı ifadesi) *GOSUB* (satır numarası, listesi)
ON (tamsayı ifadesi) *GOTO* (satır numarası, listesi)

10 ON A GOSUB 100, 200, 300, 400, 500
10 ON ELMA GOTO 1000, 2000, 3000, 4000

KOMUT: Tam sayı ifadesindeki değere göre ilgili alt programa ya da, ilgili satıra gider. Sonuç 1 ise listedeki ilk satır numarası seçilir. 2 ise ikincisi seçilir v.s. Yukarıdaki satır 10'da = 1 olunca 100. satırdaki alt program çağırılırdı. A= 2 satır 200'e dallanmaya neden olurdu. Ve böyle devam eder.

ON BREAK GOSUB

ON BREAK GOSUB (satır numarası)

```
10 ON BREAK GOSUB 40
20 PRİNT "PROGRAM İŞLİYOR" '
30 GOTO 20
40 CLS
50 PRİNT "ESC ye 2 kez basıldı"
60 FOR I=I TO 2000: NEXT
: RETURN
```

KOMUT: ESC tuşuna iki kez basılarak program durdurulmak istendiğinde bir alt program çağırır.

Bağlantılı komutlar: *ON BREAK STOP, RETURN*

ON BREAK STOP

```
ON BREAK STOP
10 ON BREAK GOSUB 40
20 PRİNT "Program işliyor"
30 GOTO 20
40 CLS
50 PRİNT "ESC ye iki kez basıldı"
70 POR I=1 TO 2000 : NEXT
80 ON BREAK STOP : RETURN
```

KOMUT: ON BREAK

ON ERROR GOTO

ON ERROR GOTO (satır numarası)

```
10 ON ERROR GOTO 80
20 INPUT A
30 K= A↑ 100
40 GOTO  20
80 PRİNT "HATA" ; ERR : LİST
```

KOMUT: Bir hata farkedince belirtilen bir satır numarasına git anlamındadır. Buradaki örnekte hata 70. satırda bulunmuş.

Bağlantı komutları: *ERR. ERL, RESUME*

ON SQ GOSUB

ON SQ [(kanal)] *GOSUB (satır numarası)*

```
ON SQ (2) GOSUB 2000
```

KOMUT: Verilen bir ses sırasında boş bir yer olduğunda araya girmeye (ınterrupt) imkan sağlamak içindir. (kanal) aşağıdaki değerlerden birini bildiren bir tamsayı ifadesidir.

1: Kanal A için
2: Kanal B için
3: Kanal C için

Bağlantılı komutlar: *SOUND, SQ*

OPENİN

OPENİN (kütük ismi)

```
100 OPENIN " ! INFORMATION"
```

KOMUT: Bilgisayarın hafızasındaki son program da kullanılacak bilgi içeren kasetten bir giriş kütüğü açar.

Eğer kütük isminin ilk karekteri ! ise o zaman görüntülenen kaset işlemi mesajları önlenir ve program kasette işleme hazır ilk blokdan okumaya başlar.
Bağlantı komutları: *CLOSEİN, OPEN OUT*

OPENOUT

OPENOUT (kütük ismi)
OPENOUT "!ABC"
KOMUT: Bilgisayarın hafızasındaki son program ile birlikte kullanan veriler için kasete bir çıkış kütüğü açar. Kütük ismi'nin ilk karekteri ! ise görüntülenen kaset işlemi mesajları önlenir ve program verilen isimli kütüğü meydana getiren ilk veri blokunu meydana getirir. Her blok 2K. veri içerir ve 2K yastık bellek doluncaya kadar hiçbir şey yapılmaz. Ya da kütükler *CLOSEOUT* kullanılarak kapatılır.

NOT : NEW emri herhangi bir açık kütüğü fark eder ve veriler kaybolur. Yani yastık bellekteki bilgiler silinir.

Bağlantılı komutlar: *CLOSEOUT, OPENIN*

ORIGIN

ORIGIN (X) (y), [, (sol koord.),(sağ koord.),(üst koord.),(alt koord.)]
10 CLS: BORDER 13
20 ORİGİNE 0, 0, 50, 590, 350, 50
30 DRAW. 540,350
40 GOTO 20

KOMUT: Grafik kürsörünün başlangıç noktasını saptar. Emrin seçenekli kısmı yeni bir grafik *(WINDOW)* pencere ayarlama komutunu içerir. Bu da kullanılan pıxel adresleme tekniğinden dolayı bütün ekran modlarında işleyebilir.

ORIGIN, koordinatları ana pencerenin sol, alt tarafında 0.0 olan noktadadır. Koordinatlar sola ve yukarı doğru büyürler. Grafik planlayıcıları ve organizatörler için EK V1 ya bakın. Eğer ekranın dışında bir konumu belirtilmişse pencere köşeleri o zaman belirtilen konumun yönünde ekranda görülebilen en ufak konumda varsayılırlar.
Bağlantılı komutlar: *WINDOW.*

OUT

OUT (kapı numarası), (tamsayı ifadesi)
OUT & F8F4, 10
KOMUT: Tamsayı ifadesindeki değer (bu 0....255 arasında olmalıdır) adres olarak kapı numarasında belirtilen kapıya gönderir.
Bağlantılı komutlar: *INP, WAIT*

PAPER

PAPER [# (Kanal ifadesi),] (ınk rengi)
10 MODE 0
20 FOR p= 0 TO 15
30 PAPER p: CLS
40 PEN 15-p
50 LOCATE 6, 12: PRINT "PAPER" p
60 FOR t=1 TO 500: NEXT t
70 NEXT p

KOMUT: Karakterler için fon mürekkebini ayarlar. Karakterler metin ekranına yazıldığı

zaman, saydam mod seçilmemişse, karakter yazılmadan önce karakter fonu *PAPER* rengiyle dolar.

Görüntülenebilen renk adedi seçilen *MOD*'a bağlıdır. Eğer belirtilen renk mevcut degilse paper eldeki bir renge döner.

Bağlantılı komutlar: *INK*, *WINDOW*, *PEN*

PEEK

PEEK (adres ifadesi)

```
10 MODE 2
20 İNK 1,0: İNK 0,12 : BORDER 12
30 İNPUT "BAŞLANGIÇ"; B
40 iNPUT "BİTİS" ; E
50 FOR n=B TO E
60 V$= HEX$ (PEEK(n), 2): PRİNT V$; "at"; HEX$ (n,4)
70 NEXT
```

FONKSİYON: Hafızada belirtilen adresde ne olduğuna bakar. Yukardaki yardımcı program CPC 464'ün *RAM*'ını bir gözden geçirmenizi sağlıyor. CPC464 *RAM*'ı (0000-3FFFH) aşağıda ve yukarda (C000-FFFF) olarak okur.
Bağlantılı komut: *POKE*

PEN

PEN [# kanal ifadesi,] (ink rengi)

```
PEN #1,2
```

KOMUT: PEN belli bir ekran kanalında çizgi çizerken kulanılarak rengi ayarlar. Başlangıç durumunda kanal # 0 olan ekranla ilgilidir.
Bağlantılı komutlar: *INK*, *PAPER*

PI

```
PI
PRINT PI
3.14159265
20 MODE 2
30 RAD
40 INK 1,0
50 INK 0, 12
60 BORDER 9
70 FOR N= 1 TO 200
80 ORIGIN 420,0
90 DRAW 0,200
100 REM
110 DRAW 30*N*SIN(N*PI 4), (SIN(PI/2))*N*SIN(N)
120 NEXT
130 MOVE 0, 200
140 DRAWR 0, 50
150 DRAWR 40, 0
```

FONKSİYON: Bir çemberin çevresiyle çapı arasındaki oran. En yakın makina temsili (3.141592653468251). Yukarıdaki gibi grafik programlarında çok kullanılır.
Bağlantılı fonksiyonlar: *DEG*, *RAD*

PLOT

PLOT (X koordinatı), (y koordinatı) [, (ink rengi)]

```
10 MODE 2: PRINT 4 sayı verin
20 PRINT "Enter X origin (0-639) Y origin (0-399), yarıçap ve açı adımı" : INPUT
x, y, r, s,

30 ORIGIN x,y
40 FOR angle = 1 to 360 STEP s
50 XPOINT = r*COS (angle)
60 YPOINT = R*SIN (angle)
70 PLOT XPOINT, YPOINT
74 REM MOVE 0,0
75 REM DRAW XPOINT, YPOİNT
80 NEXT
```

KOMUT: İlk yanıt olarak 320, 200, 20,1 deneyin. *PLOT*, *DRAW* ile aynıdır.Şu farkla ki, yalnız varışdaki pixel yazılır. Yukarıdaki satır 75'den *REM*'i kaldırıp 70. satıra işlem yapmak için *REM* koyun. (74. satırda çemberi doldurmak için *REM*'i kaldırın)
Dikkat ederseniz bu işlem çevrenin etrafında tekrar tekrar dönerek çemberi dolduruyor. Unutmayın ki buradaki hesaplar radian modunda yapıldı, bu yüzden her adımdaki açı bir dereceden oldukça fazla.*25 DEG* emrini girin ve tekrar çalıştırın.
Bağlantılı komutlar: *DRAW*, *DRAWR*, *PLOTR*, *MOVE*, *MOVER*, *ORIGIN*, *TEST*, *XPOS*, *YPOS*

PLOTR

PLOTR (x koordinat), (y koordinat) [(ink rengi)]

```
10 MODE 2: PRİNT "4 SAYI VERİN"
20 PRİNT "X origin (0-639), y origin (0-399), yarıcap, acı adımı" : İNPUT x,y,r,s.
30 ORIGIN x,y
40 FOR angle = 1 to 360 STEP s
50 XPOINT = r*COS (angle)
60 YPOINT = R*SIN (angle)
70 PLOTR XPOINT, YPOINT
80 NEXT: GOTO 40
```

KOMUT: Yanıt olarak 320, 0, 20, 1 deneyin. *PLOTR*, *DRAWR* ile aynıdır , bir tek fark yalnızca varış pıxelinin yazılmasıdır.
Bağlantılı komutlar: *DRAW*, *DRAWR*, *PLOT*, *MOVE*, *MOVER*, *ORIGIN*, *TEST*, *TESTR*, *XPOS*, *YPOS*

POKE

POKE (adres ifadesi), (tamsayı ifadesi)

```
POKE & 00 FF, 10
```

KOMUT: Cihazın hafızasına direkt ulaşımı sağlar ve bilirtilen adrese 0.....255 arası tamsayı ifadesini yükler. Acemiler tarafından kullanılırsa sistemi kolaylıkla bozabilir.
Bağlantılı fonksiyon: *PEEK*

POS

POS [(# (kanal ifadesi)]

```
PRINT POS (#0)
1
```

FONKSİYON: Belli bir kanalın en son konumunu ortaya çıkartır.Bu örnekte kanal ifadesi

için bir ilk değer yoktur ve onu eksik bırakmak ***syntax error*** mesajıyla sonuçlanır.
Ekran: Metin kürsörünün X koordinatını son window başlangıcına göreceli olarak verir. Bu sol üst köşe 1,1 olarak temsil edilir.

Yazıcı: Carriage konumunu verir. Burada 1 sol kenardır.
ASC11 referans numaraları 314 den yüksek olan bütün karekterler dahildir.
Kaset çıkış kanalı: Yazıcı için olduğu gibidir.
Bağlantılı komutlar: *VPOS*

PRINT

*PRINT [# (kanal ifadesi),] [(liste)][(USING][*bölücü]
KOMUT: PRINT'ın tam açıklaması için bu bölümün sonuna bakın.
Bağlantılı komutlar: *USING, TAB, SPC*

RAD

RAD

RAD
KOMUT: Radıan modunu ayarlar.
Bağlantılı komutlar: *DEG, SIN, COS, TAN, ATN*

RANDOMIZE

RANDOMIZE [(sayısal ifade)]
10 RANDOMIZE 23
20 PRINT RND (6)
KOMUT: BASIC'ın rastgele sayı üreticisi her sayının bir önceki sayıya bağlı olduğu bir rastgele sıra meydana getirir. Aynı sayıdan başlayarak, sıra her zaman aynıdır. *RANDOMIZE*, rastgele sayı üreticisine, yeni bir başlangıç değeri verir. Bu ya verilen bir değer ya da kullanıcı tarafından girilen bir değerdir. *RANDOMIZE TIME* tekrarlanması zor bir sıralama yapar.
Bağlantılı komutlar: *RND*

READ

READ (değişken) listesi

10 FOR I=1 TO 4: READ A
20 PRİNT A*2
30 NEXT
40 DATA- 10, 20, 30, 324
KOMUT: READ, karşılığındaki *DATA* yeri bildirimlerinden sağlanmış sabitler listesinden veri getirir ve otomatik olarak bir sonraki veri bildirimindeki parçaya geçerek bunu değişkenlere verir.
RESTORE okuma noktasını veri bildiriminin başlangıcına döndürür.
DATA komutuna bakın.
Bağlantılı komutlar: DATA, RESTORE

RELEASE

RELEASE (ses kanalları)
RELEASE 4
KOMUT: Bu ses bir ses sırasına konulduğu zaman bu iki *'hold'* durumunu da içerebilir. Bu kanalda belirtilen kanallardan biri *'hold'* durumundaysa *RELEASE* ile serbest bırakılırlar. Ses kanalını tanıtmak için kullanılan ifade bitler seviyesindedir.: A= bit 0, B= bit 1, C= bit 2. Böylece 4 (binary 0100) kaynak C'yi serbest bıkakır.
Bağlantılı komutlar: *SOUND*

REM

REM (satırın geri kalan kısmı)

KOMUT: Program işleyişini hiçbir şekilde etkilemeden programa hatırlatıcılar ve fikirler eklemek için kullanılır. İki nokta üstüste: satır ayırıcıları da yok sayılır. *REM* den sonra satır sonuna kadar herşey yok sayılır. Bir satırda tek ' tırnak karekter dizi ifadesinin değil). *DATA* emrinin dışında: *REM*'e eşittir. *DATA* emrinde tek tırnak karekter dizisinin başladığını gösterir.

REMAIN

REMAIN [(tamsayı ifdase)]
REMAIN (3)
PRINT # 6, REMAIN (0);
FONKSİYON: Belirlenen geciktirme zamanlayıcısını (0.....3 sıralaması arasında) etkisiz kılar. Gecikme zamanlayıcısından kalan zamanı okur. Geciktirme zamanlayıcısı etkisiz olmazsa sıfır gelir.
Bağlantılı komutlar: **AFTER, EVERY**

RENUM

RENUM [(yeni satır numarası)][, (eski satır numarası)] [, (artışı)]
RENUM
RENUM 100,, 100
KONUT: Program satırlarını belirten artışı kullanarak belirten sıralamaya göre yeniden numaralar, (yeni satır numarası) yeni sıranın ilk numarasını verir, verilmezse 10 kabul edilir. (eski satır numarası) *RENUM*'a nerede başlayacağını tanıtır ve eğer konulmazsa ilk program satırı olduğunu varsayar. (artış) satır numaraları arasındaki artışı ayarlar, buda verilmezse 10 kabul edilir.
RENUM bütün *GOSUB, GOTO* ve öteki satır çağırımlarını da uygun olarak düzenler. Komuttan bütün öteki belirleyiciler kaldırılırsa program *RENUM 10*, 10 verilmiş gibi programı yeniden numaralar.
Satır numaları 1.....65535 arasında geçerlidir.

RESTORE

RESTORE [(satır numarası)]
RESTORE 300
10 FOR I=I TO 6
20 READ K$
30 PRİNT K$; " ";
40 NEXT
50 DATA "RESTORE", "KOMUTUNDAN","SONRA" "DATA", "YENİDEN", "OKUNABİLİR"
60 RESTORE: GOTO 10

KOMUT: Data satırından veri okuma sırası seçeneğe bağlı (satır numarasında) belirtilen veri bildiriminin başına götürür. Satır numarası koymamak okumayı birinci veri bildiriminin başına geri getirir.
Bağlantılı komutlar: *READ, DATA*

RESUME

RESUME [(satır numarası)]
ya da *RESUME NEXT*
RESUME 300

KOMUT: bir ON ERROR GOTO emriyle bir hata yakalandığı ve işleme konulduğu zaman RESUME normal program yürütülmesine izin verir. Devamı edilecek satır numarası isteğe bağlı bildirilir.
Bağlantılı komutlar: *ON ERROR GOTO*

RETURN

RETURN
RETURN

KOMUT: Bir alt programın bittiğini belirler. *BASIC* işleme onu çağıran *GOSUB* noktasından devam eder.
Bağlantılı komutlar: *GOSUB, ON GOSUB, ON BREAK GOSUB*

RIGHT $

RIGHT $ [(karakter dizisi ifadesi), (tamsayı ifadesi)]
10 CLS : A$ = "AMSTRAD"
20 PRİNT RİGHT$ (A$, 3)
RUN
EKRAN TEMİZLENİR
RAD
Ready
FONKSİYON: Verilen karakter dizisi ifadesinin sağındaki karakterleri tamsayı ifadesinde belirtilen konum dahil, çıkartır. Eğer karakter dizisi ifadesi gereken uzunluktan kısa ise o halde bütün karakter dizisi ifadesi geri gelir.
Bağlantılı komutlar: *MID$, LEFT$*

RND

RND [(sayısal ifade)]
10 RANDOMIZE 23
20 PRINT RND (6)
FONKSİYON: Rastgele bir sayı getirir. Bu sayı bir sırada bir öncekinin tekrarı olabilir ya da bir sırada ilk sayı olabilir. Yukarıdaki örnekte *RANDOMIZE* emri *RND (6)*'nın her seferinde aynı sayıyı vermesini garantiliyor (0.146940658)
RND (0) en son çıkartılan rastgele sayının bir aynısını verir. Sayısal ifade negatif olduğu yerde çıkartılan sayı sırası tahmin edilir.
Bağlantılı komutlar : *RANDOMIZE*

ROUND

ROUND (sayısal ifade) [,(tamsayı ifadesi)]
10 x=0.123456789
20 FOR r=9 TO 0 STEP -1: PRINT r, ROUND (x,r): NEXT
25 x=123456789
30 FOR r=0 TO-9 STEP -1
40 PRINT r, ROUND (x,r)
50 NEXT
FONKSİYON: Sayısal ifadeyi tam sayı ifadesinde belirtilen şekilde yuvarlar. Eğer tam sayı ifadesi sıfırdan küçükse o zaman değer mutlak bir tamsayıya yuvarlanır. Öyleki tamsayı ifadesinde saptanan sayıda sıfırlar, noktadan önceki yerlere yerleştirilir.
Bağlantılı komutlar: *INT, FIX, CINT, ABS*

RUN

RUN (karakter dizisi ifadesi)

RUN "HOŞGELDİN"
KOMUT: Kasetten bir program yükler ve onu çalıştırmaya başlar. Eğer karakter dizisi ifadesi boşsa "" *BASIC* teyp de rastladığı ilk programı yüklemeye ve çalıştırmaya kalkışır. Eğer karakter dizisindeki ilk karakter ! ise o zaman görüntülenen kaset mesajları önlenir.
NOT: BASIC teypte bir program ismi okunduğunda hemen *NEW* komutunu yerine getirir ve yüklemeye devam eder.
Bağlantılı komut: *LOAD*

RUN

RUN [(satır numarası)]
RUN 100
KOMUT: Son programı belirtilen satır numarasından ya da eğer satır numarası belirtilmezse en başından yürütmeye başlar. Son programın , kullanıcı fonksiyonları ve değişkenler hafızadan silinir.
DEFINT, DEFREAL ve DEFSTR ayarlamaları ilk konumuna getirilir. Bütün kaset kütükleri terk edilir ve yastık çıkış kaybolur.
Bağlantılı komut: *LOAD*

SAVE

SAVE (kütük ismi) [,(kütük tipi)] [, varsa ikili tabanda parametreler]
SAVE 'PROG", P
KOMUT: Hafızadaki programı kütük ismiyle saklar
, *A* Programı ASCII de saklar.
, *P* Programı korunmalı kaydeder.
, *B* Hafızanın bir alanını binary kütük olarak saklar. Örneğin ekranı

Bağlantılı Komutlar: *LOAD, RUN* (kütük ismi), *MERGE, CHAIN, CHAIN MERGE.*

SGN

SGN [(sayısal ifade)]
10 FOR I=- 10 TO 10
20 PRINT I, SGN (1)
30 NEXT

FONKSİYON: Sayısal ifadenin işareteni saptar. Sayısal ifade, sıfırdan küçükse -1 getirilir. (sayısal ifade = 0 ise 0 getirilir. Sayısal ifade sıfırdan büyükse 1 getirir.
Bağlantılı fonksiyonlar: *ABS*

SIN

SIN [(sayısal ifade)]
PRINT SIN (PI/2)
1
FONKSİYON: Sayısal ifadenin sinüsünü saptar. Eğer *Deg* emriyle başka türlü belirtilmediyse kendiliğinden Radian ölçüde hesaplar.
Bağlantılı fonksiyonlar: *COS, TAN, ATN, DEG, RAD.*

SOUND

SOUND (kanal durumu), (ton periodu) [,(süre) [, (volume, [,volume envelopu) [,(tone envelopu) [, gürültü periodu)]]
SOUND 1,200, 1000, 7,0,0,1
KOMUT: CPC464'ün ses özellikleri *BASIC*'ın en karmaşık uzantılarından biridir. 6.bölümde tanıtılmıştır.

Bağlantılı komutlar: *ENV, ENT.*

SPACE $

SPACE$ [(tamsayı ifadesi)]
SPACE$ (190)
FONKSİYON: Verilen uzunlukta bir boş karakter dizisi meydana getirir.
Bağlantılı komutlar: *PRINT, SPC, TAB*

SPEED INK

SPEED INK (tamsayı ifadesi), (tamsayı ifadesi)
5 INK 0,9,12 : INK 1,0, 26
10 BORDER 12,9
20 SPEED INK 50, 20
KOMUT: INK ve *BORDER* emirleri iki renk bağlantısına müsaitdir. Bu durumda *INK* iki renk arasında değişir. Birinci tamsayı ifadesi ilk belirlenen *INK* renginin zamanı ayarlar. İkinci tamsayı ifadesi ikinci rengin zamanı ayarlar. Renkler arasındaki zaman 0.02 saniye üniteleriyle ölçülür. (50 Hz) Renk ve tekrar zamanını seçerken dikkatli olup manyetik etkilerden kaçınmalısınız.
Bağlantılı komutlar: *INK, BORDER.*

SPEED KEY

SPEED KEY (başlama gecikmesi), (tekrar dönemi)
SPEED KEY 20,3
KOMUT: CPC 464 tuşları sürekli basılı tutulursa, veriler başlama gecikme döneminden sonra verilen (tekrar döneminde) kendi kendilerini tekrar ederler. Ayar 0.02 saniyelik ünitelerde 1.....255 arasında yapılır. Başlangıçta hız 10,10'dur.
Çok küçük gecikmeler klavyenin de-bounce altprogramları ile etkileşir. Klavyenin esas okunma hızı bu emirden etkilenmez.
Bütün tuşlar tekrarlanmaz. DEF KEY, emri kullanıcıya belli bir tuşun belirli özelliklerini yeniden tanımlama imkanı verir.
Bağlantılı komutlar: *KEY, DEF*

SPEED WRITE

SPEED WRITE (tamsayı ifadesi)
SPEED WRITE

KOMUT: Kasede iki ayrı hızda yazılabilir. Ya, tamsayı ifadesi 1 olan 2000 baud'da ya da ilk hızı olan ve tamsayı ifadesi 0 olan 1000 baud'da Teypden bir kütük yüklerden CPC464 otomatik olarak hızı kurar ve kullanıcının belirtmesi gerekmez.
Veri kayıt kabiliyeti çok emin olmayan kasette en fazla güvenlilik için 1000 baud hızı salık verilir. Bu konuda 2 bölümdeki bilgilere bakın. Bağlantılı komut: *SAVE*

SQ

SQ [(kanal)]
10 MODE 1
20 FOR N= 20 TO 0 STEP-1
30 PRINT n;
40 SOUND 1, 10+n, 100,7
50 WHILE SQ (1) >125: WEND
60 NEXT

FONKSİYON: SQ fonksiyonu kanaldaki kuyruktaki serbest girişlerin sayısını kontrol etmek için kullanılır. Kontro A 1, B2 ve C de 3'dür. Fonksiyon kanalın aktif olup olmadığını-

eğer değilse kuyruğun başındaki girişin (varsa eğer) niçin beklediğini saptar. Sonuç bitler seviyesindedir.
0, 1, 2 Kuyruktaki serbest giriş adedini gösterir.
3, 4, 5 Kuyruğa başındaki buluşma (randevu) (varsa eğer) durumunu gösterir.
6 Kuyruğun başı tutuluysa ayarlanır.
7 Kanal halen aktifse ayarlanır.
Bağlantılı komutlar: *SOUND, ON SQ GOSUB*

SQR-

SQR [(sayısal ifade)]
PRINT SQR (9)
3
FONKSİYON: Sayısal ifadenin karekökünü verir.

STOP

STOP
300 IF n < 0 THEN STOP
KOMUT: Programın yürütmesini durdurmak ama, *BASIC*'i stop emrinden sonra yeniden başlatılabilecek durumda bırakmak için kullanılır. Bu özellikle programı belli bir noktada düzeltmek için durdurmakta kullanılır.
Bağlantılı komutlar: *CONT, END.*

STR$

STR$ [(sayısal ifade)]
PRINT STR$ (&766)
1894
PRINT STR$ (& x 1010100)
84
FONKSİYON: (sayısal ifade) yi ondalık karakter dizizi haline, *PRINT* emrinde kullanılan formda çevirir.
Bağlantılı komutlar: *VAL, PRINT, HEX$, BİN$*

STRING $

STRING $ [(tamsayı ifadesi), karakter belirticisi)]
PRINT STRING $ (16, ".")
FONKSİYON: Belirtilen bir karakterden oluşan ve belli bir sayıda tekrarlanan bir karakter dizisi ifadesi verir.
Bağlantılı fonksiyon: *SPACE$*

SYMBOL

SYMBOL (karakter numarası),(sıra listesi)
5 MODE 2
10 SYMBOL AFTER 90
20 SYMBOL 93, &80, &40, &20, &10, &8, &4, &2, &1
30 FOR n=1 TO 2000
40 PRINT CHR$ (93);
50 NEXT
60 GOTO 60

KOMUT: SYMBOL emri, daha önce *SYMBOL AFTER* emrinde tanımlanan belli bir karakterin gösterimini yeniden tanımlar. Karakter numarası ASCII den ya da CPC 464'in standart karakter setinden seçilir ve bundan sonraki girişler yeni karakteri 8x8 piksel matrisde tanımlar. A O sırada (bir 8 bit sayı) kullanılarak kağıt rengini gösterir. 1. pixelin en

son ink renginde ayarlanacağını gösterir. EK 11 ve 111'e bakın. Yukarıdaki örnek karakter hücresini diogonal olarak kesen bir ters bölme işareti meydana getirir. Bu tuş köşeli (|) parantezin altındadır.
Bağlantılı komut: *SYMBOL AFTER*

SYMBOL AFTER

SYMBOL AFTER (tamsayı ifadesi)
SYMBOL AFTER 90
KOMUT: Kullanıcı tarafından tanımlanan karakter sayısını *SYMBOL AFTER* emri ayarlar. İlk ayar 240'dir, bu 16 tane kullanıcı tarafından tanımlanan karakter verir. Eğer tamsayı ifadesi 32 ile 32 den 255 kadar olan bütün karakterler yeniden tanımlanabilir.
SYMBOL AFTER emri kullanıldığında bütün kullanıcı tarafından tanımlanan karakterler ilk konumuna gelir.
Bağlantılı komut: *SYMBOL*

TAG

TAG [#(kanal ifadesi)]
10 MODE 2
11 BORDER 9
14 INK 0, 12
15 INK 1,0
20 FOR n= 1 TO 100
30 MOVE 200+n, 320+n
40 TAG
50 IF n < 70 GOTO 60 ELSE 70
60 PRINT "Hello";: GOTO 80
70 PRINT "Hoşçakal";
80 NEXT
90 GOTO 20

KOMUT: Belli bir kanala gönderilen metin grafik kürsörü konumunda yazılmak için yeniden yönlendirebilir. Bu metin ve sembollerin grafikle karışmasına imkan sağlar. Eğer kanal ifadesi konulmazsa ilk değer olarak O dır. Karakter hücresinin sol üstü grafik kürsörüne bağlanmıştır ve eğer *PRINT* bildiriminden sonra; noktalı virgül varsa, yazmayan kontrol tuşları (ör: yeni satır girmek) görüntülenir.
Bağlantılı komut: *TAGOFF*

TAGOFF

TAGOFF [# (kanal ifadesi)]
TAGOFF # 0

KOMUT: Belli bir kanalda *TAG*'ı kaldıran ve metni daha önceki metin kürsörü durumunda, *TAG* emrinden önceki noktaya, gönderir.
Bağlantılı komut: *TAG*

TAN

TAN [(sayısal ifade)]
PRINT TAN (45)
FONKSİYON: Sayısal ifadede verilen açıların tanjantını hesaplar. Sayısal ifade, --200.000.... + 200.0000 arasında olmalıdır. Aksi söylenmedikçe açılar radyan cinsindedir.
Bağlantılı fonksiyonlar: *COS, SIN, ATN, DEG, RAD*

TEST

TEST [(x koordinatı), (y koordinatı)]

PRINT TEST (300, 300)
FONKSİYON: Belirtilen grafik ekran ifadesindeki ink değerini bildirir.
Bağlantılı komutlar: *TESTR, MOVE, MOVER, PLOT, PLOTR, DRAW, DRAWR*

TESTR

TESTR (x koordinatı), (y koordinatı)
TESTR (5.5)
FONKSİYON: Son grafik kürsörünün yerindeki mürekkep değerini belirten koordinatları da ilave ederek bildirir.
Bağlantılı komutlar: *TEST, MOVE, MOVER, PLOT, PLOTR, DRAW, DRAWR*

TIME

TIME
10 DATOM = INT (TIME/300)
20 TICKER =[(TIME / 300) -DATUM]
30 PRINT TICKER ;
40 GOTO 20
FONKSİYON : Bilgisayar açıldığından beri geçen zamanı, kasetten okuma ya da yazma sürelerini katmadan (çıkartarak) bildirir. Zaman üniteleri 1/300 saniyedir.

TRON
TROFF

TRON
TROFF
KOMUT: BASIC, bir programı yürütürken izleme olanağıda verir. Bunun için programın yürütülmesinden hemen önce 1 satır numaralarını köşe parantez [ ] içinde bildirmek gerekir. *TRON* bu özelliği sağlar. *TROFF* da bunun kapatılmasını sağlar.
Bağlantılı komut: *RUN*

UNIT

UNT [(adres ifadesi)]
PRINT UNT (& FF66)
FONKSİYON: 0.....65535 arasında 16 # bit tamsayıyı değiştirir ve - 32768..... + 32767 arasında bir tamsayı değeri verir.
Bağlantılı fonksiyonlar: INT, FIX, CINT,ROUND

UPPER $

UPPER $ [(karakter dizisi ifadesi)]
PRINT UPPER$ ("amstrad")
AMSTRAD.
FONKSİYON: Karakter dizisi ifadesinde küçük harfleri büyük harfe çevirir.
Bağlantılı fonksiyonlar: *LOWER$*

VAL

VAL [(karakter dizisi ifadesi)]
10 A$ = "7 benim ugurlu sayımdır"
20 PRINT VAL (A$)
FONKSİYON: Bir karakter dizisi ifadesinin baş tarafından 'sayısal ifade' değerini alır. *STR$*'ın tersidir.
Bağlantılı fonksiyonlar: *STR$*

VPOS

VPOS [#(kanal ifadesi)]

PRINT VPOS (# 0)

FONKSİYON: Kanal ifadesi için metin kürsörünün dikey konumunu verir.
Bir zamanladır.
Bağlantılı fonksiyon: *POS*

WAIT

WAIT (kapı numarası),(sayı 1) [, (sayı 2)]

WAIT & FF 34, 20, 25

KOMUT: Belirli bir 1/0 kapısı 0.....255 arasında özel bir değer verinceye kadar operasyonu durdurur. *BASIC* 1/0 kapısını okurken çevrim içine girer. Okunan değer sıfır olmayan bir sonuç oluncaya kadar beklenir.
Sayı 2 ile *XOR*'lanır, sayı 2 ile *AND*'lenir.
Koşul olmazsa *BASIC* bir *WAIT* çevrimine takılır. Yukardaki örneği yazarsanız, bundan kurtulmak için bilgisayarı ilk konumuna getirmeniz gerekir. (reset)
Bağlantılı komutlar: *INP, OUT*

WEND

```
WEND
20 İNPUT "SAAT, DAKİKA, SANİYE"; hour, minute, second
30 CLS: datum = INT (TIME/300)
40 WHILE hour < 13
50 WHILE minute < 60
60 WHILE tick <60
70 tick = (INT(TIME/300)-datum)+ second
80 LOCATE 70,4
90 PRINT =0, USING "# #"; hour,minute, tick
100 WEND
110 Tick = 0
115 second =0
120 minute= minute+1
130 GOTO 30
140 WEND
150 minute = 100
160 hour= hour +1
170 WEND
180 hour = 1
190 GOTO 40
```

KOMUT: Belli bir koşul gelinceye kadar, WHILE/ WEND çevrimi tekrarlanır. Buradaki örnek *WHILE/WEND* çevrimini, bu tür bir yaklaşımla yapılan programların inceliğini göstermek için kullanıyor. Artık buna bir çok değişik saatin özelliklerinin tümü eklenebilir.
WEND emri *WHILE* çevrimini bitirir.
Bağlantılı komutlar: *WHILE*

WHILE

WHILE (mantıki ifade)

WHILE DAY < 0

Yukardaki örneğe bakın.
KOMUT: Belli bir koşul yerine gelinceye kadar *WHILE* çevrimi tekrarlanır. *WHILE* emri çevrimin başını tanımlar. *WEND* emri de *WHILE* çevrimini bitirir.
Bağlantılı komutlar: *WEND*

WIDTH

WIDTH (tamsayı ifadesi)

WIDTH 80

KOMUT: BASIC'e yazıcının karakter sayısının ne olduğunu söyler, bu bilgi *BASIC*'ın baskı yaparken gereken yerde satır başı yapmasını sağlar.

Bağlantılı komut: *PRINT, POS*

WINDOW

WINDOW [# (kanal ifadesi),] (sol), (sağ), (üst), (alt)

```
10 MODE 1: BORDER 6: WINDOW 10, 30, 7, 18: PAPER 2: PEN 3: CLS
20 PRİNT CHR$ (143); CHR$ (242); "Bu 1,1 noktası ve metin penceresinde"
30 GOTO: 30
```

KOMUT: Belli bir kanalı için metin penceresine ayarlar.

Bağlantılı komutlar: *ORIGIN*

WINDOW SWAP

WINDOW SWAP (kanal ifadesi), (kanal ifadesi)

WINDOW SWAP 0,2

KOMUT: Metin pencerelerini değiş tokuş eder. Örneğin kanal # 0'a gönderilen *BASIC* mesajları programı geliştirme ve işletmesinin bazı yönlerini vurgulamak için başka bir pencereyle değiştirebilir.

Bağlantılı komutlar: *WINDOW, PEN, PAPER, TAG*

WRITE

WRITE [#(kanal ifadesi),] [(yazı listesi)]

```
WRITE # 2, "MERHABA", 4,5
"MERHABA", 4,5
```

KOMUT: Bir kaç tane ifadenin değerini virgüllerle ayırarak ve karakter dizilerini çifte tırnak içine koyarak belli bir kanala yazar.

Bağlantılı komut: *PRINT*

XPOS

```
XPOS
PRINT XPOS
```

FONKSİYON: Grafik kürsörünün yatay konumunu verir..

Bağlantılı komutlar: *YPOS, MOVE, MOVER, ORIGIN*

YPOS

```
PRINT YPOS
```

FONKSİYON: Grafik kürsörünün dikey konumunu verir.

Bağlantılı komutlar: *XPOS, MOVE, MOVER, ORIGIN*

ZONE

ZONE (tamsayı ifadesi)

```
10 PRINT 1, 2, 3
20 ZONE 19
30 PRINT 4, 5, 6
```

KOMUT: PRINT'de kullanılan baskı bölgesinin enini ilk değer olan 13 den 1...255 arasında yeni bir değere değiştirilir. *NEW, LOAD, CHAİN ve RUN* "kütük ismi" emriyle ilk konumuna getirilir. (reset)

Bağlantılı komutlar: *PRINT, WİDTH*

PRINT

PRINT [# (kanal ifadesi),] [(print listesi)] [(*USING* tanımı)] [(ayırıcı)]

(print listesi) (ifade)dir [(ayırıcı) (print parçası)]*
(print parçası) (ifade)dir.
ya da
SPC [(tamsayı ifadesi)]
TAB [(tamsayı ifadesi)]

Kaset kütüğüne veri yazmak

```
10 OPENOUT "DATA"
20 PRINT = 9, "MERHABA"
30 CLOSEOUT
```

Yazıcıya veri yazmak

```
10 AA= 23 XPI
20 PRINT # 8, USING " # # # # . # # "; AA
30 PRINT # 0 AA
RUN
72256.6311
Ready
```

[Bu arada yazıcıda...]
72256.63 görünür.

KOMUT: Belirtilen format kullanılarak (kanal ifadesine) veri yazmak. Format karakterleri için tabloya bakınız. *AMSTRAD BASIC*, *PRINT* bildiriminin tüm endüstri standartlarını kullanabilir.

Format belirtilmediği yerde *BASIC* serbest format'da basar, bu durumda (print parçasını, izleyen virgül basılan parçayı bir sonraki print zone'un başından başlatır. (normalde 13) Noktalı virgül ifadeleri tek ara kullanarak ayırır.

SPC (tamsayı) belli adetde ara basar, sayısal ifade negatifse değer 0 olur. Bu ifade cihazın eninden (ekran, yazıcı vs.) daha genişse o zaman sığacak şekilde azaltılır. *SPC*'de virgül var sayılır.

TAB (tamsayı ifadesi) Belli bir print konumuna geçmek için ara basar. Eğer negatifse 1 var sayılır. *TAB* yukarda söylendiği gibi belli bir kanal enine sığmak için küçültülür. Eğer gereken konum son konuma eşit ya da fazlaysa gereken konuma ulaşmak için ara basılır. Eğer azsa satır başı yapılır ve gereken konuma erişmek için aralar izler. *TAB* emrini bu zaman bir noktalı virgülün bitirdiği varsayılır.

TAB (tamsayı ifadesi) Belli bir print konumuna geçmek için ara basar. Eğer negatifse 1 var sayılır. *TAB* yukarda söylendiği gibi belli bir kanal enine sığmak için küçültülür.

Eğer gereken konum son konuma eşit ya da fazlaysa gereken konuma ulaşmak için ara basılır. Eğer azsa satır başı yapılır ve gereken konuma erişmek için aralar izler. *TAB* emrini bu zaman bu noktalı virgül bitirliği varsayıdır.

PRINT USING "(Format belirticileri)"

SAYISAL

Belirtici	olası basamak	karakter	tanımlama	Örnek
#	1	1	Sayısal alan	####
.	0	1	Ondalık nokta	#.#
	0	1	Önden ya da arkadan gelen işaret	+##
+			Pozitif sayılarda olur ama negatif sayılardan önce olmaz	###+
-	0	1	Arkadan gelen işaret negatifse - yazar yoksa boş kalır.	##.##-
**	2	2	Önde yıldız	**###.##
$$	1	2	Dolar işareti öndeki hanenin başına konur	$$##.##
**$	2	3	Yıldız ve dolar işareti	**$#.##
,	1	1	Her üç hanede bir virgül kullanılır (yalnız ondalık noktanın solunda)	##,###,#
↑↑↑↑	0	4	Üst formatı sayılar öndeki hanenin sıfır olmıyacağı şekilde sıralanır	#,##↑↑↑

KARAKTER DİZİSİ

!	Yalnızca ilk karakter	!
\ (aralar) \	belirtilen kadar boşluk. Ve arkaya öne birer boşluk daha	\ \
&	Değişebilir uzunluk	&

9. DAHA İLERİ PROGRAMLAMA BİLGİLERİ

Bu bölümün içerdiği konular:

* Metin konumu
* Kontrol karakterleri
* İşletme sistemi
* Interrupt yapıları

9.1 Kürsör yerleri ve kontrol kod uzantıları

Bir çok uygulama programında kürsör en son pencerenin dışına konumlanabilir. Değişik operasyonlar yapılmadan önce kürsörü uygun konuma zorlarlar, bunlar şunlardır:
- Bir karakterin yazılması
- Kürsör blokunun çizilmesi
- İzleyen sayfalardaki listede yıldız işareti, kontrol koduna uyması.

Kürsörü uygun konuma zorlanma yöntemi şöyledir:
a- Kürsör sağ kenarın sağındaysa aşağı doğru bir sonraki satırın en son sütunun soluna getirilir.
b- Kürsör sol kenarın solundaysa yukarı bir sonraki satırın son sütununun en sağına getirilir.
c- Kürsör yukarı kenarın üstündeyse pencere bir satır kaydırılır ve kürsör pencerenin üst kenarının çizgisinin üstüne getirilir.
d- Kürsör alt kenarın altındaysa pencere bir satır yukarı kaydırılır ve kürsör alt çizginin üstüne getirilir.

Metin ve operasyonlar verilen sırada yapılır. Kural dışı kürsör konumları ya sıfır ya da negatif olabilir, bu da ya fazla solda ya da pencerenin üstünde demektir. Ekrana gönderilen 0.....31 arası karakter değerleri (EK 111'e bakın) ekranda bir karakter yapmazlar. (ve tedbirsize uygulanırlarsa bilgisayar işletim sisteminin kilitlenmesine yol açabilirler) ve *KONTROL KODLARI* olarak yorumlanırlar. Bazı kodlar aşağıda yazılı karakterlerin bir ya da daha fazlasının anlamını değiştirir. Bu karakterler kodun parametreleridir. Grafik ekranına gönderilen kontrol karakterleri yazılır. *(TAG* anahtar sözcüğünün tanımlamasına 8. bölümde bakın) ve *(&07 "BEL")* gibi karakterler eğer klavyeden kullanılırlarsa fonksiyonlarına uygun geleneksel bir sembol görüntülerler. (örneğin *[CTRL] G)* ancak kontrol fonksiyonlarını *PRINT CHR$* (& 07) şeklinde bir emir kullanılarak hitab edildiklerinde yürütürler.

* ile işaretlenmiş kodlar yerine getirilmeden önce kürsörü en son pencerede uygun konumuna zorlarlar. Ama kürsörü olmayan konumda da bırakabilirler. Kodlar ve anlamları önce ilk *HEX* değerleriyle (&XX) sonra da ondalık eşdeğerleriyle tanımlanırlar.

GENELLİKLE KLAVYE CTRL TUŞU YOLUYLA ELDE EDİLMEYEN EK KONTROL KARAKTER EMİRLERİ:

Değer	İsim	Parametre	Anlam
&00 0	NUL		Etkisiz. Yok sayılır.
&01 1	SOH	0....255	parametre değeriyle verilen sembolu yaz. Bu 0..31 arasındaki sembollerin görüntülenmesini sağlar.
&02 2	STX		Metin kürsörünü kapat.
&03 3	ETX		Kürsörü al. Dikkat edin. BASIC üste şeffaf yazan bir kürsör kullanıyor. Bu da yalnızca klavye inputu beklerken serbest bırakılabilir.
&04 4	EOT	0...2	Ekran **MOD**'unu ayarla. Parometre **MOD 4** te alınır. **MODE** Emriyle eşdeğerdedir.
&05 5	ENQ	0....255	Parametre karekterini grafik kürsörüne gönder.
&06 6	ACK		Metin ekranını mümkün kıl (ileriki sayfada 15, NAK'a bakın)
&07 7	BEL		Ses bliip yapar. Bunun ses kaynaklarını boşalttığına dikkat edin.
&08 8*	BS		Kürsörü bir karekter geri getir.
&09 9*	TAB		Kürsörü bir karekter ileri getir.
&0A 10*	LF		Kürsörü bir satır aşağıya getir.
&0B 11*	VT		Kürsörü bir satır yukarı getir.
&0C 12*	FF		Metin penceresini temizle ve kürsürü üst sol köşeye getir. Bu CLS emriyle eşdeğerdir.
&0D 13W*	CR		Son bulunduğu satırda kürsörü pencerenin sol kenarına götür.
&0E 14	SO	0....15	Kağıt mürekkebini ayarla. Parametre MOD 16 da alınır. PAPER emrine eşdeğer.
&0F 15	S1	0.....15	Kalem mükekkebini ayarla. Parametre MOD 16. da alınır PEN emrine eşdeğer.
&10 16*	DLE		Son bulunan karakteri sil. Karekter kürsörünü son kağıt mürekkebiyle doldurun.
&11 17*	DC1		Pencerenin sol kenarından başlayarak en son karekter konumuna kadar karakter konumu dahil temizleme. Etkilenen karekter hücrelerini son kağıt rengiyle doldur.
&12 18*	DC2		Son karekter konumundan başlayarak ve onu dahil ederek pencerenin sağ kenarına kadar temizle. Etkilenen hücreleri son kağıt rengiyle doldurun.
&%13 19*	DC3		Pencerenin başlangıcından son karekter konumuna kadar, 0 dahil, temizle. Etkilenen hücreleri son kağıt rengiyle doldurun.
&14 20*	DC4		En son karekter konumundan, 0 dahil pencerenin so-

Değer	İsim	Parametre	Anlam
			nuna kadar temizle. Etkilenen hücreleri son kağıt rengiyle doldurun.
&15 21	NAX		Text ekranını kapat. Bir ACK (&06 6) gönderildikten sonraya kadar ekran gönderilen hiç bir şeye reaksiyon göstermez.
&16 22	SYN	0.....1	Parametre MOD 2 O Saydam seçenek 1'i olası kılar.
&17 23	ETB	0....3	Parametre MOD 4 0 normal grafik mürekkep Modunu ayarlar. 1 "XOR"' 2 'AND'" 3 'OR"'
&18 24	CAN		Kağıt ve kalem mürekkeplerini değiş tokuş yap.
&19 25	EM	0.....255	Kullanıcı tarafından tanımlanabilen karakterler için
		0.....255	Matris ayarla.
		0.... 255	*SYMBOL* emriyle eşdeğer.
		0.... 255	Dokuz parametre alır. İlk parametre karekterin ayarla-
		0.....255	nacak matrisini belirtir. Sonraki sekiz matrisi belirtir. İlk
		0.... 255	byte'nin en yüksek biti karekter hücresinin sol üst
		0.....255	pixelini karşılar., son byte'nin en az alçak bit'i karekter
		0.... 255	hücresinin sağ alt pixelini karşılar.
		0.....255	
&1A 26	SUB	1..80 ²	*WINDOW* emrine eşdeğer pencere ayarlar
		1..80	İlk iki parametreler pencerenin sağ
		1..25	ve sol kenarlarını belirtir. -daha küçük
		1..25	olan değer sol kenar kabul edilir, daha büyüğü sağ. İkinci iki parametreler üst ve alt kenarlar olarak kabul edilir, daha küçük değer üst daha büyük değer alt kenar kabul edilir.
&1B 27	ESC		Etkisiz yok sayılır.
&1C 28	FS	0...15	Mürekkebi bir çift renge ayarlar..
		0...31	INK emrine eşdeğerdir. Birinci parametre
		0...31	(MOD 16) mürekkebi belirtir, sonraki ikisi (MOD 32) gereken renkleri.
&1D 29	GS	0...31	Borderl bir çift renge ayarlar.
		0...31	*BORDER* emriyle eşdeğer. İki parametre (MOD 32) INK rengi belirtir.
&1E 30	RS		Kürsörü pencerenin sol üst köşesine götür.
&1F 31	US	1...80	Son pencerede kürsörü belli bir konuma götür.
		1.....25	*LOCATE* emriyle eşdeğer. İlk parametre getirilecek sütunu, ikinci parametre satırı verir.

9.2 Cihaz işletim sistemi

CPC 464'ün house keeping karmaşık işletme sistemi tarafından sağlanmıştır. İşletme sistemi bilgisayarın içinden giriş de çıkışa kadar "trafiği idare eder".

Öncelikle donatımı BASIC yorumlayıcısı ile ilişkiye sokar.

Örneğin yanar söner mürekkep fonksiyonu, burada BASIC parametreleri geçer, ve OS görevi yapar. Bir bölümü gerekenin ne olduğunu saptar öteki bölümde bu olayların zamanlamasını saptar.

Cihaz işletme sistemine genellikle "firmware" denir ve bu BASIC'de emirlerce çağırılan makina dili alt programları içerir.

BASIC kullanırken CPC 464'ün operasyonunu kitlemek neredeyse imkansızken (***BREAK ON GOSUB*** emri haricinde) makina kodu ile yalnızca bilgisayarı ilk konumuna getirmekdan başka bir şekilde halledilmeyecek problemler yaratmak nisbeten kolaydır ve bu durumda hafızadaki tüm program ve veriler kaybolur.

Eğer cihazın hafızasında yoklamalara girmeye alt programları çağırmaya niyetleniyorsanız bunlara girişmeden önce programınızı ve listenizi saklayın, yoksa pişman olabilirsiniz. CPC 464'ün geniş "FirmWare" işletme sistemi ileri kullanım rehberinde tanımlanmıştır ve bu kullanma kitabının alanını aşar.

9.3 Interruptlar

8. Bölümde tanımlanan ***AFTER*** ve ***EVERY*** yapısıyla örneklenen bir kaç tane çok yararlı özellikleri içeren işletme sistemi sağlamak için CPC 464 Z80 interrupt yapısından geniş ölçüde yararlanır.

Bu olma zamanlayıcılarından öncelikler şöyledir.

Break ***[ESC] [ESC]***
Zamanlayıcı 3
Zamanlayıcı 2 (ve üç ses kanalı kuyruğu)
Zamanlayıcı 1
Zamanlayıcı 0

Interruptlar, interrupt noktasındaki bütün ana değişken durumlarının sonuçlarını düşündükten sonra dahil edilmelidir. İnterrupt alt programının kendisi esas programdaki değişken durumuyla istenmedik bir etkileşmekten kaçınmalıdır.

Ses kuyruklarının eş öncelikle, bağımsız interruptları vardır. Bir kez ses interrupt'ı başladımı bir başkası tarafından kesilemez. Böylece ses emirleri interruptlarını zamanlama yapısını yukarıda sözü edilen etkilere bağışık değişkenleri paylaşma olanağı sağlıyor.

Bir ses kuyruğunun interrupt'ı mümkün olunca, eğer ses kuyruğu kanalı dolu değilse hemen keser, yoksa bir sonraki ses başlayıp kuyrukda daha fazla yer olmasını bekler. İnterrupt hareketi olayı etkisiz kılar, bu yüzden alt programını eğer daha interruptlar gerekiyorsa, onu tekrar etkili kılması gerekir. Ses çıkartmaya kalkışmak ya da kuyruk durumunu kontrol etmek de herhangi bir interruptu yok eder.

[ESC] [ESC] nin bütün öteki kesmelere önceliği BASIC program operasyonunun programı kaybetmeden durdurabilmesini garantiler, yani (eğer değişik koruma tekniklerinden biriyle program bütünlüğü garantilenmemişse)

9.3 AMSOFT Assembler

Makine kodu kullanarak uzun uzadıya program yapabilmek için bir assembler kullanmak gerekecektir.. AMSOFT Assembler, editer assembler ve monitör programı içerir.

10. Interrupt özellikleri:

Bu bölümün içerdiği konular:

* AFTER
* EVERY
* REMAİN
* Esas saat

Sizin de herhalde farkettiğiniz gibi CPC 464'ün yazılımının en önemli yeniliği BASIC'den interruptları idare edebilme yeteneğidir. bu AMSTRAD'ın bir programın içinden kaç tane operasyonu ard arda fakat ayrı ayrı yapabilmesi anlamına gelir. Bu olanağa bazen çok görevlilik denir ve *AFTER* ve *EVERY* gibi yeni emirlerin uygulanmasıyla kullanılır.

Bu olanak ayrıca Sesin kuyruk ve randevu buluşma olanaklarıyla açıkça ortaya konulmuştur. Zamanlamanın bir yönü bilgisayarın içindeki quartz kontrollü zamanlama sistemi olan esas saate atıf yapılır.. Bu saat bilgisayarda olan zamanlama senkron olaylarını idare eder. Bu arada görüntü ve saatleme işlemlerini taramak da var. Donanımda bir fonksiyon zamanlama bağlantılı olduğundan bu guartz esas saatine bağlı olduğunu izlemek mümkündür.

Yazılımda kullanıma *AFTER* ve *EVERY* emirleriydedir.

Örneğin *AFTER*, siz emir satırında zamanı önceden ayarladıktan sonra programı gösterilen alt programa gönderir ve orada tanımlanan görevi yapar.

10.1 AFTER

CPC 464 bir gerçek zaman saati sürdürür. *AFTER* emri bir BASIC programının alt programları gelecekte bir zamanda çağırılmak üzere ayarlama imkanı verir. Dört geciktirme zamanlayıcısı vardır. Bunların herbirinin bağlantılı alt programları olabilir.

AFTER (tamsayı ifadesi([, (tamsayı ifadesi)] *GOSUB* (satır numarası)

Birinci tamsayı ifadesi, alt programın ne kadar zaman sonra çağırılacağını belirtir. Bu zaman 1/50 saniye ile hesaplanır.

İkinci tamsayı ifadesi dört gecikme zamanlayıcısından hangisinin kullanılacağını belirtir. Bu ifade 0...3 arası bir değer vermelidir. Eğer bu ifade konulmazsa 0 varsayılır.

Belirtilen zaman geçince, programın O konumunda *GOSUB* verilmiş gibi, alt program otomatik olarak çağırılır. Alt programı *RETURN* emri kullanılarak bittiği zaman esas program kesildiği yerden çalışmaya devam eder. Zamanlayıcıların değişik interrupt öncelikleri vardır. 3. zamanlayıcının en yüksek önceliklisi, 0 ise en alçak önceliklidir.

AFTER emri, alt programı ve verilen gecikme zamanlayıcısına bağlantılı zaman yeniden ayarlanarak, heran çıkartılabilir. Gecikme zamanlayıcıları *EVERY* emrindekilerle aynıdır.. Böylece bir *AFTER* emri belirli bir zamanlayıcı için verilmiş. Önceki *EVERY* emrinin önüne geçer, tam tersi de olabilir.

```
10 MODE 1: X = 0
20 AFTER 45 GOSUB 100
30 AFTER 100, 1 GOSUB 200
40 PRINT "EKAKOMP"
50 WHİLE X < 100
60 LOCATE # 1, 30, 1:PRINT #1, X:X=X+1
70 WEND
80 END
100 PRINT "arabirimleri"
110 RETURN
200 PRINT "ve yazılımı"
210 RETURN
```

Ana program tarafından yazılmaya imkan sağlamak için interrupt alt programında kullanılandan bağımsız kürsör konumlanması kullanılacak iki kanal (pencere) kullanımına dikkat edin.

(AFTER) emri satırında sizin önceden ayarladığınız zamandan sonra program gösterilen alt programa gider ve orada tanımlanan görevi yapar.

10.2 EVERY

EVERY emri BASIC programının düzenli aralıklarla alt programları çağırmasına imkan sağlar. Dört geciktirme zamanlayıcısı vardır. Bunların herbirine bağlantılı alt program olabilir. Bunun şekli şudur:

EVERY (tamsayı ifadesi) [, (tamsayı ifadesi)] *GOSUB* (satır numarası)

İlk tamsayı ifadesi bu alt programı çağırması arasında ne kadar bekleneceğini belirtir. Bu zaman 1/50 saniye ile ölçülür.

İkinci tamsayı ifadesi mümkün olan dört gecikme zamanlayıcısından hangisinin kullanılacağını belirtir. Bu ifade 0...3 arasında bir değer vermelidir. Eğer ifade konulmazsa 0 varsayılır.

Belirtilen zaman geçtiğinde, programı 0 durumunda *GOSUB* verilmiş gibi, alt program otomatik olarak çağırılır. Alt program normal *RETURN* emriyle bitince, ana program kesildiği yerden çalışmaya devam eder.

Zamanlayıcıların değişik interrupt öncelikleri vardır. 3. zamanlayıcının en yüksek önceliği 0 zamanlayıcısının de en alçak önceliği vardır. Zamanlayıcı biter bitmez sayım yeniden ayarlanır ve bir sonraki alt program çağrısı için geri sayıma başlar.

EVERY emri, alt programı ve belli bir gecikme zamanlayıcısı ile ilgili zaman yeniden ayarlanarak, her zaman verilebilir. Gecikme zamanlayıcıları *AFTER* emrinde kullanılanlarla aynıdır. Böylece bir *EVERY* emri belli bir zamanlayıcı için daha önce verilen *AFTER* emrinin önüne geçer, tam tersi de olabilir.

```
10 MODE 1: X= 0
20 P100=0: EVERY 10 GOSUB 100
30 P200=0: VERY 12, 1 GOSUB 200
40 PRİNT "AMSOFT"
50 WHİLE X< 200
60 LOCATE # 1,30 1:PRINT = 1,X:X=X+1
70 WEND
80 LOCATE 1,20:END
100 DI:PEN P100: LOCATE 1,2:PRINT "çevre üniteleri": El
105 IF P100=0 THEN P100 # 1 ELSE P100=0
110 RETURN
```

```
200 PEN P200: LOCATE 1,3:PRINT "ve yazılım"
205 IF P200=2 THEN P200=3 ELSE P200=2
210 RETURN
```

Aralarındaki emirler yürütülürken zamanlayıcı interruptlarını etkili ve etkisiz yapan *DI* ve *EI* emirlerinin kullanımına *DİKKAT* edin. bunun interrupt zamanlayıcısı 1'in (yüksek öncelik) zamanlayıcı 0 bir interrupt işlemindeyken (satır 100-110) meydana gelmesi ve böylece *PEN* ya da *PRINT* emrinden öteki *LOCATE* ayarını bozmasını önlemek etkisi vardır.

10.3 REMAIN

Bu fonksiyonun dört geciktirme zamanlayıcısından biri için kalan sayımları verir. Zamanlayıcıyı etkisiz kılar, eğer zaten etkisiz kılınmışsa sıfır verir. Kullanılışı şöyledir:
REMAIN [(tamsayı ifadesi)]

EK I

Olabilir Sanatı

Bilgisayara yeni başlayan kişi genellikle bilgisayarın yetenekleri ve sınırları hakkında izlenim toplamaya yardım edecek birtakım referans noktaları kurmakla başlar. Bu bölüm bilgisayarda ne olduğunu ve bunun niye olduğunu anlatmaya yönelik geniş bir rehberdir. (yol gösterici)

CPC 464'ü almanızın nedeni oyun oynamak için bile olsa bilgisayarın donanım adıyla bilinen bazı yanlarını merak ediyorsunuzdur.

Donanım dediğimiz kaldırıp taşıyabileceğiniz kısımdır. Yani bilgisayar klavye, monitör ve bağlantı kabloları vs. Aslında aşağı yukarı yazılım olmayan her şey donanımdır. Yazılım: Programlar, el kitapları kasete dayalı bilgi gibi şeylerdir.

Bilgisayar davranışının bazı özellikleri donanım sayesinde meydana gelir. Yani TV alıcısında (ya da mönitörde) renkli görüntü gibi. Bundan sonra, donanım yeteneklerinden yararlanmak artık yazılıma kalıyor (Ekranda karekterler değişik şekille özel tanımlanmış karekterler vs.)

Donanım aslında elektron ışınlarını televizyon tüpündeki ekranın içindeki elektriksel parlak yüzeye yöneltilir ve onu ışıklandırır. Yazılım bunu ne zaman yapacağını ve nasıl yapacağın söyleyerek düzenler. Yazılım ayrıca havalanan bir uzay gemisi ya da daha gerçek bir şey, klavyede yazdığınız bir mektubun görünmesi gibi bir etki yaratmak için zamanlama, kontrol ve sıralama ekler.

O halde bir bilgisayarı ötekinden farklı kılan nedir?

Yazılımsız donanımın değeri yoktur. Yazılımsız donanım da aynı derecede değersizdir. Bilgisayar bu ikisinin değişik görevler yapmak üzere biraraya gelmesiyle başlar.
Donanım ve yazılımın etkinliğini değerlendirmek için dikkat edilecek bazı temel noktalar vardır.

Kişisel bilgisayarlar için kabul edilen referans noktaları şunlardır:

1- Ekran görüntü kapasitesi -ekranda farkedilebilecek en küçük parça. Bu birkaç etkenin birleşimidir. Buna programcının emrinde olan değişik renkler, görüntünün açıkca görülebilen birbirinden farklı alanları pixeller de dahildir. Ekran görüntü kapasitesi ekranda görüntülenebilen karekter adedini de içerir.

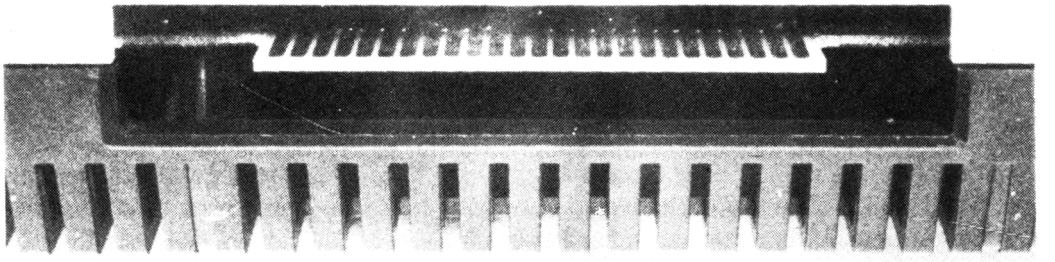
CPC 464 ün aynı fiyattaki bütün öteki bilgisayarlarla karşılaştırıldığında bütün bu açılardan çok daha üstün olduğunu göreceksiniz.

2- BASIC yorumlayıcısı.
Aslında bu bilgisayar program yapmak için donanımın özelliklerini kullanmaya çalışanlara yardımcı olacak bir ***BASIC*** yorumlayıcısı içerir. Cihazınızın içinde kendi başına bir program olan BASIC bilgisayar dili vardır.

Bu son derece geniş ve karmaşık program ***BASIC*** icad edildiğinden bu yana milyonlarca insan yılı değerinde deneyimle geliştirilmiştir. Yeni başlıyanlar için bu amaca uygun sembolik bilgisayar dilidir, ve bir dil gibi değişik lehçeleri yani türleri vardır. CPC 464 tüm ***BASIC*** türleriyle en bağdaşan ve CP/M disk çalıştıran işletme sistemleriyle hazırlanmış

ortak *BASIC* programlarını da çalıştıran bir sistemdir. *BASIC*'ın hızlı kullanımıdır. Başka bir deyişle hesapları büyük bir hızla yapar. Belki de 3 ile 5 in çarpımını 0.05 saniyede yapan bir cihazla 0.075 saniyede yapan bir cihaz arasındaki fark sizi pek ilgilendirmiyordur ama ekrana grafik şekiller çizen bir programda binlerce basit hesabın ardarda yapılması gerektiğinde 0.05 ile 0.075 arasındaki fark bayağı büyüktür.
Sık sık 'makina kodu' terimini duyacaksınız. Makina kodu, işlemden geçirileceklerin mikroişlemci diline indirgenmiş şeklidir.

Amstrad bilgisayarınızın içindeki BASIC herhangi bir bilgisayar da bulunabilecek en hızlı en geniş özelliklere sahiptir ve deneyimli bir *BASIC* programcısının doğasında olan yavaşlıkları yorumlayan cihazların doğasında olan yavaşlıkları yenecek çok daha dinamik müzikal ve görsel sonuçlara varması mümkündür



3. Genişleyebilmek.

Çoğu bilgissayarlar ek donanım parçaları eklenmesine ağırlık verirler. Yazıcılar, oyun çubukları disk üniteleri. Ne çelişkidir ki en başarılı ev bilgisayarlarının bazıları genişleme arabirimi, adı altında ek ünitelere basit yazıcı ve oyun çubuğu olarak önce gerek gisterirler. Alıcıların çoğu ilerdeki gereksinmelerini baştan düşünmüş yoksa yazıcı (***centronics compatible*** ve ***joystick***)i kendinde toplayan bir bilgisayar uzun vadede çok daha ucuza gelir.

CPC 464'in sağladığı özellikler arasında ise yerleştirilmiş ***(Centronics*** yazıcı arabirim) iki ***joystick***/oyun çubuğu olanağı stereo ses çıkışı, disk ünitesi çıkışı ek ***ROM*** uzantıları, (RS232) arabirimleri bağlanabilecek geniş içerikli bir genişleme kapısı vardır.

ROM (read only memory) önceden yazılmış programlar içeren hafızadır. Ve silinemez. Bilgisayarınızın içinde size sağlanan ***BASIC*** böyle bir ***ROM***'dur. Başka programların da buna ek olarak sağlanması ya da yerini alması olanağı vardır.

TV oyunları hazır yazılım kartuşları kullanır. Bu kartuş aslında ***ROM***'dur. Güzel plastik bir kap içinde kolay çıkartılmasını sağlayan çok kullanıma müsait bağlaç'ı vardır. Böylece ***ROM*** program bilgisi sağlamak için kaset gibi bir fonksiyon yapar.. Ancak ***ROM*** bilgisayar yüklemeyi tam anlamıyla anında yapabilir. Kasetin birkaç dakika olduğunu düşünürseniz bu farkın büyük bir avantaj olduğu kesindir.

ROM, kasetten veri kayıt gibi bilgi depolayıp sonra çıkartıp bir bilgisayara nakledilmez. Genişleme bilgisayarınızın ilerki yazılım ve çevresindeki gelişmelerden yararlanabilme imkanıdır. CPC 464 sistemi eksiksiz ve mükemmel genişleme yeteneğine sahiptir.

4 SES

Bir bilgisayarın ses özellikleri onun boş bir bisküvi tenekesine hapsedilmiş bir yığın sinek sesi çıkartma ya da herhangi bir elektronik enstrumanın sesini verebilme yeteneğini sağlar. CPC 464 bilgisayarı 3 kanal ve 8 oktav ses kaynağı kullanır. Bunlar genlik ve tonenvelopu tam olarak kontrol edilebilen çok geçerli bir müzik kalitesi üretirler. Bundan başka ses stereo birleşimlere ayrılmıştır. Burada bir kanal sol çıkışı, bir kanal sağ çıkışı sağlar. Bir üçüncüsü de ortadadır.

EK II

Bilgisayarın geçmişine bakış.

Bütün "uzman" endüstriler gibi bilgisayar da kendi terimlerini "basit dilde" anlatımında bir çok kelime gerektirecek karmaşık kavramları iletebilmek için geliştirmiştir. Kendini özel sözcüklerin bilinmezliği arkasına saklayan yalnızca yüksek teknoloji değildir. Hepimiz belli başlı meslek ve sanatların kurduğu özel terimler duvarıyla karşılaşmışızdır. Esas fark şudur ki terimlerde karışıklık, bilgisayarda olduğu gibi sözcüklerin kendilerinden değil, kullanılış biçimlerinden kaynaklanır. Bilgisayar terminolojisine alışan kişilerin çoğu açık seçik sözcükler kullanmak için özel çaba gösterirler. Bunu iletişim karmaşıklığını en aza indirmek için yaparlar.

Bilgisayarcılıkta kullunılan "basit dil" sizi yanıltmasın, bu yazım konusu değil belirli bir ilimdir ve sözcüklerin dizimi dışında iletişim yapısı gayet dolaysız ve açıktır. Bilgisayar öğretmenleri bir programda programcının tam olarak ne demek istediğini analiz etme sanatı geliştirmeye henüz kalkışmadılar.

Böyle söyledik amma bilgisayar programının anlamı ne kadar açık olursa olsun inceliği ya da duyusuzluğu açısından analiz edilecek bir çok yanı yine de vardır ve program yapımına formel bir yaklaşımın üstünde gittikçe daha fazla durulmaktadır.

Bilgisayar ile iletişimin kesinlik ve basitliğini takdir eden bir çok genç insan bilgisayarcılığı hızla anlamaktadır. 10 yaşında avukat bulamazsınız ama 10 yaşında programcı bulabilirsiniz.

BASIC'in temeli:

Gerçekçe bu ev bilgisayarı BASIC olarak bilinen bilgisayar dili sağlar. ***BASIC*** zamanımızda ***"basit dile"*** en yakın programları yazmaya elverişli dildir. ***BASIC*** şu sözcüklerin boş hafrlerinden oluşmuştur. ***Beginners All-purpose Symobolıc Instruction Code*** (Başlangıçdakilere çok amaçlı Sembolik Komut kodu) Artık dilin karmaşıklık derecesi açısından özel bir belirginliği yoktur ve çok karmaşık ve güçlü programlar ***BASIC*** kullanılarak yazılabilir. Ancak bu isim yeni başlayanlara bilgisayar dilinin labirentinde hiç değilse bir başlama noktası verdiği kesindir ve bu dilin belli bir şekilde evrensel olmasında katkıda bulunmuştur.
Buradan itibaren çok kullanılan bilgisayar terimlerinin sözlüğü tanıtılacaktır.

BASICS

BASIC bir bilgisayar dilidir. Bir sürü emri yorumlar sonrada program çalışırken verilerin üstünde işlem yapar. İnsan dilleri 5-8 bin kelime arasında iken BASIC 200 kelime ile idare etmek durumundadır. BASIC kullanırken son derece katı kurallara uymak zorundadır. Sözdizimi kesindir. Bilgisayar ile ***"günlük"*** dilde iletişim kurmaya kalkarsannız soğuk bir cevap alırsınız..

Syntax error

Bu göründüğü kadar sınırlayıcı değildir. Çünkü ***BASIC*** dili birinci olarak sayıları, sayısal verileri idare etmek üzere yapılmıştır. Sözcükler aslında bildik matematik işlemcilerin + / - vs. uzantısıdır.
BASIC'ı yeni öğrenenlerin kavramaları gereken en önemli kavram bilgisayarın yalnızca sayılarla sayısal verilerle çalıştığıdır. Mikro işlemciye verilen bilgi yalnızca sayısal veriler şeklinde verilir.

SAYI LÜTFEN

Bir bilgisayar Shakespeare'in tüm eserlerini depolamak için kullanılsa da bütün sistemde bir tek fark ya da sözcük bulunmaz. Her bilgi önce bilgisayarın bulunabileceği ve sonra da gerektiği gibi idare edebileceği sayılara çevrilir.

BASIC sözcükleri sayılar olarak yorumlar, bunları toplama çıkartma ve Bool cebri kullanarak idare eder. Bu yolla verileri karşılaştırır ve bazı özelliklerine göre seçer. Yani bir sayının ötekinden büyük olup olmadığını kontrol eder ya da tanımlanan bir görevi belli kriterlere şu *YA DA* bu sayının uymasına göre yapar.

Programın ortamı yoluyla bilgisayar her görevi basit *EVET/HAYIR* işlemlerine indirger. Çarpma işlemi bir çok toplama ile yapılır. *BASIC*'e verilen 37×10 komutu 37 in kendisiyle 10 kez toplanmasıyla sonuç verir.

Mikroişlemcinin bir kısmı 10 için sayısal verilerle diğer bir kısmı 37 için doludur. 37 her kendi kendiyle toplanışda 10 dan bir eksilir. Sonuçda 10 lar bitince sonuç mikroişlemcinin bir başka bölümüne çıkış için gider.

Bu işlemlerde ince bir yan görmüyorsanız haklısınız. Bilgisayarcılık hakkında ilk ve en önemli gerçeği öğrendiniz. Bilgisayar esas olarak en basit tekrarlanan işlemleri çok çabuk ve kesin bir doğrulukla yapan bir alettir. Böylece BASIC programdan verilen komutları yorumlar ve bunları CPU'nun idare edebileceği dile çevirir. Bilgisayar mantığı ancak iki bildirimi anlar "evet" ya da "hayır" Bunlar binary notasyonda "1" ve "0"la gösterilirler. Bool cebrinde ise "doğru" ve "yanlış" vardır. "belki" diye bir şey yoktur.

Bu iki belirli bildirim arasında gidip gelme işlemi digital terimin özüdür. Doğal dünyada çoğu işlemler bir "durağan" durumdan diğerine yavaş ve düzgün bir ilerlemeyle geçerler. Yani geçiş iki durum arasında bir çizginin yolunu izleyerek yapılır. İdeal digital bir ortamda ve bir durumdan ötekine bir anda geçilir. Ama yarı iletken bilimin fiziği çok kısa bir gecikme olacağını söyler, buna çoğalma gecikmesi denir. ***(Propagation delay)*** ve işte bu gecikmelerinin birikmesi bilgisayarı işlemlerinde cevap gelmeden önce geçen zamanın sorumlusudur.

Bilgisayarların zaten bir görevi bilmesi , o ilk görevin sonucu üstünde çalışmaya başlamadan önce, çok kısa bir süre beklemesi gerekirdi. Yani yapay bir gecikme verilmesi gerekirdi. Digital süre siyah ya da beyazdır. Geçişdeki gri tonlarının hiç bir anlamı yoktur. Düzgün ilerleme değişik grilerden geçer.

Son yanıt 0 ya da 1 ise "aşağı yukarı" doğru olmasına olanak yoktur. Bilgisayarların bazen sayısal verileri idare ederken yapmış gibi görünmeleri, işlem yapabileceği sayıların sınırlarından dolayı olur. "fazla büyük" verileri yer imkanlarına göre sıkıştırmak gerektiğinde toparlama hataları olur. Örneğinn 999.999.999, 1000.000.000 olur..
Eldeki iki sayının 0 ya da 1 olduğu bir dünyada 1 den öteye nasıl sayılır?

Bitler ve Byteler

Biz ondalık sisteme dayalı sayıları anlamaya alışmışızdır. Burada referans noktası 10 dur. Yani 0 dan 9 a kadar on vardır. (bunu 1-10 olarak kullanmak genelde tercih edilir) Sayıların sıralaması 0 dan 1 e olduğu sistem binary sistemdir ve bu sistemin çalıştığı (ünitlere) birimlere bit'ler denir. (*BI*nary digi*T*)in kısaltılmış halidir. 0 ve 1 in ondalık bağını unutmak ve fonksiyonlarına göre tamamen değişik (işaretler) sembolle kullanmak işimizi daha kolay anlaşılır yapacaktır ve süreklilik olduktan sonra, rahatlıkla kullanılabilir.

Bit'lerle ondalık notasyon arasındaki ilişki kolay anlaşılırdır.

Binary digitlerin en büyük sayısını bildirmek için önüne sıfırlar koyarak sayıyı tam bit sayısı yapmak aslında alışılmış bir şeydir:
ondalık 7
00111 binary
5 bit notasyon kullanılarak
olur.
ya da
ondalık 7
5 bit rotasyon kullanılarak
00111 binary olur.

Binary sistemde figürler, verilen 2 gücünün var olup olmadığını (0) belirtmek için, sütunlarda yalnızca gösterici olarak kabul edilebilir.

$2^0 = 1$

$2^1 = 2 = 2 = (2^0)$

$2^2 = 4 = 2x2 = 2\ (2^1)$

$2^3 = 8 = 2x2x2 = 2(2^2)$

$2^4 = 16 = 2x2x2x2 = 2(2^3)$

Bu durumda sutunlar şöyle görünür.

2^4	2^3	2^2	2^1	2^0
1	0	0	1	1
(16	0	0	2	1) = 19

Binary digit bilgilerinden söz ederken daha kısa bir metod sağlamak için byte terimi kullanılır. Byte 8 bitlik bilgidir. Bu durumda bir byte'da dapolanabilecek en yüksek sayı (binaryy) 11111111. ya da (ondalık) 255 dir. Bu 256 değişikliği ima eder, buna 00000000 da dahildir ve bilgisayar için gayet geçerli verilerdir.

Bilgisayarlar veriler 8 bit çarpımlarıyla idare etmeye yatkındırlar. 256 çok büyük bir sayı değildir, bu yüzden hafızayı iadere edebilmek için geçerli yol bulabilmek için hafızaya hitab etmek metodu sağlamak amacıyla bir byte kullanılır. Bu bir matrisdir. Yatay ve dikey adresleri vardır bunlarla bu matrisin elemanlarının yeri bulunabilir.

0	1	2	3	4	5	6	7	8	9
1									
2									
3									
4									
5				1		1			
6									
7									
8									
9									

Bu Matris, 1 den 9 a kadar adres numalararı kullanılarak (10x10)' kadar bilgi parçasının yerini bulabilir 4.5 konumundaki parça 1 dir. 6.5 deki de 1 dir.

Böylece 256 x 256 lik bir binary matris, yatay ve dikey kolonlarına 8 bit adresler kullanarak 65.536 yeri tek tek idare edilebilir. Böylece bizim '0' ve '1' imiz 65.536 tek elemandan birini tanıyabilme yeteneğine çıktılar.

Binary için ikinci ifade yolu kilobyte'dır.(kByte da kısaca K) Bu 1024 byte dir. 1024, kilo terimine daha alışkın olduğumuz ondalık kullanımına en yakın binary çarpımıdır. Bu da bilgisayar niçin '64K'lık hafızası var diye tanımlandığını açıklıyor. Aslında hafızası 65.536 bytelik (64 x 1024) tür.

Çok şükür ki BASIC yorumcusu bütün gereken değiştirmeleri sizin yerinize yapar ve binary sistemi tam anlamadan da çok verimli bir programcı olmak mümkündür. Ama binary'in anlamını bilmek, bilgisayar ilminde çalışmalarınız ilerledikçe kaçınılmaz olarak ortaya çıkan bir çok 'sihirli' ya da anlamlı sayıyı farketmenize yardımcı olacaktır.
Binary'nin ve değişik anlamlı sayının 255, 1024 vs idraki için çaba sarfetmeye değer. Çünkü bunların gelecekte de bilgisayarların can alıcı noktası olmaya devam edeceği bellidir. Yalnızca iki durumla çalışmakdan gelen kesinlik ve basitlik başka bir sayı temelinin doğuracağı büyük karmaşıklıktan üstün olacağı kesindir.

ANCAK...

Basit ve ince olsa da binary notasyon, bir bakışta kolaylıkla okunmadığı için biraz dolaylı ve hataya yatkındır. Binary'in programlara kısa yazı (steno) olarak görev yapar bir kaç tane bağlantılı sayma sistemi vardır. Mikrobilgisayarlarda çok kullanılan böyle bir sayı sistemine Hex denir. (hexa decimal'in kısa hali)
Burada sayı 16 tabanlıdır ve tek karekterle 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
Hexadecimal sistemi bir byte in sekiz bitini dört bitlik bloklara ayırabilir.
15 dört bit sayı olduğuna göre: 1111 binary. Birinci blok '15'in tam ünite sayısını belirtir, ikinci de 'kalan'ı belirtir. Ve işte burada binary'nin inceliği ortaya çıkmaya başlar.
Binary rotasyonu tanıtan tabloyu yeniden düşünülür.

Decimal	Binary	CPC464-ese	Hexadecimal
0	0	fl	0
1	1	*	1
2	10	*fl	2
3	11	**	3
4	100	*flfl	4
5	101	*fl*	5
6	110	**fl	6
7	111	***	7
8	1000	*flflfl	8
9	1001	*flfl*	9
10	1010	*fl*fl	A
11	1011	*fl**	B
12	1100	**flfl	C
13	1101	**fl*	D
14	1110	***fl	E
15	1111	****	F
16	10000	*flflflfl	10

Bir 8 bit sayı 11010110 Kendi içinde bölünebilir ve 4 bit sayı olarak düşünülebilir. Bunlar nibbles, Hex D6 olarak bilinirler. Bu kitabda hex tabanlı sayılar & sembolüyle tanıtılacaktır. Örneğin 06 ve bu da Assemby dil programı çoğu programcının doğrudan makina kodu programlarına en yakın gelebildiği dil programıdır. Çünkü assembly dil programı programın esas makine kodu sayılarını belirtmek için basit harf mnemonics kullanmasına izin verir.

HEX kullanırken, son sayıdaki 16 adedini alabilmek için önce birinci hanenin değerini hesaplamanız gerekir, sonra da toplam ondalık eşdeğerini bulmak için, hex notasyonunun ikinci yarısının gösterdiği kalanı ilave etmeniz gerekir. & D6 gibi sayıyı 13 + 6 ya da 136 olarak göstermek isteği olabilir,. ama aslında bu 13*16) + 9 = 214 dür.

'89' gibi ondalık bir sayıyı okurken de işlem aynıdır. Yani (8 x 10) + 9 tek farkı 10 ile çapmanın 16 ile çarpmaktan daha kolay olmasıdır ama biraz çalışma ile 16 ile çarpmak da aynı derecede kolay gelir.

Buraya kadar aklınız fazla karışmadan gelebildinizse bilgisayarın ana prensiplerini kavramak yolunda ilerliyorsunuz demektir, Hatta belki de neden bunca büyütülür bu iş diye merak bile ediyorsunuzdur.

Haklısınız.

Bir bilgisayar çok basit kavram ve fikirleri idare eden bir aletdir. Sadece bu görevleri çok hızlı yapar (bir saniyede milyonlarca kere) bir de hem verilen verileri hem de sonuca varıncaya kadar olan binlerce ana sonucu muazzam bir hatırlatma yeteneği vardır.

Bilgisayarımızın teorisini sürdürmek istiyorsanız, bilgisayarcılık hakkında binlerce kitap bulabilirsiniz. Bazıları aklınızı karıştırır bazıları ise sayı sistemiyle bilgisayarın onları işlemesindeki temel ilişkiyi basit bir yolla anlatır.

EK III

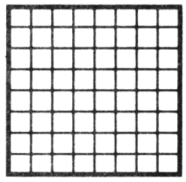
CPC 464 'ün ASC11 karakterleri ve grafik karekterleri.

111. 1 ASC11

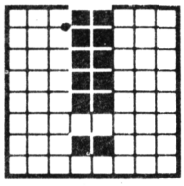
Kolay referans için burada standart ASC11 referans karekter setini ondalık octal ve Hex rotasyonlarıyla birlikte uygun yerde ASC11 kodlarıyla veriyoruz. CPC 464 karekter hücrelerini her biri de ayrıntıyla gösterilmiştir.

DEC	OCTAL	HEX	ASCII
0	000	00	NUL ([CTRL]A)
1	001	01	SOH ([CTRL]B)
2	002	02	STX ([CTRL]C)
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	~

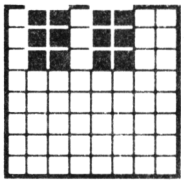
III. 2 CPC464 karekter tablosu



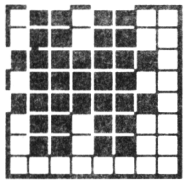
32 &H2C
&X00100000



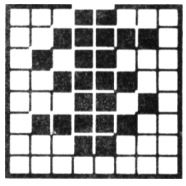
33
&H21
&X00100001



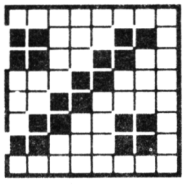
34
&H22
&X00100010



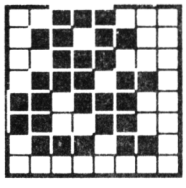
35
&H23
&X00100011



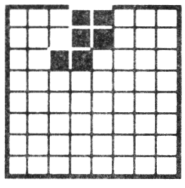
36
&H24
&X00100100



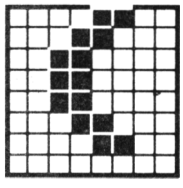
37
&H25
&X00100101



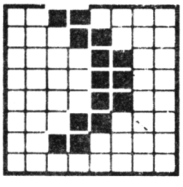
38
&H26
&X00100110



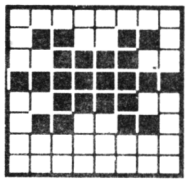
39
&H27
&X00100111



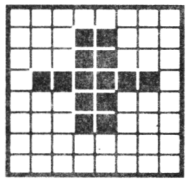
40
&H28
&X00101000



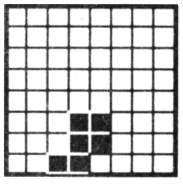
41
&H29
&X00101001



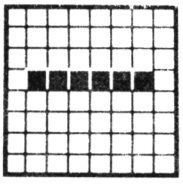
42
&H2A
&X00101010



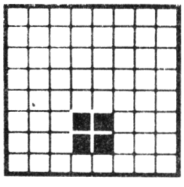
43
&H2B
&X00101011



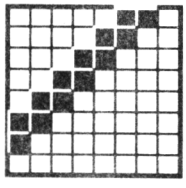
44
&H2C
&X00101100



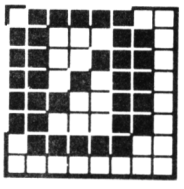
45
&H2D
&X00101101



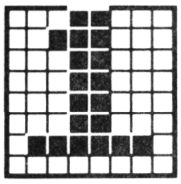
46
&H2E
&X00101110



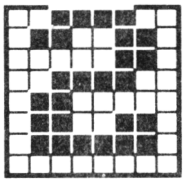
47
&H2F
&X00101111



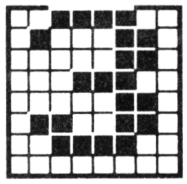
48
&H30
&X00110000



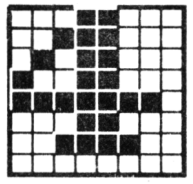
49
&H31
&X00110001



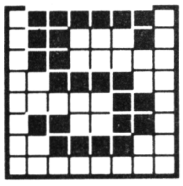
50
&H32
&X00110010



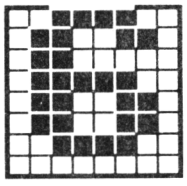
51
&H33
&X00110011



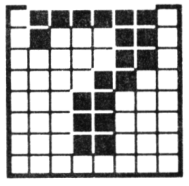
52
&H34
&X00110100



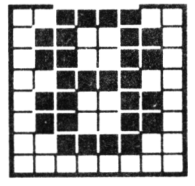
53
&H35
&X00110101



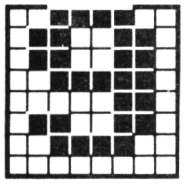
54
&H36
&X00110110



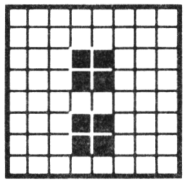
55
&H37
&X00110111



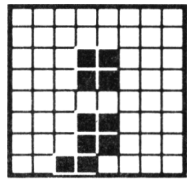
56
&H38
&X00111000



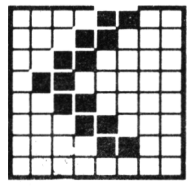
57
&H39
&X00111001



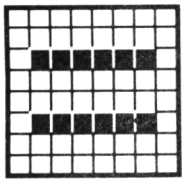
58
&H3A
&X00111010



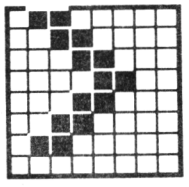
59
&H3B
&X00111011



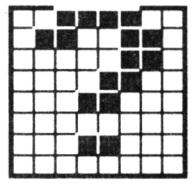
60
&H3C
&X00111100



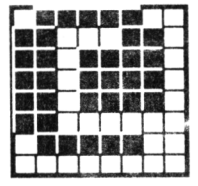
61
&H3D
&X00111101



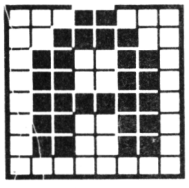
62
&H3E
&X00111110



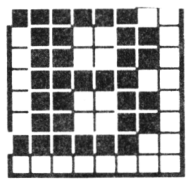
63
&H3F
&X00111111



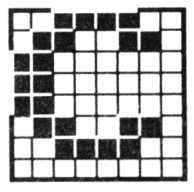
64
&H40
&X01000000



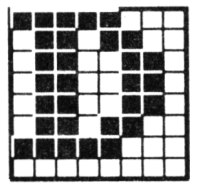
65
&H41
&X01000001



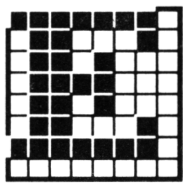
66
&H42
&X01000010



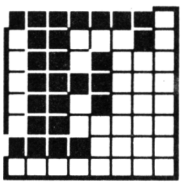
67
&H43
&X01000011



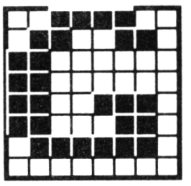
68
&H44
&X01000100



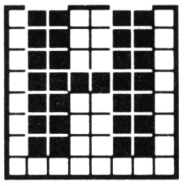
69
&H45
&X01000101



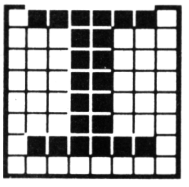
70
&H46
&X01000110



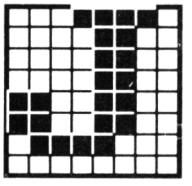
71
&H47
&X01000111



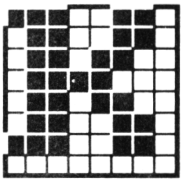
72
&H48
&X01001000



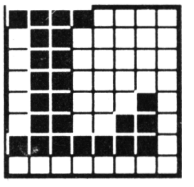
73
&H49
&X01001001



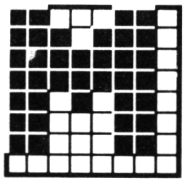
74
&H4A
&X01001010



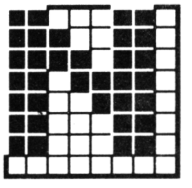
75
&H4B
&X01001011



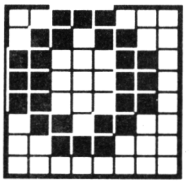
76
&H4C
&X01001100



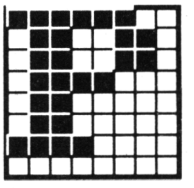
77
&H4D
&X01001101



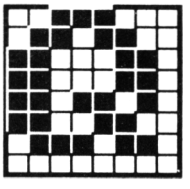
78
&H4E
&X01001110



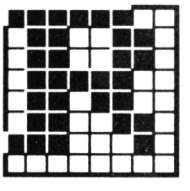
79
&H4F
&X01001111



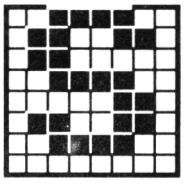
80
&H50
&X01010000



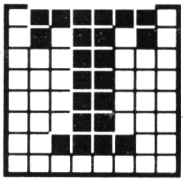
81
&H51
&X01010001



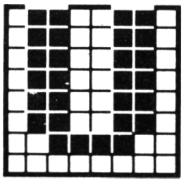
82
&H52
&X01010010



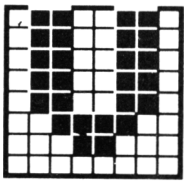
83
&H53
&X01010011



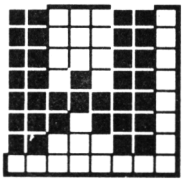
84
&H54
&X01010100



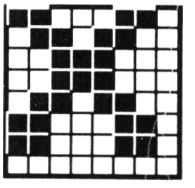
85
&H55
&X01010101



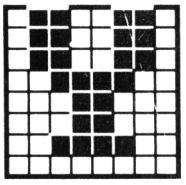
86
&H56
&X01010110



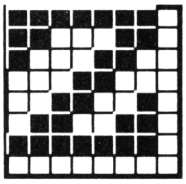
87
&H57
&X01010111



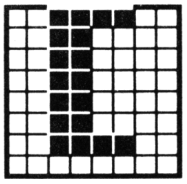
88
&H58
&X01011000



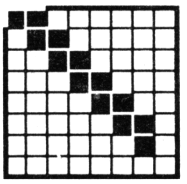
89
&H59
&X01011001



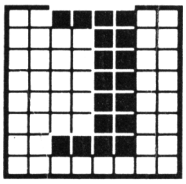
90
&H5A
&X01011010



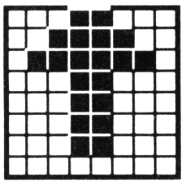
91
&H5B
&X01011011



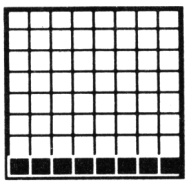
92
&H5C
&X01011100



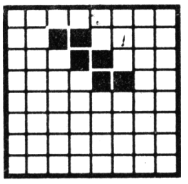
93
&H5D
&X01011101



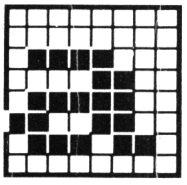
94
&H5E
&X01011110



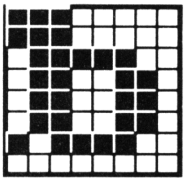
95
&H5F
&X01011111



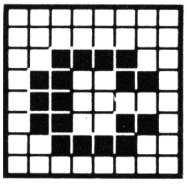
96
&H60
&X01100000



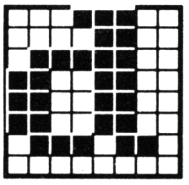
97
&H61
&X01100001



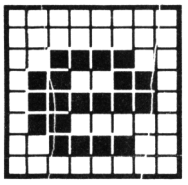
98
&H62
&X01100010



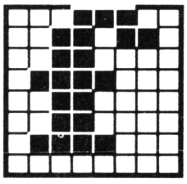
99
&H63
&X01100011



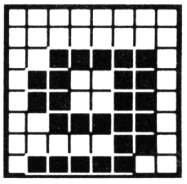
100
&H64
&X01100100



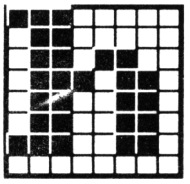
101
&H65
&X01100101



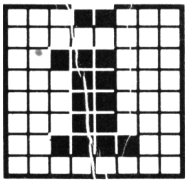
102
&H66
&X01100110



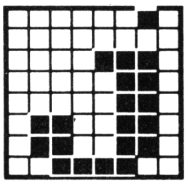
103
&H67
&X01100111



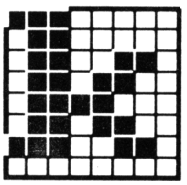
104
&H68
&X01101000



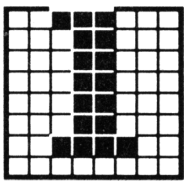
105
&H69
&X01101001



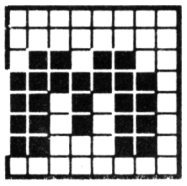
106
&H6A
&X01101010



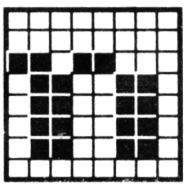
107
&H6B
&X01101011



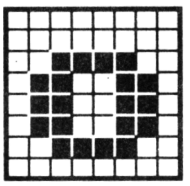
108
&H6C
&X01101100



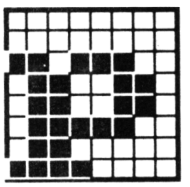
109
&H6D
&X01101101



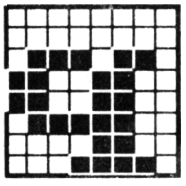
110
&H6E
&X01101110



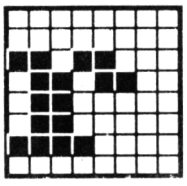
111
&H6F
&X01101111



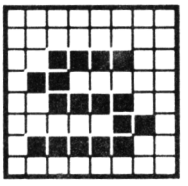
112
&H70
&X01110000



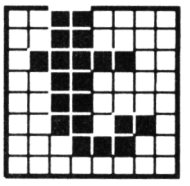
113
&H71
&X01110001



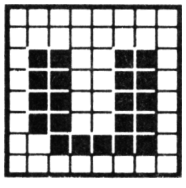
114
&H72
&X01110010



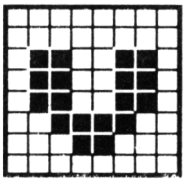
115
&H73
&X01110011



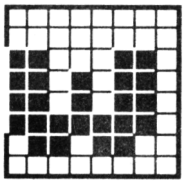
116
&H74
&X01110100



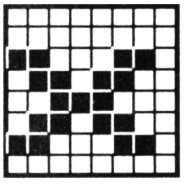
117
&H75
&X01110101



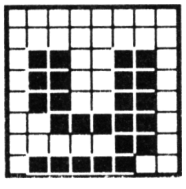
118
&H76
&X01110110



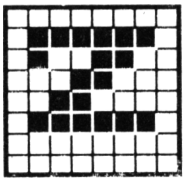
119
&H77
&X01110111



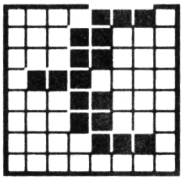
120
&H78
&X01111000



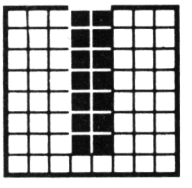
121
&H79
&X01111001



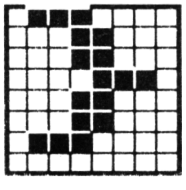
122
&H7A
&X01111010



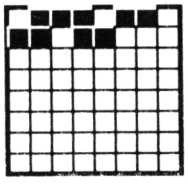
123
&H7B
&X01111011



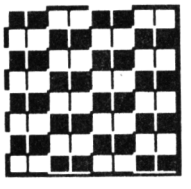
124
&H7C
&X01111100



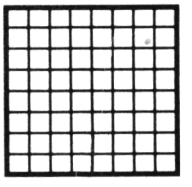
125
&H7D
&X01111101



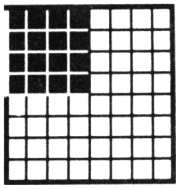
126
&H7E
&X01111110



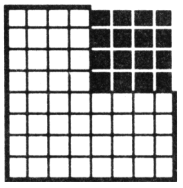
127
&H7F
&X01111111



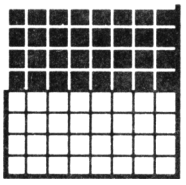
128
&H80
&X10000000



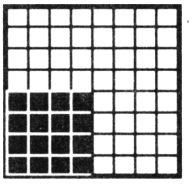
129
&H81
&X10000001



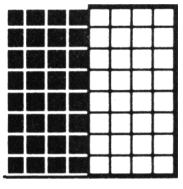
130
&H82
&X10000010



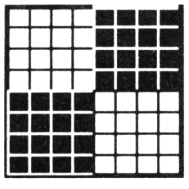
131
&H83
&X10000011



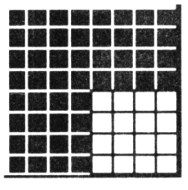
132
&H84
&X10000100



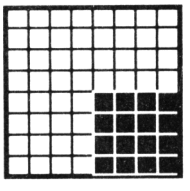
133
&H85
&X10000101



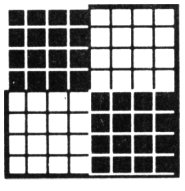
134
&H86
&X10000110



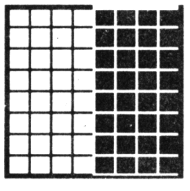
135
&H87
&X10000111



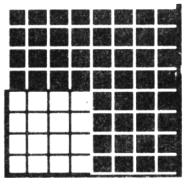
136
&H88
&X10001000



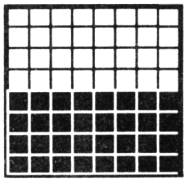
137
&H89
&X10001001



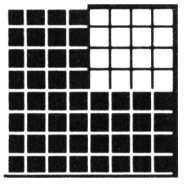
138
&H8A
&X10001010



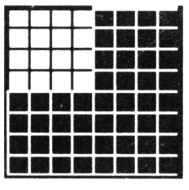
139
&H8B
&X10001011



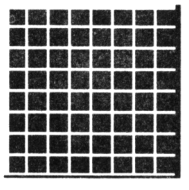
140
&H8C
&X10001100



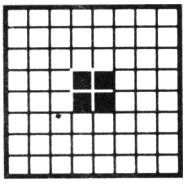
141
&H8D
&X10001101



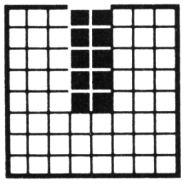
142
&H8E
&X10001110



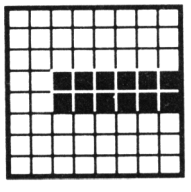
143
&H8F
&X10001111



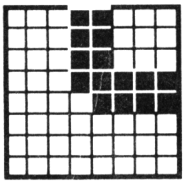
144
&H90
&X10010000



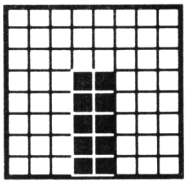
145
&H91
&X10010001



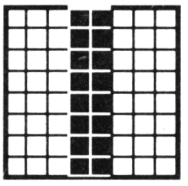
146
&H92
&X10010010



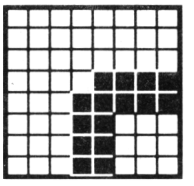
147
&H93
&X10010011



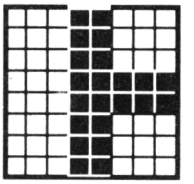
148
&H94
&X10010100



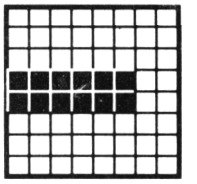
149
&H95
&X10010101



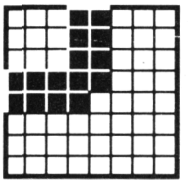
150
&H96
&X10010110



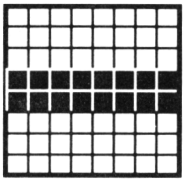
151
&H97
&X10010111



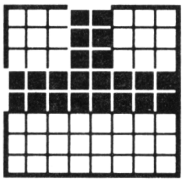
152
&H98
&X10011000



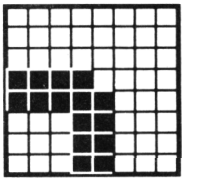
153
&H99
&X10011001



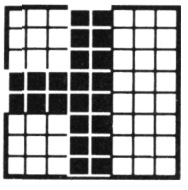
154
&H9A
&X10011010



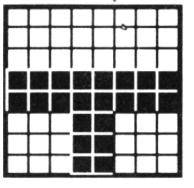
155
&H9B
&X10011011



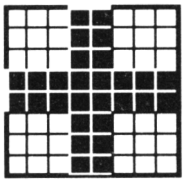
156
&H9C
&X10011100



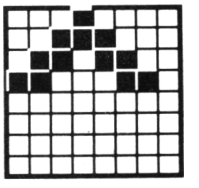
157
&H9D
&X10011101



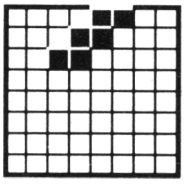
158
&H9E
&X10011110



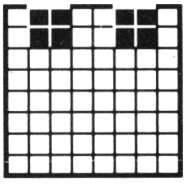
159
&H9F
&X10011111



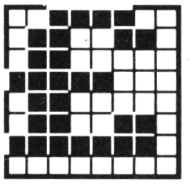
160
&HA0
&X10100000



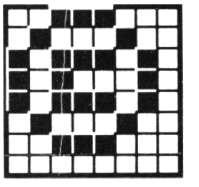
161
&HA1
&X10100001



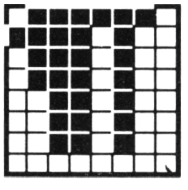
162
&HA2
&X10100010



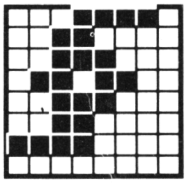
163
&HA3
&X10100011



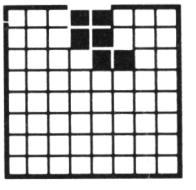
164
&HA4
&X10100100



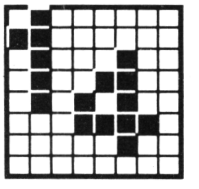
165
&HA5
&X10100101



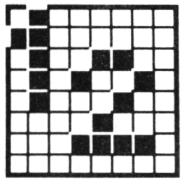
166
&HA6
&X10100110



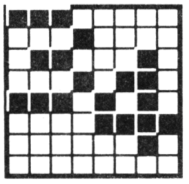
167
&HA7
&X10100111



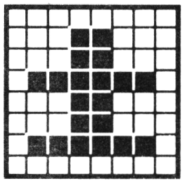
168
&HA8
&X10101000



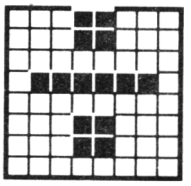
169
&HA9
&X10101001



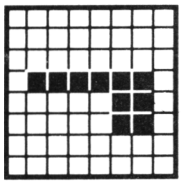
170
&HAA
&X10101010



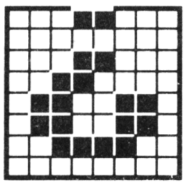
171
&HAB
&X10101011



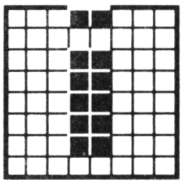
172
&HAC
&X10101100



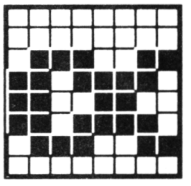
173
&HAD
&X10101101



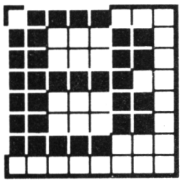
174
&HAE
&X10101110



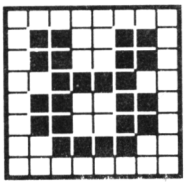
175
&HAF
&X10101111



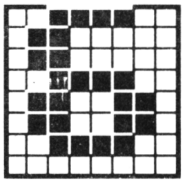
176
&HB0
&X10110000



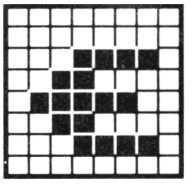
177
&HB1
&X10110001



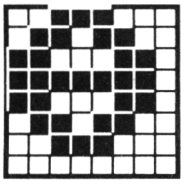
178
&HB2
&X10110010



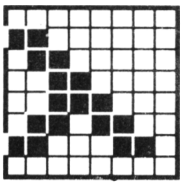
179
&HB3
&X10110011



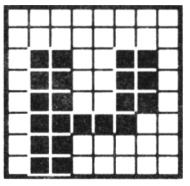
180
&HB4
&X10110100



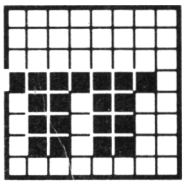
181
&HB5
&X10110101



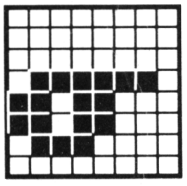
182
&HB6
&X10110110



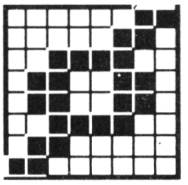
183
&HB7
&X10110111



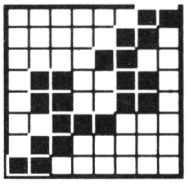
184
&HB8
&X10111000



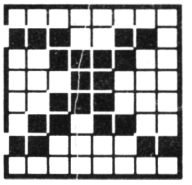
185
&HB9
&X10111001



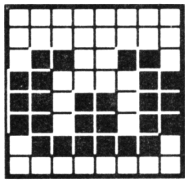
186
&HBA
&X10111010



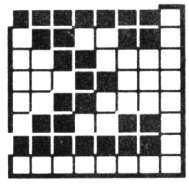
187
&HBB
&X10111011



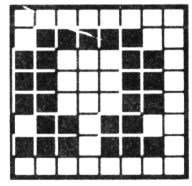
188
&HBC
&X10111100



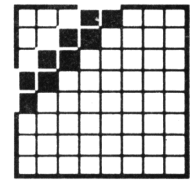
189
&HBD
&X10111101



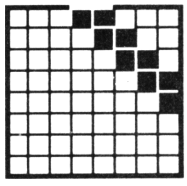
190
&HBE
&X10111110



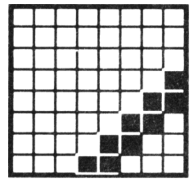
191
&HBF
&X10111111



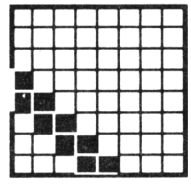
192
&HC0
&X11000000



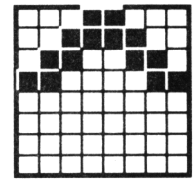
193
&HC1
&X11000001



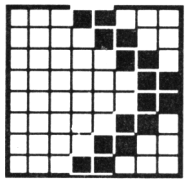
194
&HC2
&X11000010



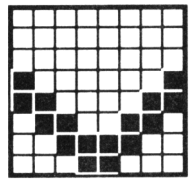
195
&HC3
&X11000011



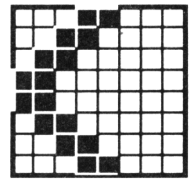
196
&HC4
&X11000100



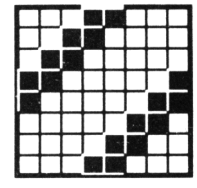
197
&HC5
&X11000101



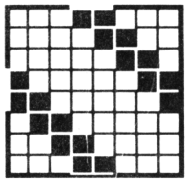
198
&HC6
&X11000110



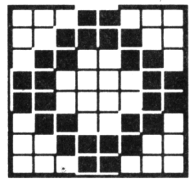
199
&HC7
&X11000111



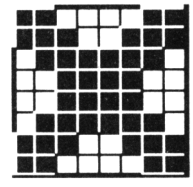
200
&HC8
&X11001000



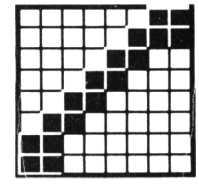
201
&HC9
&X11001001



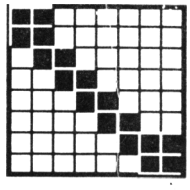
202
&HCA
&X11001010



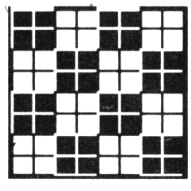
203
&HCB
&X11001011



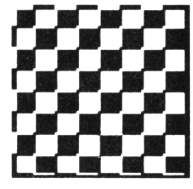
204
&HCC
&X11001100



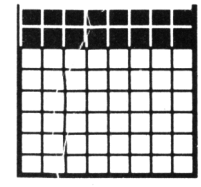
205
&HCD
&X11001101



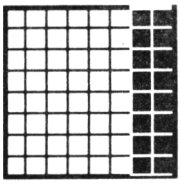
206
&HCE
&X11001110



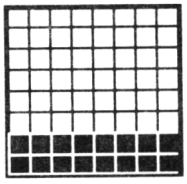
207
&HCF
&X11001111



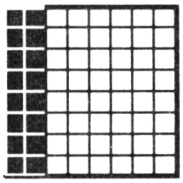
208
&HD0
&X11010000



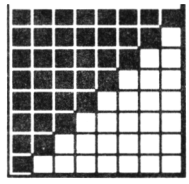
209
&HD1
&X11010001



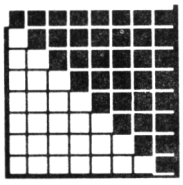
210
&HD2
&X11010010



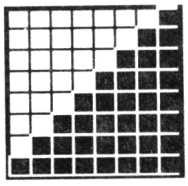
211
&HD3
&X11010011



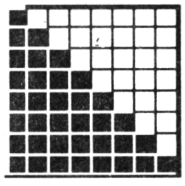
212
&HD4
&X11010100



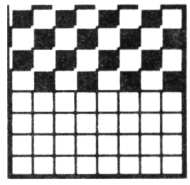
213
&HD5
&X11010101



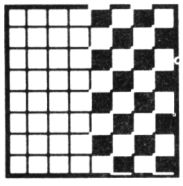
214
&HD6
&X11010110



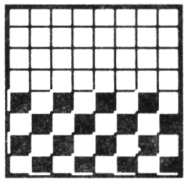
215
&HD7
&X11010111



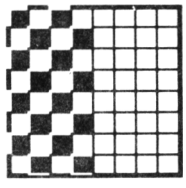
216
&HD8
&X11011000



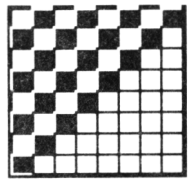
217
&HD9
&X11011001



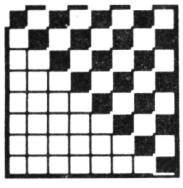
218
&HDA
&X11011010



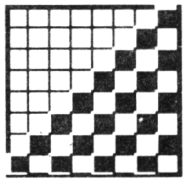
219
&HDB
&X11011011



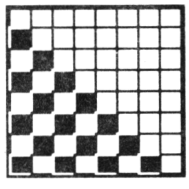
220
&HDC
&X11011100



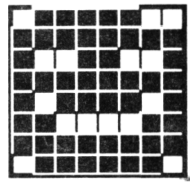
221
&HDD
&X11011101



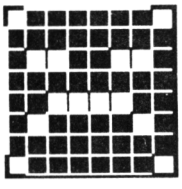
222
&HDE
&X11011110



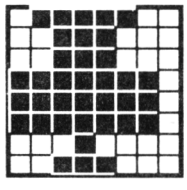
223
&HDF
&X11011111



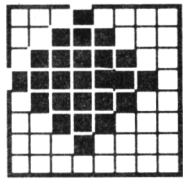
224
&HE0
&X11100000



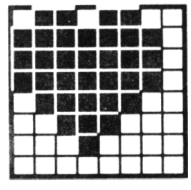
225
&HE1
&X11100001



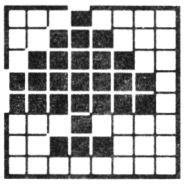
226
&HE2
&X11100010



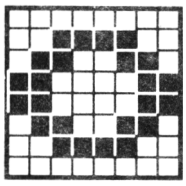
227
&HE3
&X11100011



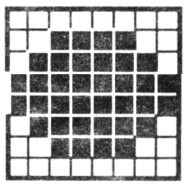
228
&HE4
&X11100100



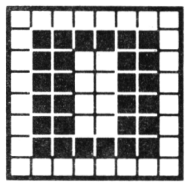
229
&HE5
&X11100101



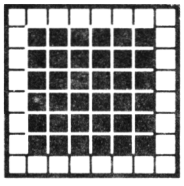
230
&HE6
&X11100110



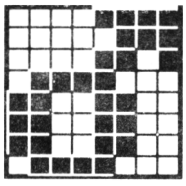
231
&HE7
&X11100111



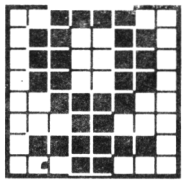
232
&HE8
&X11101000



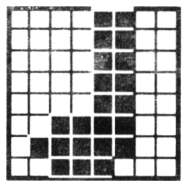
233
&HE9
&X11101001



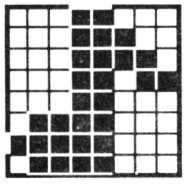
234
&HEA
&X11101010



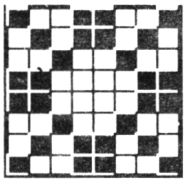
235
&HEB
&X11101011



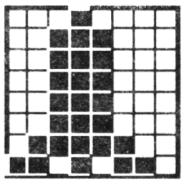
236
&HEC
&X11101100



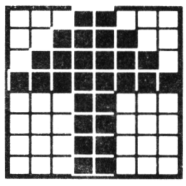
237
&HED
&X11101101



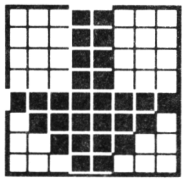
238
&HEE
&X11101110



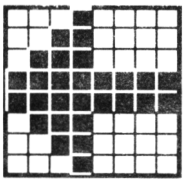
239
&HEF
&X11101111



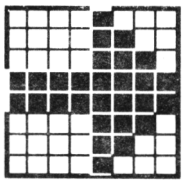
240
&HF0
&X11110000



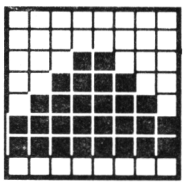
241
&HF1
&X11110001



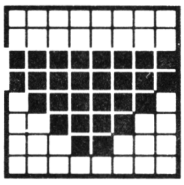
242
&HF2
&X11110010



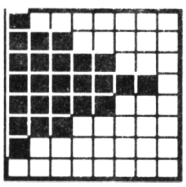
243
&HF3
&X11110011



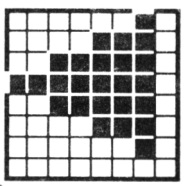
244
&HF4
&X11110100



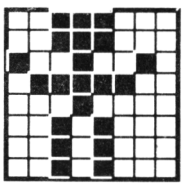
245
&HF5
&X11110101



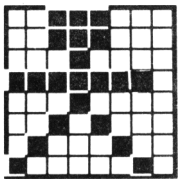
246
&HF6
&X11110110



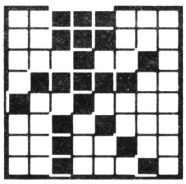
247
&HF7
&X11110111



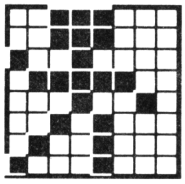
248
&HF8
&X11111000



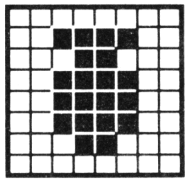
249
&HF9
&X11111001



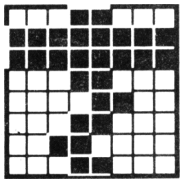
250
&HFA
&X11111010



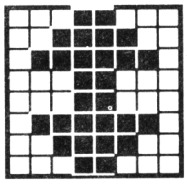
251
&HFB
&X11111011



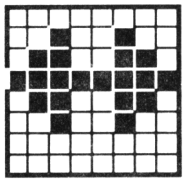
252
&HFC
&X11111100



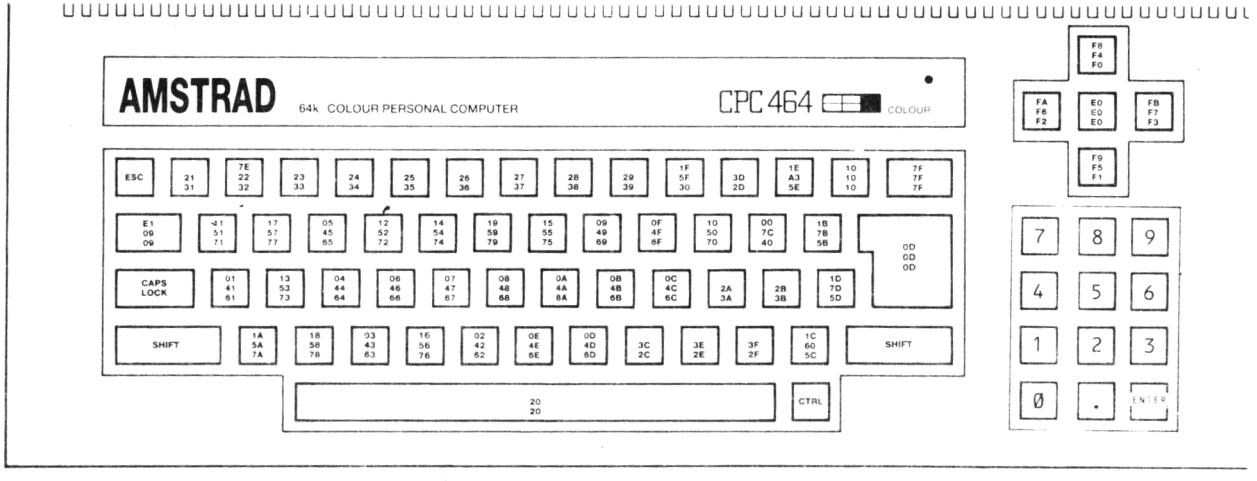
253
&HFD
&X11111101



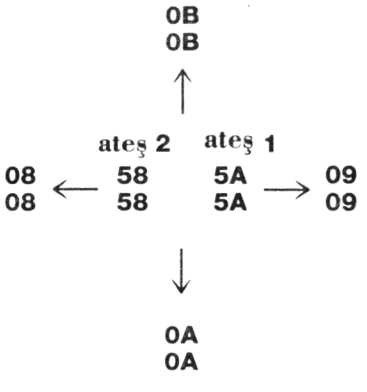
254
&HFE
&X11111110



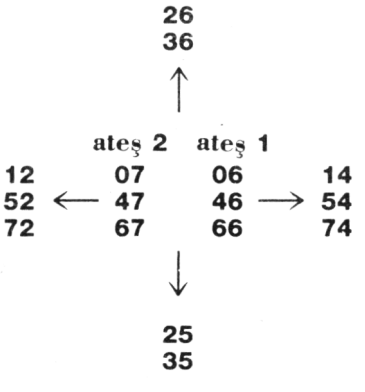
255
&HFF
&X11111111

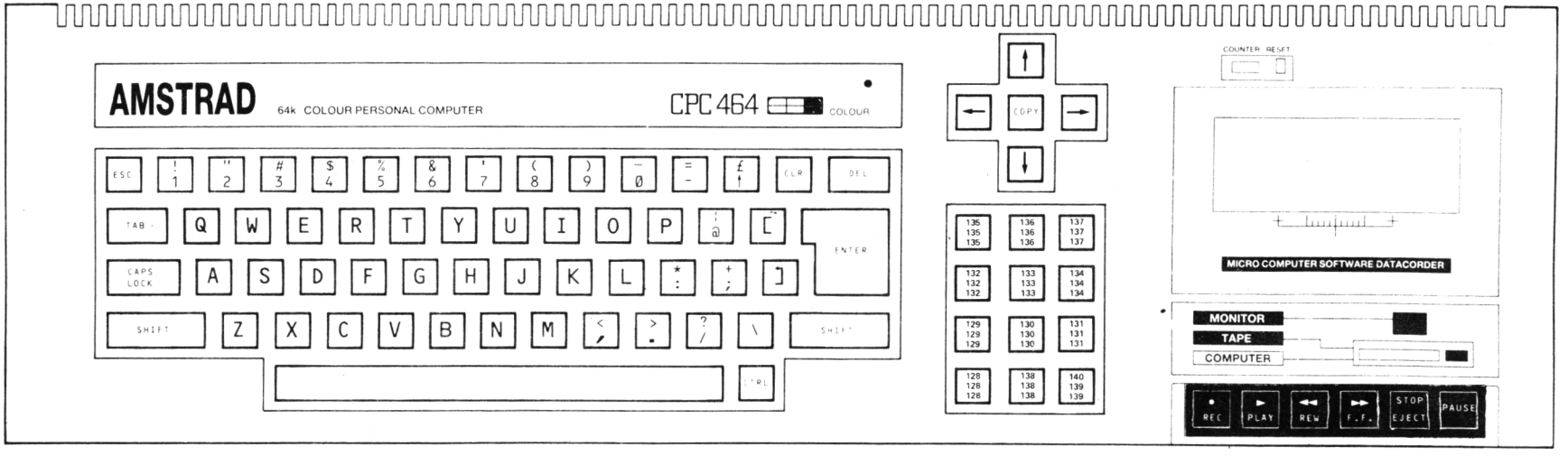


Oyun çubuğu 0



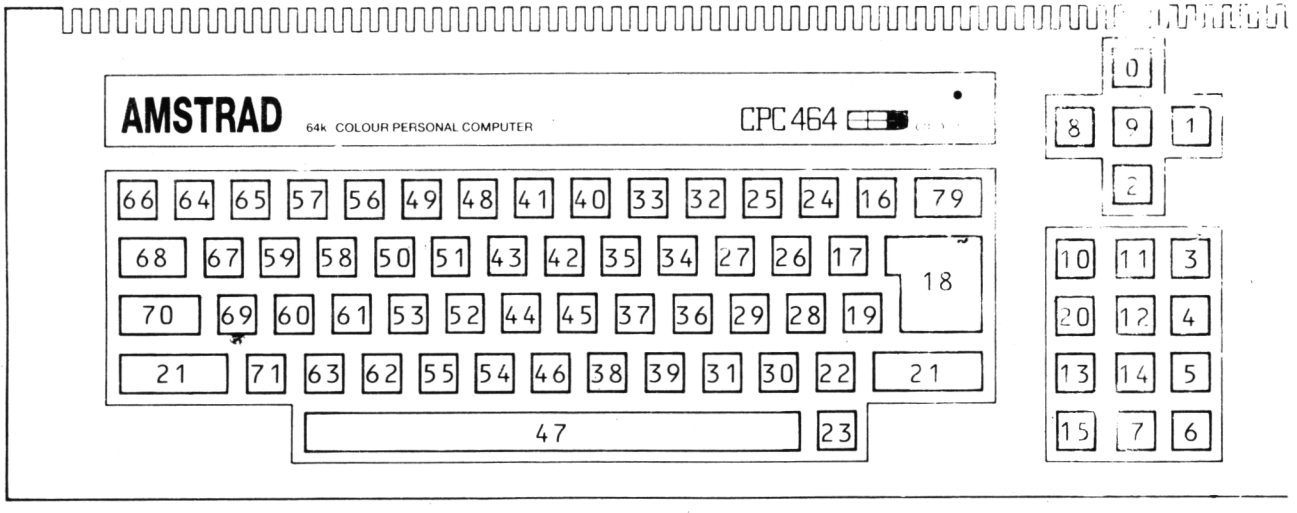
Oyun çubuğu 1



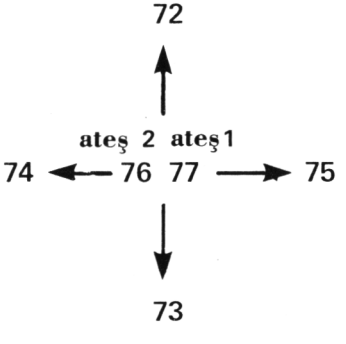


ek	değer	ascii
128	0	30
129	1	31
130	2	32
131	3	33
132	4	34
133	5	35
134	6	36
135	7	37
136	8	38
137	9	39
138	.	2E
139	[enter]	0D
140	run"	52,55,4E,22,0D

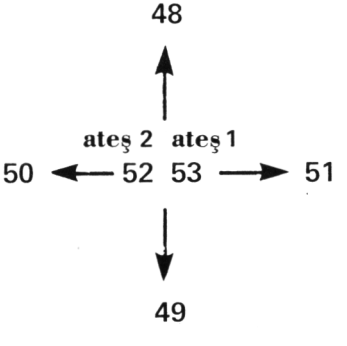
EK KARAKTERLER



Oyun çubuğu 0



Oyun çubuğu 1



EK IV

CPC 464 de deneyimli kullanıcılar için notlar:

CPC 464 alçak fiyatlı renkli kişisel bilgisayar, kurulmuş teknolojilerin cömert, etkili ve mükemmel kullanımını çok çekici bir format da toplar. Bu özellikleri genişleyebilme kapasitesiyle de birleşince yeniler için de deneyimler için de çok çekici bir bilgisayar ortaya çıkar. Donanım ve *ROM* yazılımı için başlayanlara da deneyimli bilgisayar alanlara da dost bir ortam sağlamak üzere biçimlendirilmiştir. Bu ortamda olan yazılımların isteğe göre değişme olanağını yanında yeni yazılımlar da CPC 464 değişik özelliklerden yararlanmak üzere yazılabilir.

Sistemin başlıca özellikleri şunlardır:

Z80CPU
Bütün dünyada ev bilgisayarlarında en çok kullanılan mikroişlemcilerdir. Ve en iyi desteklenen yazılım tabanı vardır. Özellikle CPC 464. ı CP/M kullanma potansiyeli sağladığı için. Tek ve orıginal interrupt idaresi yapısı CPC 464'ün *BASIC*'de *AFTER* ve *EVERY* gibi yeni özellikler kazandırdığı kontrol olanakları da sağlamıştır.

64K RAM

BASIC kullanımında *ROM* değişimi tekniği kullanılması sayesinde kullanıcıya 42K dan fazla standart olarak büyük bir miktar *RAM* sağlanmıştır.

EKRAN

CPC 464 ün üç temel ekran idare *MOD*'u vardır. Buna 80 sütun metin, 27 renklik bir palet ve 640 x 200 pıxeli kadar ekran görüntü kapasitesi dahildir.

GERÇEK KLAVYE:

Tam özellikle daktilo-stili klavye, mantıklı bir kürsör tuş demeti ve her fonksiyon tuşunun kullanımı iki katına çıkartan standart sayısal tuş tablosu vardır.

İÇE YERLEŞTİRİLMİŞ KASET

Kaset depolaması içe yerleştirilmiş bir özellik olarak sağlanmıştır.
Böylece ek bağlantı karışıklığı, seviye ayarı v.b. olmadan işlem yapma olanağı vardır. I K baud ya 2 k baud hızıyla (hızıyla seçimli) yazmak ve yazılımın saptadığı otomatik okuma hızı vardır.

BASIC

İngiltere'de yazılmış standart bir *BASIC*, *BASIC*'lerden ummaya alıştığınızdan çok daha hızlı ve çok yünlü olduğu gibi çok grafik ve ses uzantılarının yanısıra geniş bir firmware desteğine sahiptir.

GENİŞLETİLMİŞ KAREKTER SETİ:

Klavyeden ve CHR$ (n) fonksiyonu kullanılarak elde edilen sembol ve grafikler dahil tam bir 8-bit karekter seti vardır.

Kullanıcının tanımladığı tuşlar:

32'ye kadar karekter dizisi olan 32 tuş kullanıcı tarafından tanımlanabilir. 255 karekterlik tam bir set (hepsi ASCII ve ek olarak 100 den fazla karekter) isteğe göre kullanıcı tarafından tekrar tanımlanabilir.

Alt programlar

Bir çok makina kodu alt program *BASIC*'den çağırabilir.

EKRAN MODLARI

3 *EKRAN MODU* operasyonu vardır.
a) normal

MOD 1: 40 sütun x 25 satır
320 x 200 pixel-tek tek istenebilir 4 renk.
b) Çok renkli mod.
20 sütun x 25 satır 16 'ink' 'mürekkep' text.
160 x 200 pixel tektek istenebilen 16 renk.
c) Yüksek rezolusyon (ekran görüntü kapasitesi) modu.
80 sütun x 25 satır, 2 mürekkep text.
640 x 200 pixel tektek istenebilen 2 renk.

RENK SEÇİMİ:

(Bu kitapda, aşağıdaki tanımlamaların amacına uyularak *SİYAH* renk kabul edilmiştir.)

BORDER ekran modunda bağımsız olarak istenilen iki renge ayarlanabilir. Yani yanan sönen ya da tek renk yani sabit kullanılabilecek (ink) mürekkep sayısı seçilen ekran moduna bağlıdır. Her mürekkep iki renge yani yanan sönen ya da tek renge yani sabit ayarlanabilir. Daha önce anlatıldığı gibi kullanılacak mürekkep sayısı ekran moduna bağlıdır. Metin kağıdı metin kalemi ve grafik kalemi durumuna göre eldeki mürekkeplere ayarlanır.
Metin yazısı yarı saydam ya da opal olabilir. Yani kağıt rengini kâle almadan üstüne yazar ya fonun durmunun üstüne yazar.

PENCERELER (WINDOWS)

Kullanıcı içine karekterlerin yazılacağı sekize kadar text penceresi ve içinde çizim yapılacak grafik pencere seçebilir.
Ekran modu ayarlandığında pencereler ilk konumuna döner.
Eğer text pencereleri tüm ekrana ayarlıysa daha hızlı çalışır. Eğer text penceresi ekrandan küçükse bunu yazılım yapar bu nispeten daha yavaştır.

KÜRSÖR

CPU klavyeden giriş beklemediği zamanlar, kürsör etkisizdir ve böylece otomatik olarak görev yapar.
Kürsör boş bir renk karesidir.

POLİFONİK SES

CPC 464'ün ses olanakları General Instuments AY8910 ailesinden endüstri standardı ses generatörü tümdevresi kullanılarak yaratılır. Bu alet 3 kanal (ses) ile işler. Bunlardan herbiri bağımsız olarak ton ve genlik ayarı yapılabilir. Beyaz gürültü gereğine göre katılabilir. Üç kanal sol sağ ve orta olarak görünür. İçindeki yazılım genlik ve ton için envelope olanaklar sağlar. Ses jeneratörünün iç genlik envelopu normal olarak kullanılmaz.

Yazıcı arabirimi:

Handshake operasyonları için ('busy' işaret satırı kullanılarak endüstri standardı Centronics arabirimi sağlanmıştır.

KOORDİNATLAR

Text başlangıcı ekranın sol üst köşesidir ve fizik konumlar ekranda ekran moduyla değişir. Grafikin başlangıcı sol alt köşedir ve bu zaman ekranın yüksek resolu siyah modunda olduğu varsayılır. Ama mürekkep hesaplar bu ekran modunda doğru olarak yapılır.

NOT:

Normal ekran modunda bir pixelin IKİ yatay adresi vardır.Çok renkli modda her pixelin 4 yatay adresi vardır. Pixel konumunu tanımlamak için bu dördünden herhangi biri kullanılabilir.

Dikey aksların 0.... 399 arasında koordinatları vardır.

Bunlar ikiye bölünürler ve 0.....199 arası fizik konumu vermek üzere kesililer.

GENİŞLEME ROMLARI:

Bütün ***ROM***lar hafızanın üst 16K sını işgal ederler. (***BASIC*** buradadır) ve firmware de daha çağırılabilecek 240 ek 16K eklemek için olanaklar vardır.
Genel Bakış-
CPC 464 ün başlıca donanım özelliklerinin kısa bir özeti.

1) Donanım:

1.1) Ana CPC 464 kasasının içinde.

Bilgisayar, klavye, kaset veri kaydedici ve oparlör. RGB ve ışık çıkışları vardır.

1.1.1) LSI yongaları

4MHz hızında çalışan Z80 A mikroişlemcisi ekrana ulaşımla tazelenen 64K bytlık 64Kxl dinamik RAM
BASIC ve işletme sistemi içeren 32K bytlık ***ROM*** [Custom logiç bir matris LS] yonganın içinde olmayan mantığın hemen hepsini: özellikle zamanlamayı ve renk çıkartımını ve DMA devre sistemini içinde toplar.
NB.
Planlama karışıktır ve ekran moduyla değişir. Yazılım gerektirirse CRTC tomarlama (yana doğru 1/40 ekran eniyle) ve yuvarlama (8 scon satırıyla yukarı ve aşağı) kullanılabilir. CRTC nin içindeki parametreler satır adedini ve frame hızını, genişlik ve sınırların konumunu seçer.

Ses Jenerator Yongası AY-3-8912:-

3 ses. Ses üç kanaldan alınır ve iç oparlörde volume kontroluyla kontrol edilen mono bir çıkış meydana getirmek için eşit bir şekilde karıştırılır. Ayrıca dıştan bir stero çıkış vardır ki burada...

Sol= Kanal A+ 1/2 kanal C. Sağ= Kanal B + 1/2 Kanal C. Bu yonga aynı zamanda klavye ve oyun çubuğu girişi bilgisi de alır.

1.1.2 Dış Girişler.

Genel amaçlı PCB bağlantıları (12) ve dış printer (12) (paralel centronics) oyun çubuğu için giriş fişleri (14) (9 iğne D tipi), stereo ses çıkarımı (15), video çıkarımı (10) (***RGB*** sync ya da composite video ya da ilumınance ve sync)

1.2 Dış Kasa:

CPC 464 sistemi seçime bağlı 2 tip monitör sağlar. Her biri bilgisayar için 5 v güç kaynağı içerir, satılan ülkedeki standart voltaja uyması düşünülerek biçimlendirilmiştir. Bunlara ek olarak seçenekli PSU ve UHF modulatörü MP1 vardır.

Centronics arabirimli printeri ve Hi Fi ünitesini bağlamak için kablo gerekecektir.

1.3 Görüntü Özelliği:

Ekran hafızasının 16K sından işler. Aletin 27 rengi vardır. Ve bunlar paletten rahatlıkla seçilebilir. Paletdeki mürekkep sayısı seçilen ekran moduna bağlıdır. Gerekirse mürekkep aynı renge ayarlanır.

Ekrandaki pixeler belli bir mürekkebin noktaları olarak tanımlanır. (fon ya da kağıdın eldeki paletden bir mürekkebe gerektirdiğini unutmayın.) BORDER aktif resim bölgesini çevreler ve bundan bağımsız olarak 27 renkten ayarı yapılır.

MOD	MÜREKKEP ADEDİ	DİKEY NOKTALAR	YATAY NOKTALAR	YATAY KOLON
Normal	4	200	320	40
Yüksek Relolusyon	2	200	640	80
Çok renkli	16	200	160	20

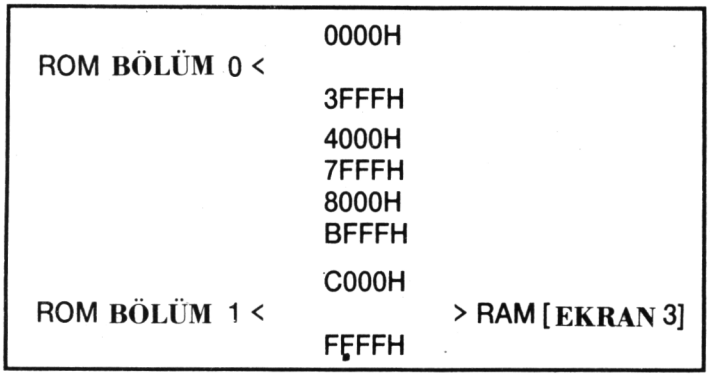
NOT: tek renk monitör kullanılırsa bilgisayar gri tonları verir.

Renklerin parlaklığa göre sıraları şöyledir:

GRİ SEVİYESİ	RENK	GRİ SEVİYESİ	RENK
0	SİYAH	13	BEYAZ
1	MAVİ	14	PASTEL MAVİ
2	PARLAK MAVİ	18	TURUNCU
3	KIRMIZI	16	PEMBE
4	(MAGENTA) PEMBE	17	PASTEL MAGENDA
5	MOR	18	PARLAK YEŞİL
6	PARLAK KIRMIZI	19	DENİZYEŞİLİ
7	EFLATUN	20	PARLAK CYAN
8	(ÇİNGENE PEM) PARLAK MAGENTA	21	TURUNÇ YEŞİLİ
9	YEŞİL	22	PASTEL YEŞİL
10	CYAN (MAVİ)	23	PASTEL CYAN
11	GÖK MAVİSİ	24	PARLAK SARI
12	SARI	25	PASTEL SARI
		26	PARLAK BEYAZ

1.4) HAFIZA HARİTASI

RAM'ın 64K si şöyle yer ayrımlarına uyar:



Dikkat edin: *ROM*'un parçası ekran ***RAM ile yerdeğiştirir ve bu surette BASIC*** operasyonları sırasında ***RAM*** kullanana mümkün olduğu kadar fazla alan verir.
Bir adreste hem RAM hem de ROM olduğu zaman OKUMA ROM u alır YAZMA da RAM'ı alır. ROM kısmından biri söndürülüp (kapatılıp) aynı adreste RAM'a okuma sağlanabilir.

1.5) Ekleme Yeteneği:

1.5.1 İlave ROM lar:

İçteki ROM un herhangi bir parçası yerine seçilecek ek ROM lar için tedarikli olunmuştur. Bu işlem bağlanacak bir arabirim ile gerçekleşecektir.

1.5.2) EK RAM

Ek RAM içteki RAM'ın herhangi bir parçasının yerine devreye sokulabilir. Adres kararı ve bölüm seçimi çıkışı bağlı bir modulün içinde olacaktır. Bu hafıza yalnız okunabilir olacaktır ve bu bilgisayardan ek RAM'a yazmak için I/0 planlaması ile ilgili özel şema gerekir.

1.5.3) Ek 1/0

1/0 PORT adreslerinin çoğu bilgisayar tarafından ayrılmıştır ve özellikle aşağıdaki adresler 7F xx e kadar hiç kullanılmamalıdır. Aşağıdakiler dış donatımla kullanılabilir F8xx, F9xx, FAxx, FBxx
Adres AIØ alçakken genişleme arabirimleri AO dan A7 ye kadar adresleri seçmelidir.
F800'den FBFF sıralamasındaki çıkış kapısı 1/0 kanalları şöyle ayrılmıştır.
AO -A7
adresleri
00— 7B **kullanmayın**
7C - 7F Disk arabirimi'i için ayrılmıştır.
80 - BB **Kullanmayın.**
BC - BF ilerde kullanım için ayrılmıştır.
CO - DB **Kullanmayın**
DC - DF İletişim arabirimleri için ayrılmıştır.
EO - FF kullanımı çevre ünitelerine müsaittir.

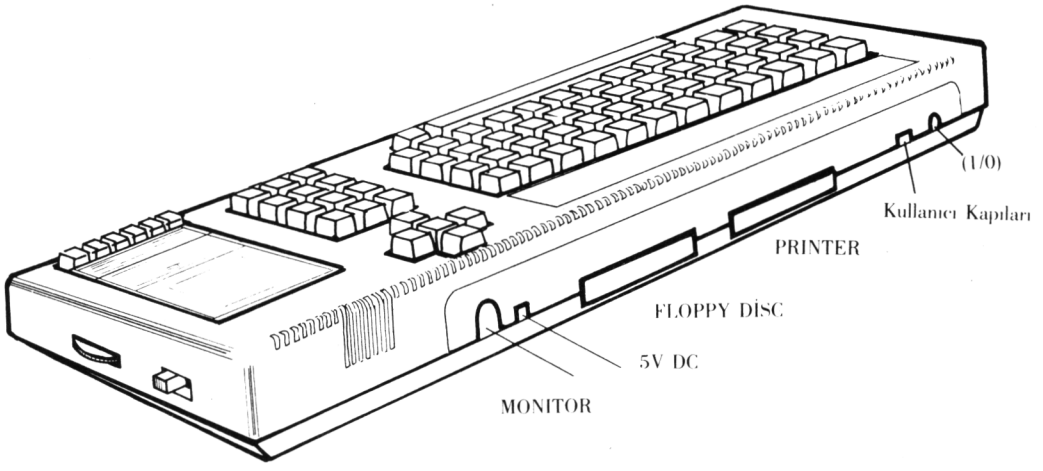
B registerini adres seçeneğinin üst yarısına koyan (A15-A8) Z80 komutları kullanılmalıdır.

2) Klavye

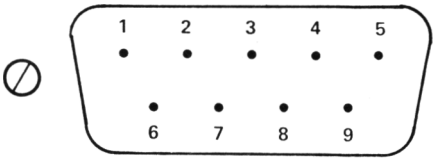
Bilgisayarı ilk konumuna getirmek için ***[CTRL] [SHIFT] ve [ESC]*** tuşlarına hep birlikte basılır. Yazılı karekter veren tuşlar ya da kürsör hareketi, sayısal tuş tablosundaki tuşlar

hariç, fırmware kontrolu altında kendilerini tekrarlarlar.
[ESC] programı yürütmesini durdurur tekrar [ESC] basılırsa yürütme bitirilir. Bir başka tuşa basılırsa program yürütmesine devam edilir.
[CAPS LOCK] caps lock tuşuyla çalışan. ***Shift*** (kilidi) de
[CTRL] [CAPS LOCK] beraber basılarak çalışır.
Copy kürsörü ***input*** kürsöründen ***[SHIFT]*** ve kürsör tuşlarına beraber basılarak ayrılır.
Giriş copy kürsörünün altındaki karekterden ***[COPY]*** tuşuna basarak elde edilir.
Kürsör tuşları ***input buffer ında*** düzeltmelere imkan vermek içindir. Bunlar ekranı satırlarına yayılabilirler. Kürsör tuşları, herhangi bir klavye ***girişi*** alınmadadan önce klavye ***girişinin*** başlangıcını ekranda herhangi bir konuma koymak için kullanılabilir. Ama klavye ***girişi*** aldıktan sonra koruma saptanmıştır. Yeni ***girişi*** texti 0 konumunda ekranda olan textin üstüne yazar.
[DEL] geriye doğru siler ***[CLR]*** ileriye doğru siler.

EK V.

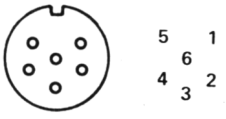


OYUN ÇUBUĞU KAPISI FİŞİ



PIN 1	YUKARI	PIN 6	ATEŞ 2
PIN 2	AŞAĞI	PIN 7	ATEŞ 1
PIN 3	SOL	PIN 8	ORTAK
PIN 4	YEDEK	PIN 9	ORTAK 2
PIN 5			

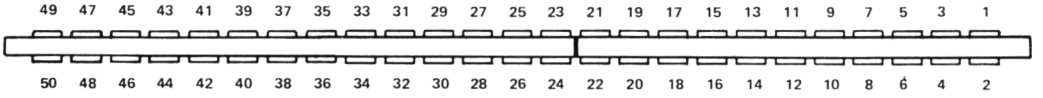
VİDEO ÇIKIŞI



PIN 1	KIRMIZI	PIN 4	SENKRONİZASYON
PIN 2	YEŞİL	PIN 5	TOPRAK
PIN 3	MAVİ	PIN 6	PARLAKLIK

DİSK ÇIKIŞI

50 YOLLU 0.1 EDGE CONNECTOR

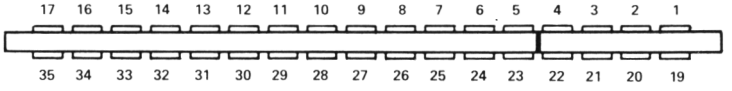


PIN 1	SOUND	PIN 18	A0	PIN 35	$\overline{INT}$
PIN 2	GND	PIN 19	D7	PIN 36	$\overline{NMI}$
PIN 3	A15	PIN 20	D6	PIN 37	$\overline{BUSRD}$
PIN 4	A14	PIN 21	D5	PIN 38	$\overline{BUSAK}$
PIN 5	A13	PIN 22	D4	PIN 39	READY
PIN 6	A12	PIN 23	D3	PIN 40	$\overline{BUS\ RESET}$
PIN 7	A11	PIN 24	D2	PIN 41	$\overline{RESET}$
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	$\overline{HALT}$		

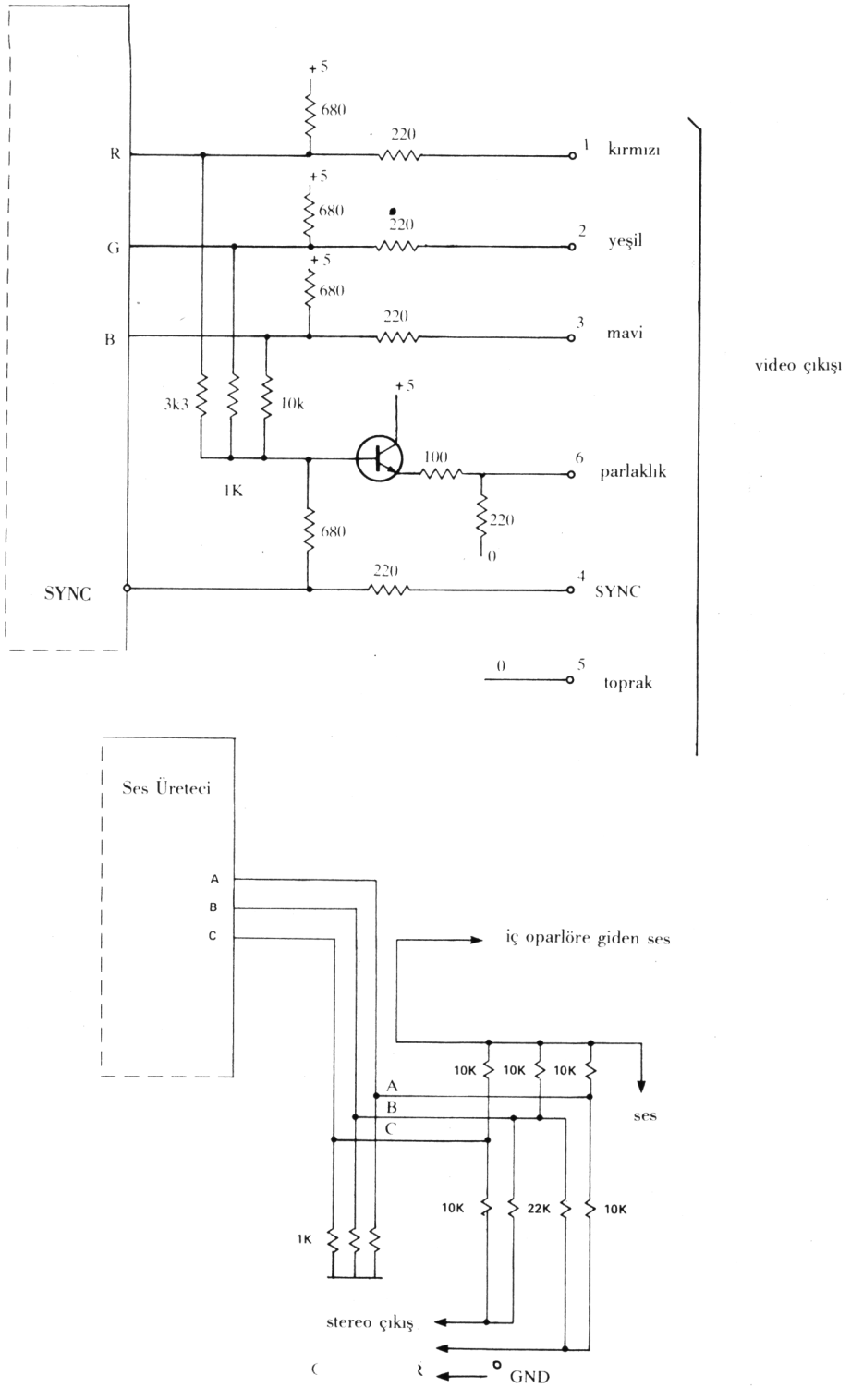
YAZICI ÇIKIŞI

34 YOLLU 0.1 EDGE CONNECTOR

VIEWED FROM REAR



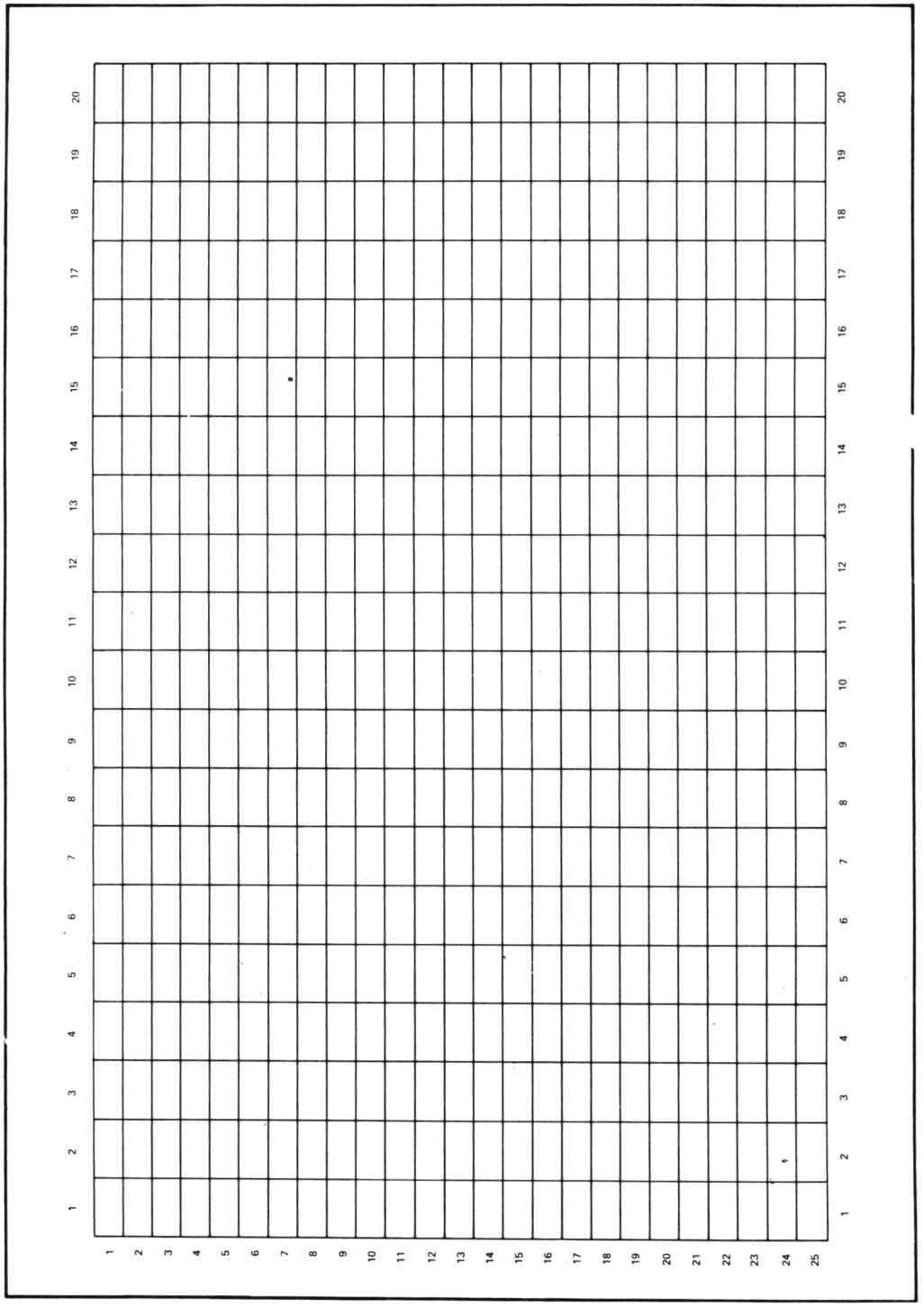
PIN 1	$\overline{STROBE}$	PIN 19	GND
PIN 2	D0	PIN 20	GND
PIN 3	D1	PIN 21	GND
PIN 4	D2	PIN 22	GND
PIN 5	D3	PIN 23	GND
PIN 6	D4	PIN 24	GND
PIN 7	D5	PIN 25	GND
PIN 8	D6	PIN 26	GND
PIN 9	D7	PIN 27	GND
PIN 11	BUSY	PIN 28	GND
PIN 14	GND	PIN 33	GND
PIN 16	GND	geriye kalanlar	


+5
680
220
R
1
kırmızı
+5
680
220
G
2
yeşil
+5
680
220
B
3
mavi
video çıkışı
3k3
10k
+5
100
6
parlaklık
1K
680
220
0
220
4
SYNC
SYNC
0
5
toprak
Ses Üreteci
A
B
C
iç oparlöre giden ses
10K
10K
10K
A
B
C
ses
10K
22K
10K
1K
stereo çıkış
GND

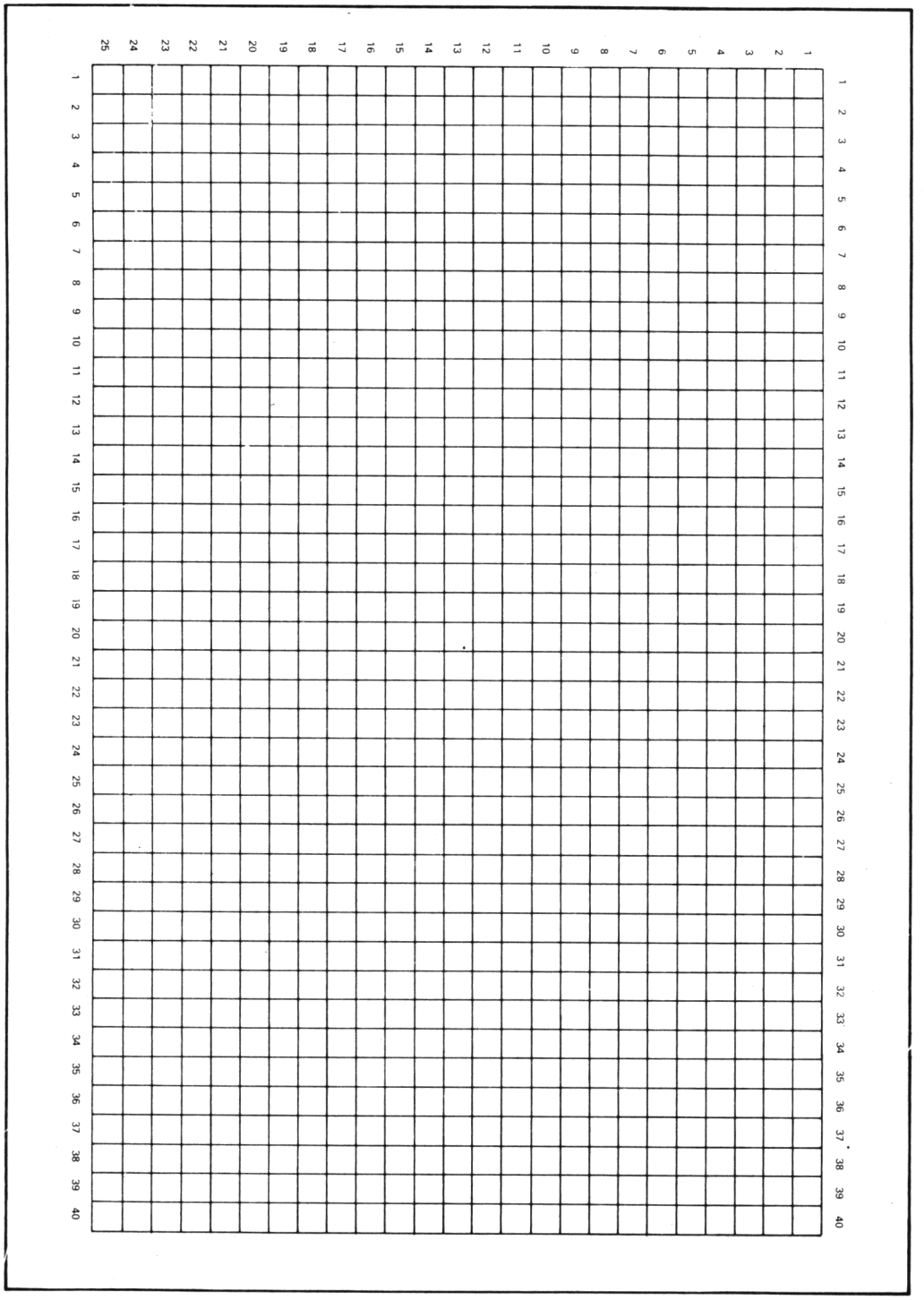
EK VI

metin ve pencere planı

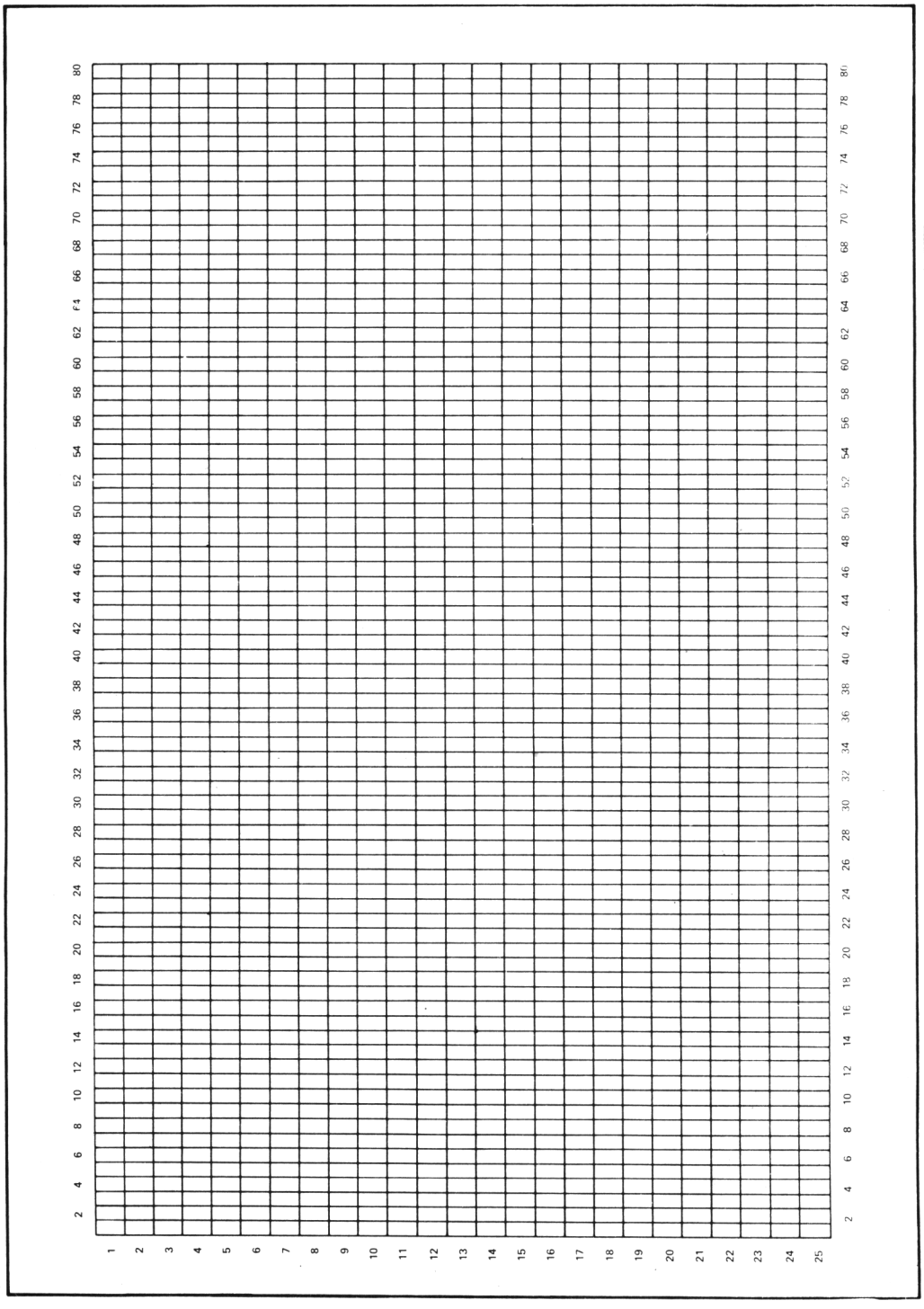
MODE 0 20 kolon



Mode 1 40 kolon



Mode 2 80 kolon



EK VII

Aşağıdaki tablo, olağan, yumuşak notalar için gamların tam 8 oktav sıralamasında salık verilen ton ayarlamasını verir.
Üretilen frekans tam gereken frekans değildir, çünkü period ayarı bir tamsayı olmalıdır. Görülecek hata esas frekans ve gereken frekans arasındaki farktır.

NOTA	FREKANS	PERİOD	HATA ORANI	
C	32.703	3822	-0.007%	
C#	34.648	3608	+0.007%	
D	36.708	3405	-0.007%	
D#	38.891	3214	-0.004%	
E	41.203	3034	+0.009%	Oktav- 3
F	43.654	2863	-0.016%	
F#	46.249	2703	+0.009%	
G	48.999	2551	-0.002%	
G#	51.913	2408	+0.005%	
A	55.000	2273	+0.012%	
A#	58.270	2145	-0.008%	
B	61.735	2025	+0.011%	

NOTA	FREKANS	PERİOD	HATA ORANI	
C	65.406	1911	-0.007%	
C#	69.296	1804	+0.007%	
D	73.416	1703	+0.022%	
D#	77.782	1607	-0.004%	
E	82.407	1517	+0.009%	Oktav- 2
F	87.307	1432	+0.019%	
F#	92.499	1351	-0.028%	
G	97.999	1276	+0.037%	
G#	103.826	1204	+0.005%	
A	110.000	1136	-0.032%	
A#	116.541	1073	+0.039%	
B	123.471	1012	-0.038%	

NOTA	FREKANS	PERIOD	HATA ORANI	
C	130.813	956	+0.046%	
C#	138.591	902	+0.007%	
D	146.832	851	-0.037%	
D#	155.564	804	+0.058%	
E	164.814	758	-0.057%	Oktav- 1
F	174.614	716	+0.019%	
F#	184.997	676	+0.046%	
G	195.998	638	+0.037%	
G#	207.652	602	+0.005%	
A	220.000	568	-0.032%	
A#	233.082	536	-0.055%	
B	246.942	506	-0.038%	

NOTA	FREKANS	PERIOD	HATA ORANI	
C	261.626	478	+0.046%	
C#	277.183	451	+0.007%	
D	293.665	426	+0.081%	
D#	311.127	402	+0.058%	
E	329.628	379	-0.057%	
F	349.228	358	+0.019%	Oktav- 0
F#	369.994	338	+0.046%	
G	391.995	319	+0.037%	
G#	415.305	301	+0.005%	
A	440.000	284	-0.032%	
A#	466.164	268	-0.055%	
B	493.883	253	-0.038%	

NOTA	FREKANS	PREIOD	HATA ORANI	
C	523.251	239	+0.046%	
C#	554.365	225	-0.215%	
D	587.330	213	+0.081%	
D#	622.254	201	+0.058%	
E	659.255	190	+0.206%	
F	698.457	179	+0.019%	Oktav- 1
F#	739.989	169	+0.046%	
G	783.991	159	-0.277%	
G#	830.609	150	-0.328%	
A	880.000	142	-0.032%	
A#	932.328	134	-0.055%	
B	987.767	127	+0.356%	

NOTA	FREKANS	PERIOD	HATA ORANI	
C	1046.502	119	-0.374%	
C#	1108.731	113	+0.229%	
D	1174.659	106	-0.390%	
D#	1244.508	100	-0.441%	
E	1318.510	95	+0.206%	
F	1396.913	89	-0.543%	
F#	1479.978	84	-0.548%	Oktav- 2
G	1567.982	80	+0.350%	
G#	1661.219	75	-0.328%	
A	1760.000	71	-0.032%	
A#	1864.655	67	-0.055%	
B	1975.533	63	-0.435%	

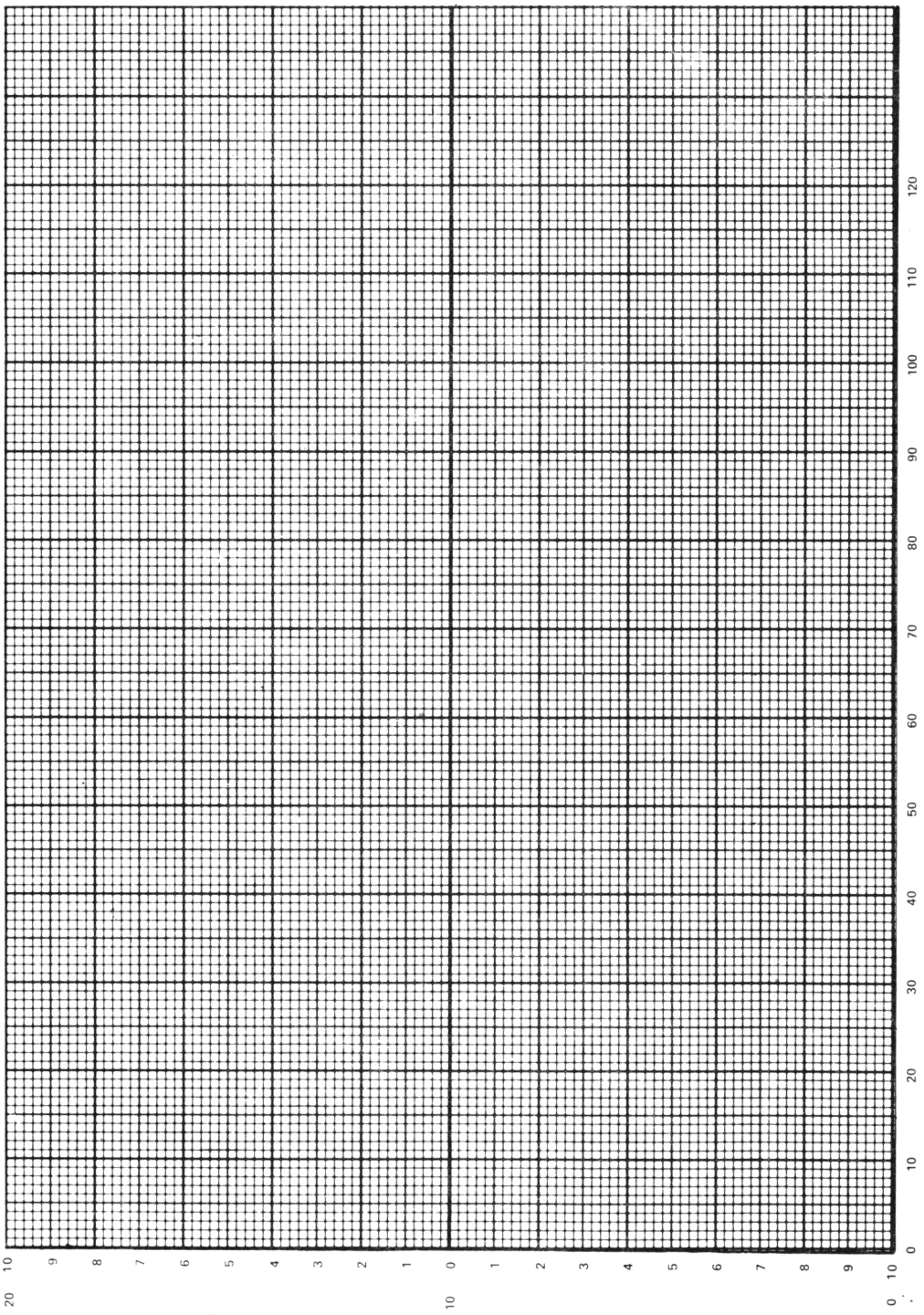
NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	2093.004	60	+0.462%	
C#	2217.461	56	-0.662%	
D	2349.318	53	-0.390%	
D#	2489.016	50	-0.441%	
E	2637.021	47	-0.855%	
F	2793.826	45	+0.574%	Oktav- 3
F#	2959.955	42	-0.548%	
G	3135.963	40	+0.350%	
G#	3322.438	38	+0.992%	
A	3520.000	36	+1.357%	
A#	3729.310	34	+1.417%	
B	3951.066	32	+1.134%	

NOTE	FREQUENCY	PERIOD	RELATIVE ERROR	
C	4186.009	30	+0.462%	
C#	4434.922	28	-0.662%	
D	4698.636	27	+1.469%	
D#	4978.032	25	-0.441%	
E	5274.041	24	+1.246%	
F	5587.652	22	-1.685%	
F#	5919.911	21	-0.548%	Oktav- 4
G	6271.927	20	+0.350%	
G#	6644.875	19	+0.992%	
A	7040.000	18	+1.357%	
A#	7458.621	17	+1.417%	
B	7902.133	16	+1.134%	

Hesaplama Yöntemi

FREKANS = 440 3 42↑ (OKTAW + (10-N) /12))
PERIOD = ROUND (125000 / FREQUENCY)

Ses envelopu planlaması



EK VIII

Hata Kodları ve Ayrılmış Sözcükler

Hata numaraları ve Hata mesajları

BASIC anlamadığı ya da işlemden geçiremediği bir program bildirimi, sözcük ya da değişkene rastladığı zaman durur ve bir hata mesajı gösterir. Mesajın şekli genellikle neyin yanlış olduğunu belirtir. Ama bazen, hata programın girişi sırasında olan topografik bir hata ise *BASIC* düzeltme moduyla ve yanlış giriş yapılan satırla uyarır. Dikkatsiz (yanlış) yazan kişinin en sık karşılaştığı mesaj *Syntax error* (NO 2) mesajıdır, ve *BASIC* program modunda karşılaşmışsa düzeltme satırıyla uyarır. Direct ya da doğrudan modda yalnızca bir hatanın yapıldığını bildirir ve son yazılan satırın problemi fark etmek için görünür olduğunu varsayar.

Eğer programın başına *ON ERROR GOTO* emri konulmuşsa hatayı farkettiği zaman verilen satır numarası için bilgisayara gönderim yapar. Aşağıdaki örnekte hata farkedildiğinde bilgisayar 1000. satıra gönderilmiştir.

10 ON ERROR GOTO 1000

Program

1000 PRINT CHR$ (7) : MODE 2: INK 1,0 : INK 0,9 : CLS : LIST

Bu durumda CPC 464 biip sesi çıkartır. Son kullanılan ekranı temizler, 80 sütun görüntü için uygun bir renk kombinasyonuna değişir ve program için *LIST* edilir.

Eğer hata (sözdizim hatasıysa) *Syntax error* listing'in en altında satır edit (düzeltme) modunda düzeltme bekler ama *syntax error* mesajı önlenir. 100 den bir önceki satırda *END* ile program bitirmeyi unatmazsanız ekranda sonuçları kurtarabilirsiniz.

Geçerli giriş için *BASIC* hata mesajı vermez, dolayısıyla hâlâ (error) mesajı görüldüğünde programda bir hata aramalıdır, genellikle çıkan mesajı size hatayı düzeltme işleminde yol gösterici olacaktır.

Bir çok alanda olduğu gibi burada da hatalarınızdan çok şey öğreneceksiniz. O halde CPC 464'ün çok hoşgörülü bir hoca olmasından yararlanın: CPC 464'ün sabrı tükeninceye kadar siz denemekten bıkacaksınız. *BASIC*'ın çıkartığı bütün hata mesajları burada numara sırasıyla verildiği gibi mümkün nedenleri de kısaca tanımlanmıştır.

1- Unexpected NEXT (Beklenmiyen NEXT)

FOR çevriminde olmıyan bir *NEXT* emriyle karşılaşılmıştır ya da *NEXT* emrindeki kontrol değişkeni *FOR* unkine uymuyordur.

2- Syntax Error

Sözdizim hatası. *BASIC* verilen satırı anlamıyor yapısında yasal olmayan bir şey var.

3- Unexpected RETURN. (beklenmiyen *RETURN)* Alt programda olmadan *RETURN* emriyle karşılaşılmıştır.

4- DATA exhausted (Veri bitti)

Son verinin sonundan sonra *READ* emri okumaya kalkışmıştır.

5) İmproper argument (Uygunsuz girdi)

Bu genel amaçlı bir yanlıştır. Bir fonksiyonun girdisi ya da bir emir parametresi bir şekilde geçersizdir.

6) Owerflow (taşma)

Taşan bir aritmetik işlemi sonucudur. Bu bir ondalık sayı taşması olabilir, bu durum bir

operasyon 1.7E-38 den daha büyük bir değer vermiştir. Ya da bü bir ondalık sayıyı 16 bitlik tamsayıyla çevirme girişimin sonucudur.

7) Memory full. (hafıza dolu)
En son program ya da değişkenleri hafızayı aşıyordur. Bir hafıza emri, eğer **BASIC**'ın hafızası çok fazla alçak ya da olanaksız derecede yüksek değere ayarlanmaya girişilmişse bu hatayı verir. Unutmayın ki açık bir kaset kütüğünün ona ayrılmış bir tamponu vardır bu da hafızanın kullanabildiği değerleri kısıtlar.

8) Line does not exist (bu satır yok)
Referans verilen satır bulunamıyor.

9) Subscript out of range
Bir matris referansındaa indislerden biri küçük ya da çok büyük.

10) Array already dimensioned
DIM bildiriminde matrislerden biri önceden belirtilmiş.

11) DİVİSİON by zoro
Bu gerçek bölmede, tamsayı bölmesinde, tamsayı ya da üsleme işlemlerinde olabilir. 0 ile bölme bu sonucu verir.

12) Invalid direct command
Son yapılan emir girişimi direkt modda geersizdir.

13) Type mismatch.
Karekter dizisi değeri gereken bir yere sayısal değer verilmiştir. Ya da tam tersi ya da *READ* ya da *INPUT*da geçersiz bir şekilde oluşturulmuş bir sayı bulunmuştur.

14) String space full
Programda hafızanın alamayacağı kadar karakter dizisi üretilmiş.

15) String too long (karakter dizisi fazla uzun)
255 & i geçen bir karekter dizisi- Bir kaç tane karekter dizisini ekliyerek üretilmiş olabilir.

16) String expression too complex (karekter dizisi ifadesi fazla karmaşık)
Bir karekter dizisi ifadesi bir kaçtane ara karekter dizisi değerleri üretebilir. Bu değerlerin sayısı makul sınırları geçince *BASIC* pes eder ve bu hata ile sonuçlanır.

17) Cannot CONT inue (devam edilemez)
Bir nedenden ötürü son program *CONT.* kullanılarak yeniden başlatılamaz. Unutmayın ki *CONT STOP* emrinden sonra yeniden başlatmak içindir. *[ESC] [ESC]* ya da bir hata ve bu arada programda bir değişiklik yeniden başlamayı imkansız yapar.

18) Unknown user function (bilinmeyen kullanıcı fonksiyon)
Yeni istenen *FN* için *DEF FN* yürütülmemiştir.

19) RESUME missing (RESUME eksik)
Hata işlemi modundayken programın sonuyla karşılaşmıştır. (Yani bir *ON ERROR GOTO* yüzünden)

20) Unexpected RESUME (beklenmiyen bir RESUME)
Resume yalnızca hata işlenmesi modundayken geçerlidir.
(Yani bir ON ERROR GOTO ile)

21) Direct commond found doğrudan emir bulundu)
Kasetten program yüklerken satır numarası olmayan bir satır bulunmuştur.

22) Operand missıng
BASIC tamamlanmış bir ifade ile karşılaşmıştır.

23) Line too long (Satır fazla uzun)
BASIC iç formuna çevrilince çok büyük olan bir satır.

24) EOF met
Kaset input kanalındaki bir kütüğün geçmiş sonu okuma girişiminde bulunulmuştur.

25) File type error (Kütük tipi hatası)
Okunan kütük uygun tipde değil. *OPENIN* yalnızca *ASCII* tip text kütüklerine hazırlar.

LOAD, RUN. vs. yalnızca *SAVE* tarafından üretilen tip kütüklerler uğraşmaya hazırlar.
26) NEXT missing (NEXT eksik)
FOR emrine uyacak *NEXT* bulunamıyor.
27) File already open (Kütük açık)
Bir önce açılan kütük kapatılmadan *OPENIN* ya da *OPEN OUT* emri uygulanmış.
28) Unknown command (bilinmeye emir)
BASIC dışardan bir emre alıcı bulamıyor.
WEND emrine uyan bir WEND bulunamıyor.
29) WEND missing (WEND eksik)
WHİLE emrine uyan bir *WEND* bulunamıyor.
30) Uneppected WEND (beklenmiyen WEND)
While döngüsünde olmadan bir *WEND* ile karşılaşıldı ya da *WEND* son While döngüsüne uymuyor.

BASIC Komutları

Aşağıdakiler *BASIC* komutlar. ayrılmışlardır ve değişken isim olarak kullanılamazlar.

ABS, AFTER, AND, ASC, ATN, AUTO

BIN$, BORDER

CALL, CAT, CHAIN, CHR$, CINT, CLEAR, CLG, CLOSEIN, CLOSEOUT, CLS, CONT, COS, CREAL

DATA, DEF, DEFINT, DEFREAL, DEFSTR, DEG, DELETE, DI, DIM, DRAW, DRAWR

EDIT, EI, ELSE, END, ENT, ENV, EOF, ERASE, ERL, ERR, ERROR, EVERY, EXP

FIX, FN, FOR, FRE

GOSUB, GOTO

HEX$, HIMEM

IF, INK, INKEY, INKEY$, INP, INPUT, INSTR, INT

JOY

KEY

LEFT$, LEN, LET, LINE, LIST, LOAD, LOCATE, LOG, LOG10, LOWER$

MAX, MEMORY, MERGE, MID$, MIN, MOD, MODE, MOVE, MOVER

NEXT, NEW, NOT

ON, ON BREAK, ON ERROR GOTO, 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

DİZİN

SAYFA NO

Baskı : Doyuran Matbaası
Ofset Hazırlık : Sistem Foto Dizgi - 523 82 82

EKAKOMP

Büyükdere Caddesi Ayazağa Asfaltı 3. Yol No: 35
Ayazağa/İSTANBUL
Tel: 176 26 60/3 Hat
Telex: 245 89 — ENBA Tr.

...ve modern bir ev için Amstrad'lar veya Sinclair'ler size en geniş seçme şansını sunmaktadır.

AMSTRAD CPC 464: Monitörlü, kasetli komple sistem. 64K RAM, 32K ROM, 27 ayrı renk, 3 kanal ses, paralel yazıcı çıkışı ve joystick çıkışı.

AMSTRAD CPC 6128: Monitörlü, disketli komple sistem. 128K RAM, 48K ROM, 170 KB disket kapasitesi, 27 ayrı renk, 3 kanal ses, CP/M 2.2 ve CP/M 3.1 işletim sistemi, paralel yazıcı çıkışı ve joystick çıkışı.

SINCLAIR SPECTRUM + 2 : Kasetli sistem. 128K RAM, 32 K ROM, 8 renk, TV bağlantılı 3 kanal ses, 48K ve 128K BASIC programlama modu. Hesap makinesi, RAM disc, seri haberleşme çıkışı, 2 joystick çıkışı.

SINCLAIR SPECTRUM + 3 : Diskli sistem, 128K RAM, 64K ROM, 8 renk, TV bağlantılı 3 kanal ses, 48K ve 128K BASIC programlama modu. Hesap makinesi, RAM disc, 2 joystick çıkışı, 170 KB disket kapasitesi, paralel yazıcı çıkışı.

AMSTRAD'IN TURKIYE'DE TEK YETKILI TEMSILCISI

EKAKOMP EKAKOMP EKAKOMP

EKAKOMP BİLGİSAYAR SAN. ve TİC. A.Ş. MECLİSİ MEBUSAN CAD. SOMER HAN. No: 81-83. FINDIKLI - ISTANBUL. TEL: 151 37 24-25. TELEX: 25023 EKOP TR.