YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Ms-Dos ve Xen. İşl. Sist. ve Bas. Cob. Prog. Dil.

Yüksek Lisans Tezi

İbrahim Emiroğlu

1990

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MS-DOS ve XENIX İŞLETİM SİSTEMLERİ VE BASIC , COBOL PROGRAMLAMA DİLLERİ

YÜKSEK LİSANS TEZİ

MATEMATİK MÜH. İBRAHİM EMİROĞLU

İSTANBUL 1990

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

değerli öğrencilerimizden
Abdulkadir Tepeli'ye
mutluluk, başarı dileklerimle

26.03.1990

MS-DOS ve XENIX İŞLETİM SİSTEMLERİ VE BASIC , COBOL PROGRAMLAMA DİLLERİ

YÜKSEK LİSANS TEZİ

MATEMATİK MÜH. İBRAHİM EMİROĞLU

İSTANBUL 1990

İÇİNDEKİLER

I. BÖLÜM

DOS İŞLETİM SİSTEMİ

SAYFA NO

SAYFA NO

SAYFA NO

SAYFA NO

SAYFA NO

SAYFA NO

SAYFA NO

SAYFA NO

TEŞEKKUR

Bu tezin hazırlanmasında değerli fikirleri ile bana yardımcı olan çalışmamı sağlayarak, yakın ilgi ve desteğiyle her sorunuma iyi niyetle çözüm bulan , hocam , Sayın Yrd.Doç.Dr.Hüseyin BAŞLIGİL'e çalışmam boyunca bana destek olan , Hocam , Sayın Doç.Tahir Şişman'a ve diğer hocalarıma , değerli arkadaşlarıma bu vesile ile teşekkür edebilmek benim için büyük bir görev ve mutluluktur.

ÖNSÖZ

Türkiye'de bilgisayar gün geçtikçe büyük ilgi uyandırmaktadır. Özellikle mühendislik eğitiminde bilgisayar büyük tatbik alanı bulmuştur.

Bilgisayarları en verimli şekilde kullanabilmek için, onlarda kullanılan işletim sistemlerini ve bilgisayar programlama dillerini iyi bir şekilde bilmek gerekir.

Bu tezin hazırlanmasındaki amaç , DOS ve XENIX işletim sistemleri ve yüksek seviyeli programlama dillerinden BASIC ile COBOL dillerini ayrıntıları ile incelemektir. Tez bu sıra dahilinde geliştirilmiş ve verimli bir çalışmanın ürünü olarak ortaya çıkmıştır.

PREFACE

Computer in Turkey make a great effections on Turkish people day after day. Especially engineering computer has found a long aplication area.

For using computers most productively , we most know their operating system and programming languages pretty well.

The purpose of preparing this thesis are abserving DOS and XENIX opreating systems , BASIC and COBOL high programming laguages more detailed. Thesis was improved in this form and appeared to be productive working product.

I. BÖLÜM
DOS İŞLETİM SİSTEMİ

I.1 DOS İŞLETİM SİSTEMİNE GİRİŞ

Bu bölümde herhangi bir DOS disketi ile yapılabilecek işlemler açıklanacaktır. DOS disketi ile yapılabilecek işler aşağıda sıralanmıştır.

I.1.1. Kütüklerin karşılaştırılmasında , kopyalanmasında görüntülenmesinde , silinmesinde , yeni isim verilmesinde kullanılabilir ,

I.1.2. Sabit disk veya disketin formatlanmasında ,

I.1.3. Farklı yazıcı ve ekran seçeneklerinin ayarlanmasında kullanılabilir.

I.1.4. Sistemin herhangi bir tuşa basıncaya kadar durdurulmasında,

I.1.5. Zarar görmüş bir disk veya disketten beilirli bir kütüğün yeniden kurtarılmasına, istenir ise , zarar görmüş disk veya disketin tamamıda kurtarılabilir.

I.1.6. Ekranın grafik çizimi yazıcıya aktarmak istenildiği zaman ,

I.1.7. Sistem başka bir iş yaparken yazıcıda kütüklerin yazdırılmasında ,

I.1.8. Sabit disk üzerinde kütüklerin kopyalanmasında ve düzenlenmesinde ,

I.1.9. Belirli bir kütüğün bir bölümünün kütük içerisinde araştırılmasına,

I.1.10. Bir defada ekran dolusu verinin görüntülenmesinde,

I.1.11. Yeni sistem prompt'unun ayarlanmasında ,

I.1.12. Uzantısı .EXE olan kütüklerin uzantısının .COM'a çevrilmesinde ,

I.1.13. Sadece okunur kütük yapmada ,

I.1.14. Herhangi bir anda bir program veya komuttan çıkışta CTRL-BREAK komutunun DOS ile kontrol edilmesinin sağlanmasında ,

I.1.15. Alt directory'lerin oluşturulmasında ve silinmesinde

I.1.16. Disk veya disketteki tüm dizinlerin görüntülenmesinde

I.1.17. Disk hatalarının kontrol edilmesinde ,

I.1.18. Tarih ve zamanın ayarlanmasında ,

I.1.19. Tarih ve zaman formatlarının seçilmesinde DOS komutları kullanılabilir.

I.2. DİSKET SÜRÜCÜLERİNİN TİPLERİ

Üç tip disket sürücüsü vardır. Bunlar ;

I.2.1. Tek Taraflı Disket Sürücüleri : Bu tip disket sürücüleri 160KB/180KB 'a kadar formatlanmış disketleri okuyabilirler.

I.2.2. Çift Taraflı Disket Sürücüleri : Bu tip disket sürücüleri 320KB/360KB 'a kadat formatlanmış disketleri okuyabilirler.

I.3.3. Yüksek Kapasiteli Disket Sürücüleri : Bu tip disket sürücüleri 1.2MB'e kadar formatlanmış disketleri okuyabilirler.

I.3. DİSKET TİPLERİ

I.3.1. Tek Taraflı (160KB/180KB) : Bu tip disketlerde 40 iz (track) her izde de 8/9 sektör bulunmaktadır ve 160kb/180kb'ye kadar bilgi içerebilme kapasitesi vardır. (1K=1024byte).

I.3.2. Çift Taraflı (320KB/360KB) : Bu tip disketlerde 40 iz (track)) her izde de 8/9 sektör bulunmaktadır ve 320KB/360KB'e kadar bilgi içerebilme kapasitesi vardır. (1K=1024byte).

I.3.3. Yüksek kapasiteli (1.2MB) : Bu tip disketlerde 80 iz (track) her izde de 15 sektör bulunmaktadır. 1.2MB 'te kadar bilgi içerebilme kapasitesi vardır. (1M=1.048.576byte).

I.4. DİSKETLER VE SÜRÜCÜ UYUMLULUĞU

Değişik disket ve sürücü tipleri arasında okuma ve yazma için bazı kombinasyonlar vardır. Bunlar :

* Tek taraflı sürücülerde tek taraflı disketleri okuyup üzerine kayıt yapabiliriz.
* Çift taraflı sürücülere tek tek taraflı ve çift taraflı disketleri okuyup üzerine kayıt yapabiliriz.
* Yüksek kapasiteli sürücülerde tek taraflı,çift taraflı ve yüksek kapasiteli disketleri okuyup üzerinde işlem yapabiliriz.

I.5. KÜTÜK TANIMLANMASI

Kütük belirleme , DOS 'a tanımlanmış kütüğü arayacağı yeri söyler. Bir file tanımı 3 parçadan oluşmaktadır.

I.5.1. Sürücü Adı : Sürücü tanımlayıcısını gösterir. Bir sürücü belirlemek için bir sürücü harfi ve onu izleyen 2 nokta üstüste girilir.

Örnek 1.5.1.a

A: Sistemdeki A sürücüsünü , B: B sürücüsünü , eğer bu parametre kullanılmaz ise DOS varsayılan sürücüyü kullanır.

I.5.2. Kütük Adı : 1-8 karakterler arasında olabilir. Bir kütük adı yazıldığında DOS karakterlerin geçerliliğini kontrol eder.

I.5.3. Kütük Adı Eki (.EXT) : Bir noktayı izleyen en fazla üç karakterden oluşur.

Örnek 1.5.3.a.

.BAS , .COM , .EXE , .PAS , .COB

I.6. DOS AYGIT ADLARI

Belirli isimlerin DOS 'da özel anlamları bulunmaktadır. Bunlar DOS aygıt isimleri olarak adlandırılırlar.

CON : Konsol klavye/Ekranı , giriş aygıtı olarak adlandırılır.

Bu kullanıma son vermek için F6 tuşuna veya CTRL-Z yaparak ENTER 'e basmalıyız.

LPT1 veya PRN : İlk paralel yazıcı.

I.7 GLOBAL KÜTÜK ADI KARAKTERLERİ

İki özel karakter ? ve * kütük adı ve onun uzantısı içinde bulunabilir.

I.7.1. ? karakteri : bir kütük adı veya uzantısında bulunan ? , herhangi bir karakterin o konumu doldurabileceğini gösterir.

Örnek I.7.1.a

A>dir g?.exe

yazıldığı takdirde 2 karakterli ve g ile başlayan ve uzantısı .exe olan kütükleri listeler.

I.7.2. * karakteri : Bir kütük adı veya kütük adı uzantısında bulunan * herhangi bir karakterlerin kütük adında veya uzantısında bulunan diğer konumları işgal

edebileceğini gösterir.

Örnek I.7.2.a

A>DIR *.EXE

yazıldığı takdirde A sürücüsünde bulunan disket içerisinde uzantısı .EXE olan tüm kütükler listelenir.

I.8 SABİT DİSKİN HAZIRLANMASI

DOS işletim sistemi ile çalışan bilgisayarlarda sabit disk bulunuyor ise , sabit diskin kullanılabilmesi için hazırlanması gerekir. Eğer sabit diski kullanıma hazır hale gelmeden kullanılmaya çalışılır ise ekranda ;

Invalid drive specification

şelinde bir mesaj belirir. Bu mesajda geçersiz sürücü tanımı yapıldığı belirtilmektedir. Sabit diski hazırlayabilmek için DOS komutlarından FDISK komutu kullanılır. Sabit diskin hazırlanmasında FDISK komutunun kullanılabileceği yerler şunlardır ;

* DOS bölümlerinin oluşturulması ,
* İşlek bölümü değiştirmek ,
* DOS bölümünün silinmesi ,
* DATA bölümünün görüntülenmesi ,
* Sonraki sabit disk sürücüsünün belirlenmesi ,

FDISK komutu daha ayrıntılı bir şekilde ileriki konularda açıklanacaktır.

I.8.1 Sabit Disk Sürücü Tanımlayıcıları

A ve B harfleri disket sürücüleri için kullanılır. C , D veya E harfleri ile sabit diskler için sürücü tanımlamaları yapılır. Eğer bilgisayarın tek disket sürücüsü ve bir disk sürücüsü var ise var ise , disket sürücüsü olarak A ve B harfleri , disk sürücüsü olarak C harfi kullanılır. Eğer bilgisayarda 2 disket sürücüsü ve 2 sabit disk var ise , A ve B harfleri disket sürücülerini , C ve D harfleri ise disk sürücü adını alır.

I.9. CONFIG.SYS (GÖRÜNÜM) KÜTÜĞÜ VE ÖZELLİKLERİ

CONFIG.SYS kütüğü sistemin görümleneceği şekli açıklar.

DOS disketinde CONFIG.SYS kütüğü EDLIN editörü veya COPY CON deyimi ile açılabilir. Bu işlem aşağıda belirtildiği şekilde yapılabilir.

A>COPY CON CONFIG.SYS

yazılıp RETURN'e basılır. Config.sys görünüm kütükleri yazılır. Yanlız her komuttan sonra RETURN tuşuna basılmalıdır. Görünüm komutları yazıldıktan sonra F6 tuşuna veya Ctrl-Z tuşuna basılarak DOS disketinde CONFIG.SYS kütüğü oluşturulur. CONFIG.SYS kütüğüne yazılacak komutlar sistem tekrar açılıncaya kadar geçerli olmaz. Çünkü DOS , işletim sistemi her yüklenişinde CONFIG.SYS kütüğü arar ve bu kütük bulunduğu takdirde CONFIG.SYS kütüğü içerisindeki komutları yorumlar. Eğer DOS disketinde CONFIG.SYS kütüğü bulunmamış ise DOS görünüm kütükleri için varsayışan değerleri işleme katar.

Bir CONFIG.SYS Kütüğü İçerisinde Bulunabilecek Görünüm Kütüklerinin Başlıcaları Şunlardır.

1.9.1 BREAK Komutu

Herhangi bir anda DOS'dan bir işlem yapılamsı gerektiğinde programdan kaçarak DOS'a komut etme imkanını verir.

Genel Yazılım Formatı :

BREAK=ON/OFF

Bu komutun default (yazılmadığı taktirde) değeri BREAK=OFF 'tur. BREAK=OFF yazılmış ise standart çıkış işlemleri , standart giriş işlemleri , standart yazım işlemleri , standart yardımcı işlemler yapılmak istendiği takdirde CTRL-BREAK tuşu ile programın akışı kırılabilir.

BREAK=ON yazılmış ise istenen her zaman CTRL-BREAK tuşu ile DOS kontrol edilebilir. Bu şekildeki yazılım şekli program derlendiği anda herhangi bir hata veya döngü ile karılaşıldığı zaman CTRL-BREAK ile program kırılabilir.

1.9.2. BUFFERS Komutu

DOS'un bellek içerisinde yer ayıracağı disk arabelleklerinin sayısını belirleme imkanı verir.

Genel Yazılım Formatı :

BUFFERS=x

burada x ile belirtilen sayı 1 ile 99 sayısı arasında bir tamsayıdır. Bu komut yazılmadığı takdirde yani default değeri bazı makinalar için 2 bazıları için 3'tür.

Ara bellek DOS'un bir disket veya diskten okunan veya

bir disk yada diskete yazılan bilgileri saklaması için kullandığı bir bellek bloğudur. Eğer kütükte 128 byte tutarında bilgi okundu ise , DOS bu bilgiyi kendi ara belleklerinden birine yerleştirecek ve daha sonra bu bigiyi ara bellekten uygulama alanına aktaracaktır. Bu işlem yapıldıktan sonra kullanılan arabellek kullanıldığı anlaşılsın diye işaretlenir. DOS daha sonraki bilgi aktarımı için başka bir ara bellek kullanacaktır. Bu süre sonra tüm arabellekler önceden kullanılmış bilgileri kapsayacaktır. DOS'da ne kadar çok ara bellek bulunuyor ise belleğe konacak bilgi tutarı o denli artar.

Disk veya diskette herhangi bir kayıt okunmak yada yazılmak istendiği takdirde DOS'un yapacağı ilk işlem okunmak yada yazılmak istenen kayıdın bir ara bellekte olup olmadığını araştırmaktır. Aranan kayıt ara bellekte mevcut ise DOS disk veya disketten sektörü okumaya gerek duymadan uygulama alanına aktarır. Kayıt ara bellekte mevcut değil ise DOS sektörü disk veya disketten okuyacak veya disk yada diskete yazacaktır.

Kayıtların rastgele okunduğu ve yazıldığı uygulamalarda (bir çok BASIC , COBOL , PASCAL uygulamalarında olduğu gibi) bir ara belleğe önceden yazılmış olan doğru kaydı bulma ihtimali ara bellek sayısı ne kadar çok olur ise o derece artış gösterir.

Uygulamaların hepsi birbirinden farklı olduğu için bunların hepsi için uygun olacak ara bellek sayısını bulmak mümkün değildir. Herhangi bir uygulama için uygun olabilecek ara bellek sayısı denemek yolu ile bulunabilir. Ancak çalıştırılacak program çok sayıda kayıt okuma ve yazma işlemi yapıyor ise DOS ara bellek sayısı 10 ile 20 arasında seçildiğinde en uygun çözümü verir.

Çok sayıda ara bellek kullanıldığı takdirde sistemin işleyişi yavaşlayabilir. Bunun nedeni aranan kayıdın ara bellekten bulununcaya kadar geçen sürenin , diskten veya disketten okununcaya kadar geçen süreden daha fazla olmasıdır.

Sonuç olarak BUFFER komutu bir disketi okumadan veya

yazmadan önce DOS'un geçici olarak veri depoladığı bellek içindeki alanı açar.

I.9.3. COUNTRY Komutu

Kullanılmak istenen ülkenin tarih ve zamanını ayarlar.

Genel Yazılım Formatı :

COUNTRY=xxx

yukarıda belirtilen xxx 3 haneli uluslarası kod dur. Default değeri (yazılmadığı takdirde alacağı değer) A.B.D. ülke kodo olan 001 'dir.

Bazı ülkelerin ülke kodları aşağıda verilmiştir.

ÜLKELER	ÜLKE KODU
A.B.D	001
HOLLANDA	031
BELÇİKA	032
FRANSA	033
İSPANYA	034
İTALYA	039
İSVİÇRE	041
İNGİLTERE	044
DANİMARKA	045
İSVEÇ	046
NORVEÇ	047
ALMANYA	049
AVUSTURYA	061
FİNLANDİYA	358
İSRAİL	972

I.9.4 DEVICE Komutu

Tanımlanan bir kütük adı ile bir disket sürücüsünün belirlenmesi imkanı verir. Bu kütükler RAMDISK.SYS , ANSI.SYS , VDISK.SYS kütüklerinden birisi olabilir.

Genel Yazılım Formatı :

DEVICE=Kütük adı

CONFIG.SYS kütüğüne DEVICE=RAMDISK.SYS /200 ilavesi yapılır ve makina yeniden açılır ise Floppy diskli bir makina için A,B sürücülerine ilaveten 200K 'lık bir C sürücüsü (Ramdisk açar) açar.

CONFIG.SYS kütüğüne DEVICE=VDISK.SYS 400 512 64 ilavesi yapılır ve makina yeniden açılır ise Floppy diskli bir makina için A,B sürücülerine ilaveten 512 byte kesim ile 400K 'lık bir C sürücüsü açar. Config.sys kütüğüne bu yazılıp sistem tekrar açıldığında VDISK.SYS şu mesajı verir.

VDISK Version 1.0 virtual disk C:
Buffer size:400
Sector size:512
Directory entries:64

bu iletiler floppy diskli makina için açılan sürücü adını , kesim ölçüsünü , sürücünün kullanabileceği byte miktarını ve dizin girişlerinin sayısını belirlemektedir.

I.9.5. FBCS Komutu

Makina açıldığında DOS'un kontrol edebileceği kütük kontrol bloklarının sayısının belirlenmesini sağlar.

Genel Yazılım Formatı :

FBCS=m,n

m: 1 defada açılabilecek FBCS ile açılmış kütüklerin toplam sayısını verir. Default (FBCS komutu kullanılmadığı takdirde) değer 4'tür. m sayısının değeri 1 ile 255 sayıları arasında bir değerdir.

n: 1 programda FBCS ile açılmış m 'den fazla kütük kullanılmak isteniyor ise DOS ile otomatik olarak kapatılmayacak ve FBCS ile kapatılmış kütüklerin sayısını belirler. Default (yazılmadığı takdirde alacağı değer) 0'dır.

CONFIG.SYS kütüğüne ;

FBCS=3,1

şeklinde bir ifade edilecek olur ise bir defada açılabilecek FBC kütüğü 3 , FBC ile otomatik olarak kapatılacak kütük sayısı 1 tanedir.

I.9.6 FILES Komutu

Aynı anda açılabilecek maksimum kütük sayısını belirler. Default değer (yazılmadığı takdirde alacağı ilk değer) 8'dir.

Genel Yazılım Formatı :

FILES=x

burada x ile belirtilen sayı aynı anda açık tutulabilecek kütük sayısıdır. Bu sayı 8 ile 255 sayısı arasındadır.

Config.sys kütüğüne FILES=17 ilavesi yapılacak olur ise DOS aynı anda 17 tane kütüğü açık bulundurabilecektir.

I.9.7. LASTDRIVE Komutu

Erişilebilecek maksimum sürücü adı harfinin belirtilmesinde kullanılır.

Genel Yazılım Formatı :

LASTDRIVE=x

burada x A ile Z arasında herhangi bir alfabetik harf olabilir. Bu komut ile DOS tarafından geçerli olabilecek son sürücü adını belirlenir. DOS'un erişebileceği kütük sayısını 17 ile sınırlamak ister isek CONFIG.SYS kütüğüne LASTDRIVE=R yazılmalıdır. Bu durumda DOS'un kabul edebileceği son sürücü harfi R dir.

I.10. DOS KOMUTLARININ ÇEŞİTLERİ

DOS komutları iki çeşittir. Bunlar iç (Internal) komutlar ve dış (External) komutlardır.

I.10.1 İÇ (İNTERNAL) KOMUTLAR

DOS'un içerisinde bulunan komutlardır. Bu tip komutları kullanabilmek için disket sürücüsünde DOS disketinin bulunmasına gerek yoktur.

Örnek I.10.1.a.

CLS komutu ekran temizler ,

ERASE silinmek istenen kütükleri siler ,

DIR disk veya disket içerisindeki dizinleri listeler ,

I.10.2. DIŞ (EXTERNAL) KOMUTLAR

Bu komutlar program kütükleri gibi disk veya disket üzerinde bulunurlar. Bu komutları kullanabilmek için kullanılacak komutları içeren disketin sürücüde bulunması gerekir. Dış komutların kullanılmasında kütük adı uzantıları kullanılmazlar. (Bu komutları kullanabilmek için DOS ortamında iken kütük adı yazılıp RETURN'a basılması yeterlidir.)

Dış komutlar kullanılır iken kütük adı uzantısının belirtilmasine gerek yoktur.

Örnek I.10.2.a.

FORMAT disket formatlamak için kullanılır.

BACKUP disketleri yedeklemek için kullanılır.

I.11. DOS KOMUTLARI

I.11.1 DATE Komutu

DOS 'a bilinen bir tarihin girilmesinde veya değiştirilmesinde kullanılan DOS komutudur. Diskette veya diskte bir kütük açıldığında veya değiştirildiğinde , dizin içerisinde kütüğün yanında , kütüğün değiştirildiği veya açıldığı tarihi yazar. Bir iç komuttur.

Genel Yazılış Formatı :

DATE [mm-dd-yy]/[dd-mm-yy]/[yy-mm-dd]

mm : Yılın kaçıncı ayı içerisinde olunduğunu belirlemek içindir. 1 ile 12 sayıları arasında olabilir.

dd : İçerisinde bulunulan ayın kaçıncı gününde olunduğunu bildirmek içindir. 1 ile 31 sayıları arasında olabilir.

yy : Yılı belirtmek içindir. İçerisinde bulunan yıl 4 rakamlı veya içerisinde bulunulan yılın son 2 rakamı yazılabilir.

Eğer DATE komutu hiçbir parametre kullanılmadan kullanılır ise ekranda aşağıdaki cevap bekleyen ileti görüntüye gelir.

```
A>DATE
Current date is Tue 1-01-1980
Enter new date (mm-dd-yy) :
```

Burada tarih için format mm-dd-yy , dd-mm-yy veya SELECT komutu ile seçilmiş olan ülke koduna göre yada CONFIG.SYS kütüğünün COUNTRY alt komutunda verilen ülke koduna göre belirlenir. Tarih girilirken parçaları ayırmak için (-) , (.) , (/) işaretlerinden herhangi biri kullanılabilir.

Örnek I.11.1.a.

1.Haziran 1989 tarihini bilgisayara girmek isteyelim. Bu tarih aşağıdaki şekillerde girilebilir..

```
A>DATE
Current date is Tue 1-01-1980
Enter new date (mm-dd-yy) :6-1-1989
A>DATE
Current date is Tue 1-01-1980
Enter new date (mm-dd-yy) :6.1.1989
```

```
A>DATE
Current date is Tue 1-01-1980
Enter new date (mm-dd-yy) :6/1/1989
```

Eğer geçerli tarih girilecek olur ise DOS bu tarihi kabul eder. Geçerli tarih yazılmadığı takdirde :

```
A>DATE
Current date is Tue 1-01-1980
Enter new date (mm-dd-yy) :15/1/1989

Invalid date
Enter new date (mm-yy-dd):
```

şeklinde mesaj verir.

Eğer tarihi olduğu gibi bırakmak istenir ise RETURN tuşuna basılmalıdır.

I.11.2 TIME KOMUTU

Sistemin zamanını değiştirmek için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

```
[d:] TIME [hh:mm:[ss.[xx]]]
```

[d:] varsayılan sürücü adı harfi,

hh: saati tanımlamak içindir. 0 ile 23 sayıları arasında herhangi bir sayı olabilir.

mm: dakikayı tanımlamak içindir. 0 ile 59 sayıları arasında herhangi bir sayı olabilir.

ss: saniyeyi tanımlamak içindir. 0 ile 59 sayıları arasında herhangi bir sayı olabilir.

xx: saniyenin yüzde birini tanımlamak içindir. 0 ile 99 sayıları arasında bir sayı olabilir.

Eğer parametreler yazılmadan kulanılacak olunur ise ekranda aşağıdaki görüntü oluşur.

```
Current time is:hh:mm:ss.xx
Enter new time:
```

Eğer geçerli bir zaman girilmedi ise ekranda ;

```
Invalid time
Enter new time:
```

şeklinde bir mesaj görülür.

Eğer zaman olduğu gibi bırakılmak istenir ise ; TIME

komutu icra ettirildikten sonra , yeni zamanı yazmadan RETURN tuşuna basılmalıdır.

I.11.3. CLS KOMUTU

Ekranı temizlemek için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

[d:] CLS

[d:] sürücü adı harfi (A , B , C ....)

Örnek I.11.3.a.

Ekranı tamamen temizlemek istenilir ise

A>CLS

yazıp RETURN tuşuna basılmalıdır. Bu şekilde ekran tamamen temizlenir. Ekranın sol üst köşesinde PROMPT belirlenir.

I.11.4. PROMPT KOMUTU

Yeni bir sistem hatırlatıcısı düzenlemek istenildiği zaman kullanılan bir iç komuttur.

Genel yazılış formatı :

PROMPT [Prompt parametreleri]

Örnek I.11.4.a.

A>PROMPT pgtgdg

yazılışında PROMPT'un görünümü aşağıdaki gibi olur.

A:\>0:17:17:17>THU 6-1-1989>

Eğer A sürücüsündeki disket ESRA adlı alt directory'de bulunsa idi PROMPT'un görünümü aşağıdaki gibi olacaktı.

A:\ESRA>0:17:17:17>THU 6-1-1989>

Örnek I.11.4.b.

Yukarıda verilen PROMPT 'tan normale dönebilmek için aşağıdaki komutu vermeliyiz.

A:\ESRA>0:17:17:17>THU 6-1-1989>PROMPT

yazıp RETURN tuşuna basılır ise PROMPT normale döner. (A> şekline döner)

Prompt parametreleri olarak aşağıdaki parametrelerin bir veya birkaçını kullanabiliriz. PROMPT parameteleri $ işaretinin sağına yazıldığında geçerli olurlar.

$	$ karakterini PROMPT 'a verir.
t	PROMPT'ta zamanı gösterir.
d	PROMPT'ta günün tarihini gösterir.
p	PROMPT'ta varsayılan sürücünün izlemekte

olduğu dizin görülür.

n varsayılan sürücü harfi.

g ">" karakterini PROMPT'a verir.

b "|" karakterini PROMPT'a verir.

q "=" karakterini PROMPT'a verir.

h Bir önceki karaktere dönüş yapma ve onu silmek için kullanılabilir.

e ESC (ESCAPE) karakterini PROMPT'a verir.

Prompt komutu yukarıda belirtilen karakterlerden farklı her karakteri değersiz kabul eder. PROMPT ile beraber herhangi bir parametre belirtilmemiş ise normal DOS sistem hatırlatıcısı olan np (A>) kullanılır.

I.11.5. PATH KOMUTU

Bilgisayara yüklenmek istenen DOS kütüklerinin aranacağı sürücüler veya alt directory'leri belirtmek için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

[d1:]PATH=[kütüklerin aranacağı sürücü adı veya alt directory adları]

[d1:] varsayılan sürücü adı,

[kütüklerin aranacağı sürücü adı veya alt directory adları] ile yüklenmek istenen kütüklerin aranacağı yol belirtilmektedir. Bu sürücü adları veya alt directory adları (;) işareti ile birbirinden ayrılmalıdır.

PATH komutu parametresiz kullanılır ise ekranda , kütüklerin aranacağı yol görüntülenir.

Örnek I.11.5.a.

Yüklenmek istenen herhangi bir dos komutunu ;

1. A sürücüsünde bulunan disket içerisinde ,
2. A sürücüsünde bulunan disket içerisindeki ESRA adlı alt directory 'de ,
3. B sürücüsünde bulunan disket içerisinde ,
4. C sürücüsü içerisinde bulunan DOS adlı alt directory'de aratabilmek için ;

```
A>PATH=A:\;A:\ESRA;B:\;C:\DOS;
```

komutunun icra ettirilmesi gerekir.

Bu komut icra ettirildikten sonra GWBASIC programı bilgisyara yüklenmek istenilir ise , GWBASIC adlı program A sürücüsünde bulunan disket içerisinde aranır. Bu disket içerisinde bulunur ise program yüklenir. Eğer program bulunamaz ise program , A sürücüsünde bulunan disket içerisindeki ESRA adlı alt directory'de aranır. Oradada bulunmaz ise B sürücüsünde bulunan disket içerisinde aranır. Yine bulunmaz ise C sürücüsünde bulunan DOS adlı alt directory'de aranır.

I.11.6. SET KOMUTU

Komut işleticisi ortamına dizgi yerleştiren bir iç komuttur. DOS girilen her PROMPT ve PATH komutunu komut işleticisi ortamına yazar. Bunun için PATH ve PROMPT komutlarının ortama yazılması için SET komutunun kullanılmasına gerek yoktur.

Genel Yazılım Formatı:

[d1:]SET [isim=parametreler]

[d1:] varsayılan sürücü adı ,

Bazı programlar üzerinde bir takım işlemler yapıldıktan sonra tekrar DOS ortamına çıkabilmek için A sürücüsüne DOS disketinin takılmasını isterler. Bu durum karşısında A sürücüsüne DOS disketi takılıp RETURN tuşuna basılır ise DOS ortamına (Prompt'a) çıkılır.

SET komutu kullanılarak , DOS disketi isteme işlemi A sürücüsünde değilde B sürücüsünde bulunan disket veya herhangi bir sürücüde disk veya disketin bir alt directory'sinden istetilebilir. Bunu sağlamak için SET komutu ile COMPSEC komutu kullanılmalıdır.

Eğer COMMAND.COM programının B sürücüsünde bulunan disket içerisinde aranılması isteniyor ise ;

A>SET COMPSEC=B:\COMMAND.COM

komutu , COMMAND.COM programının C diskinin DOS adlı alt directory'sinde aranması isteniliyor ise ;

A>SET COMPSEC=C:DOS\COMMAND.COM

komutu icra ettirilmelidir.

SET komutu parametresiz icra ettirilecek olunur ise komut işleticisi ortamında bulunan diziler ekranda görüntülenir.

Örnek I.11.6.a.

A>SET

yazılıp RETURN'a basılır ise komut işleticisi ortamında bulunan dizgiler ekranda görüntülenir. Bu dizgiler PATH , PROMPT , COMPSEC komutlarının aldığı değerlerdir.

I.11.7. VER KOMUTU

Bilgisayarın açıldığı DOS disketinin versiyonunu bildiren bir iç komuttur. DOS versiyonu ,tek basamaklı asıl versiyon sayısı ve bu sayının arkasından gelen nokta ile verilmiş olan 2 basamaklı ikincil versiyon sayısını içerir.

Genel Yazılım Formatı :

[d:] VER

[d:] varsayılan sürücü harfi adı ,

Örnek I.11.7.a.

3.30 DOS disketi ile açılmış olan bir makina için ;

A>VER

komutu icra ettirilir ise ekranda ;

IBM Personal Computer DOS Version 3.30

şeklinde bir mesaj iletisi görülür.

Makinanın açılmış olduğu DOS disketinin versiyonu 3 ve ikincil DOS versiyonu 30 dur.

I.11.8. VERIFY KOMUTU

Bir disk veya disket üzerine yazılacak olan bilginin doğru kayıt edilip edilemiyeceğini kontrol etmek için kullanılan bir iç komuttur.

Genel Yazılım Formatı:

[d:]VERIFY [ON/OFF]

[d:] varsayılan sürücü adı,

VERIFY ON komutu kullanılıyor ise , DOS disk veya diskete yazılacak olan bilgilerin hatasız okunabileceğini doğrulamak amacı ile bir sağlama işlemi yapar. Bu işlem fazladan zaman aldığından bilgiler disk veya diskete yazılırken sistemin işleyişi yavaşlayacaktır.

VERIFY OFF komutu kullanılıyor ise , disk veya diskete yazılacak olan bilgilerin doğru okunup okunmayacağı DOS tarafından kontrol edilmez.

VERIFY komutu parametresiz olarak kullanılır ise , VERIFY

komutunun o anki durumu ekranda görüntülenir.
VERIFY komutunun default (hiç kullanılmadan önceki durumu) değeri VERIFY OFF 'tur.
Örnek I.11.8.a.
Disk veya disket içerisine yazılacak olan bilgilerin doğru olarak okunup okunamayacağı DOS tarafından kontrol edilmek istenir ise ;

A›VERIFY ON

komutu icra ettirilmelidir.

I.11.9. VOL KOMUTU

Belirtilen disk veya disket sürücüsünde bulunan disk veya disketin adını ekranda görüntüleyen bir iç komuttur.
Genel Yazılım Formatı :

[d1:]VOL [d2:]

[d1:] varsayılan sürücü harf adı ,
[d2:] adı öğrenilmek istenen disk veya disket sürücüsünün harf adı ,
Örnek I.11.9.a.
A sürücüsünde bulunan disk veya disketin adı öğrenilmek istenir ise ;

A›VOL

komutu icra ettirilmelidir. Bu komutun icrası sonucu ekranda ;

Volume in drive A is ‹Disket adı›

şeklinde bir mesaj görüntülenir.

I.11.10. LABEL KOMUTU

Disk veya disketin adının değiştirilebilmesini sağlayan bir dış komuttur.
Genel Yazılım Formatı :

[d1:]LABEL [d2:]‹disketin ismi›

[d1:] LABEL kütüğünün bulunduğu disk veya disket sürücüsünün harf adı.
[d2:] ismi değiştirilmek istenen disk veya disket sürücüsünün harf adı.
LABEL komutu parametresiz olarak kullanılır ise ekranda ;

Volume in drive [d:] is ‹disketin eski adı›
Volume label(11 Characters, ENTER for none)?

şeklinde bir mesaj görüntülenir. Burada [d:] ile adı değiştirilmek istenen disk veya disket sürücüsünün harf adı belirtilmektedir. Girilecek yeni isim 11 karakteri aşmamalıdır.

Örnek I.11.10.a.

A sürücüsünde adı MATEMATIK olan disketin adını ESRA yapabilmek için ;

A>LABEL A:

komutu icra ettirilmelidir. Bu komut icra ettirildiği takdirde ekranda ;

Volume in drive A is MATEMATIK

Volume label(11 Characters, ENTER for none)?

şeklinde bir mesaj belirtilir. Burada soru işaretinin yanına disketin yeni adı ESRA yazılıp RETURN tuşuna basılır ise disketin adı ESRA olarak değiştirilmiş olur.

I.11.11. BREAK KOMUTU

Çalışmakta olan bir programın CONTROL-BREAK tuşu ile kırılıp kırılamayacağı zamanları belirlemek için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

[d:] BREAK=[ON/OFF]

[d:] Prompt ile belirtilen sürücü adı harfi.

Eğer BREAK=ON parametresi verilmiş ise bir programın herhangi bir zamanda CTRL-BREAK ile programın kırılmasını sağlar.

Eğer BREAK=OFF parametresi verilmiş ise , bir programın akışı yanlızca ;

Standart giriş işlemlerinde ,
Standart çıkış işlemlerinde ,
Standart yazıcı işlerinde ,
Standart yardımcı işlemlerde ,

CTRL-BREAK ile kırılabilir.

BREAK=ON veya BREAK=OFF yazılmadığı takdirde (Default Değer) BREAK=OFF 'tur.

Eğer BREAK komutu parametresiz kullanılır ise BREAK komutunun o anki değeri ekranda :

BREAK=ON veya BREAK=OFF

şeklinde bir mesaj verir.

Örnek I.11.11.a.

Eğer program çalışır iken herhangi bir anda programın çalışması Ctrl-BREAK tuşu ile kırılması isteniyor ise ;

A>BREAK=ON

yazılıp RETURN tuşuna basılmalıdır. Bu durumda herhangi bir anda Ctrl-BREAK tuşuna basılır ise programın icrası kırılır ve sisteme çıkılır.

Örnek I.11.11.b.

A>BREAK=OFF

yazılıp RETURN tuşuna basılır ise programın icrası ancak ;

Standart giriş işlemlerinde ,
Standart çıkış işlemlerinde ,
Standart yazıcı işlerinde ,
Standart yardımcı işlemlerde ,

işlemleri sırasında Ctrl-BREAK tuşu ile kırılabilir.

I.11.12. DIR KOMUTU

Disk veya disket içerisinde bulunan kütükleri listelemek için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

[d1:]DIR [d2:]‹fileadı.fileuzantısı][/P][/W]

[d1:] varsayılan sürücü adı ,

[d2:] kütüğü veya kütükleri listelenmek istenen disk veya disket sürücüsünün adı ,

Eğer ‹fileadı.fileuzantısı› belirtilmiş ise sadece belirtilen file adı ve file uzantısı listelenir.

[/P] ekran dolduğu zaman görüntülemeye ara verir. Eğer listelenecek kütük sayısı bir ekran dolusundan fazla ise ekranda ;

Strike a key when ready...

şeklinde bir mesaj görülür. Bu mesajda , kütüklerin devamı görülmek isteniyor ise herhangi bir tuşa basılması gerektiği belirtilmektedir.

[/W] geniş görüntü formatı şeklinde disk veya disketin içindeki kütükleri görüntüler.

Kütük adı ve uzantısı parametreleri belirtilirken , ? ve *

global kütük karakterleri kullanılabilir.

Örnek I.11.12.a.

A sürücüsünde bulunan kütüklerin tamamı listelenmek isteniyor ise ;

A>DIR

komutunun icra ettirilmesi gerekir.

Örnek I.11.12.b.

A sürücüsünde bulunan ve uzantısı .BAS olan kütükler listelenmek istenir ise ;

A>DIR *.BAS

komutu icra ettirilmelidir.

Örnek I.11.12.c.

B sürücüsünde bulunan disket içerisindeki kütükleri geniş sayfa düzeninde görüntülemek istenir ise ;

B>DIR /W

veya

A>DIR B:/W

komutlarından birisi icra ettirilmelidir.

Örnek I.11.12.c.

A sürücüsünde bulunan disket içerisinde bulunan kütükleri yazıcıya aktarmak için ;

A>DIR >PRN

komutu icra ettirilmelidir.

I.11.13. FORMAT KOMUTU

Tayin edilen veya varsayılan sürücüdeki disk veya disketi DOS işletim sisteminin kabul edebileceği bir kayıt biçimine hazır hale getirmek için kullanılan dış komuttur. Disk veya disket formatlanırken, disk veya disket üzerinde bulunabilecek hataları analiz eder ve hatalı alanın miktarını byte olarak verir. Formatlama esnasında disk veya disket içerisinde bulunan tüm bilgiler zedelenir (silinir). Bu yüzden bir disk veya disket formatlanırken çok dikkatli davranılmalıdır.

Genel Yazılım Formatı :

[d1:] FORMAT [d2:][/S][/1][/8][/V][/B][/4]

[d1:] FORMAT adlı DOS kütüğünün bulunduğu sürücünün harf adı,

[d2:] Formatlanacak disk veya disketin bulunduğu sürücünün harf adı .
[/S] , d1 ile belirtilmiş sürücüde bulunan disk veya disketten , d2 ile belirtilen sürücüde bulunan disk veya diskete IBMBIO.COM , IBMDOS.COM ve COMMAND.COM kütüklerinin kopyalanmasını sağlar. d1 ile belirtilen sürücüde bulunan disk veya disket içerisinde IBMBIO.COM , IBMDOS.COM ve COMMAND.COM kütükleri bulunmuyor ise , d1 ile belirtilen sürücüye bu kütükleri içeren bir disketin takılması gerektiğini belirten bir mesaj verir.
[/1] , sürücü tipine bakılmaksızın bir disketin tek taraflı kullanımında disketi formatlamak içindir.
[/8] , disketi her iz'de (track) 8 kesim olarak formatlamak içindir. Eğer /8 parametresi kullanılmayacak olunur ise her bir iz'in kullanılması 9 kesim olacağı FORMAT komutu tarafından kabul edilir. /8 parametresi ile , her bir iz'de 8 kesim kullanılacağı DOS'a bildirilir.
[/V] Disk veya diskete isim vermek için kullanılır.
[/B] Sistem komutları olan IBMBIO.COM , IBMDOS.COM ve COMMAND.COM için ayrılmış saha ile disketi 8 kesimlik iz'de formatlamak için kullanılır. IBMBIO.COM , IBMDOS.COM ve COMMAND.COM kütükleri varsayılan sürücüdeki disket içerisinde olmayabilir. Bu parametre SYS komutu sayesinde yer alabilecek herhangi bir DOS versiyonunu (1.00 , 1.10 , 2.00 , 2.10 , 3.00 , 3.10 , 3.20 , 3.30 gibi DOS versiyonları) üzerinde disket formatlamak için kullanılır.
[/4] , yüksek kapasiteli sürücü içerisinde çift taraflı disket formatlamak içindir.

I.11.13.1 FORMAT Komutu İle İlgili Notlar

* Kullanılmamış ve yeni disk ve disketler DOS tarafından kullanılır hale getirilmek için formatlanırlar .
* Formatlama esnasında disk veya disket üzerinde bulunan kütüklerin tamamı zedelenir. (yeniden kullanılmaz hale gelir.)
* Formatlama esnasında disk veya disket üzerinde bulunan hatalı iz'lerin üzerine veri kütüğü tahsis edilmesin diye ,

bu hatalı izler işaretlenir.

* IBMBIO.COM ve IBMDOS.COM kütükleri hidden files (saklı kütükler) olduklarından , bu kütükler directory listelenmesinde görülmezler.

* /V parametresi kullanıldı ise formatlama işlemi tamamlandıktan sonra disk veya diskete isim isteyen bir mesaj verilecektir. Bu isim 1 ile 11 karakter arasında olabilir. Kütük adlarında kabul edilen tüm karakterler disk veya diskete isim olarak verilebilir.

* Formatlama işlemi bittikten sonra ekrada ,

Toplam disk alan miktarını ,

Hatalı olarak işlenmiş alan miktarını ,

/S kullanıldı ise DOS işletim sistemine ayrılmış alan ,

Disket içerisine yazılabilecek toplam alan miktarını ,

bildiren bir mesaj belirir.

* Formatlama işlemi sürücü tipine bağlı olarak değişir. Eğer formatlanacak disketin tek yüzü başarılı olarak okunabilir veya kaydedilebilir ise disket tek yüzlü formatlanır. Disket iki yüzlü ve /1 parametresi kullanılmaz ise disket iki yüzlü olarak formatlanır. iki yüzlü formatlanan disketler tek taraflı disket sürücüleri tarafından okunamazlar.

* Formatlama işlemi tek taraflı sürücülerde disket tek taraflı olarak formatlanabilir. Çift taraflı sürücülerde disket tek taraflı ve çift taraflı formatlanabilir. Yüksek kapasiteli sürücülerde disketler tek taraflı , çift taraflı ve yüksek kapasiteli olarak formatlanabilir.

* /S parametresi belirtilmiş ise sistem kütükleri olan IBMBIO.COM , IBMDOS.COM ve COMMAND.COM kütükleri , varsayılan sürücüden formatlanacak disk veya disket içerisine kopyalanır.

Örnek I.11.13.a.

B sürücüsünde bulunan disketi , sistem komutlarını (IBMBIO.COM , IBMDOS.COM ve COMMAND.COM) içerecek ve bu diskete isim verecek şekilde formatlayabilmek için ;

A sürücüsüne IBMBIO.COM , IBMDOS.COM ve COMMAND.COM DOS kütükleri ile FORMAT komutunu içeren disketi takılır ve

A>FORMAT B:/S/V

yazılarak RETURN tuşuna basılır. Daha sonra ekranda ,

Insert new diskette for drive B:
and strike ENTER when ready

şeklinde , B sürücüsüne formatlanacak disket takılıp , ENTER tuşuna basılması gerektiği belirtilen bir mesaj belirir. B sürücüsüne formatlanacak disket takılıp ENTER tuşuna basılır ise ekranda ,

formatting...

(format başladı) şeklinde bir mesaj belirir. Formatlama işi bittikten sonra ekranda , /S parametresi belirtildiği için

formatting... format complete
system transferred

şeklinde formatlama işleminin tamamlandığını ve sistem komutlarının (IBMBIO.COM , IBMDOS.COM ve COMMAND.COM) kopyalandığını belirten bir mesaj verilir. Eğer /S parametresi verilmese idi bu mesaj verilmeyecekti. Daha sonra /V parametresi belirtildiği için ekranda formatlanacak olan diskete isim verilmesi gerektiğini belirten bir mesaj verilir.

Volume label (11 chracter , ENTER for none)?

disketin ismi verilip ENTER tuşuna basılır ise , ekranda toplam disk miktarı , diskette bozuk alan var ise bozuk alan miktarı , sistem komutları için kullanılan toplam miktarı ve disket içerisinde kullanılabilir toplam alan miktarı , aşağıda belirtildiği şekilde byte olarak verilir.

xxxxxx bytes total disk space
xxxx bytes used by systems
xxxx bytes in bad sector
xxxxxx bytes in available on disk

Format another(Y/N)?

Formatlama işlemine son vermek için bu soruya N (Hayır) cevabı verilmelidir. Bu soruya Y (Evet) cevabı verilir ise B sürücüsüne başka bir disket takılması gerektiğini belirten bir mesaj verilir. Böylece bu soruya Y (Evet) cevabı verildiği sürece disket formatlama işlemi devam edecektir.

I.11.14. SYS KOMUTU

IBMBIO.COM ve IBMDOS.COM işletim sistemi kütüklerini belirlenmiş ilk sürücüde bulunan disk veya disket içerisinden , tanımlanmış ikinci disk veya disket sürücüsünde bulunan disk veya diskete aktaran bir dış komuttur.

Genel Yazılım Formatı :

[d1:] SYS [d2:]

[d1:] , SYS komutu ve işletim sistemi kütükleri olan IBMBIO.COM ve IBMDOS.COM kütüklerinin bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , işletim sistemi kütükleri olan IBMBIO.COM ve IBMDOS.COM kütüklerinin yükleneceği disk veya disket sürücüsünün harf adı.

SYS komutu ile COMMAND.COM kütüğü herhangi bir disk veya diskete yüklenemez. Bunun için , sistem kütüklerinin yükleneceği disk veya diskete COMMAND.COM kaopya edilmelidir. SYS komutu ile işletim sistemlerinin yükleneceği disk veya disket daha önce /S parametresi ile formatlanmış olmalıdır.

I.11.15. FDISK KOMUTU

FDISK komutu ile bir disk üzerinde DOS bölümü oluşturmak için kullanılan bir komuttur. FDISK programının yüklenmesi ;

* A sürücüsüne FDISK komutunu içeren disket takılır.

A>FDISK

yazılarak RETURN'a basıldıktan sonra aşağıdaki menü görüntülenir.

```
        IBM personal Computer
        Fixed Disk Setup Program version 3.30
        (C) Copyright IBM Corp. 1983,1984
         FDISK Options
         Current Fixed Disk Drive:
         Choose one of the following :
                1. Create DOS Partition
                2. Change Active Partition
                3. Delete DOS partition
                4. Display Partition Data
```

Yukarıda verilen menü makinaya eğer bir sabit disk takılı ise görüntülenir. Eğer makinaya takılı olan sabit disk sayısı 1'den fazla ise 4 seçeneğe ilaveten 5. tercih hakkı da görüntülenecektir. Bu aşağıdaki gibidir.

5. Select Next Fixed Disk Drive

Bu ikinci sabit disk sürücüsünü tanımlamak için kullanılan bir seçenektir.

1. Creat DOS Partition : Bu seçenek DOS bölümü oluşturulmak istendiği takdirde tercih edilmelidir. Bu seçenek tercih edildiği takdirde aşağıdaki menü ekranda oluşur.

Create DOS Partition

Current Fixed Disk Drive: 1

Do you wish to use the entrie fixed disk
for DOS (Y/N).....?[Y]

a) Sabit Diskin tamamının DOS 'a Ayrılması (Y (evet) cevabı): Eğer sabit diskle kullanılmak istenen yanlızca DOS işletim sistemi ise yukarıda verilen "Do you wish to use the entrie fixed disk" sorusuna Y (evet) cevabı verilir. Bu durumda sabit diskin tamamı DOS işletim sistemi için ayrılır. Aşağıdaki mesaj görüntüye gelir.

System will now restart
(Sistem şimdi yeniden başlayacak)

Insert DOS diskette in drive A:
(DOS disketini A sürücüsüne takın)

Press any key when ready . . .
(Hazır olduğunuzda bir tuşa basınız)

A sürücüsüne DOS disketinizi sürüp herhangi bir tuşa basınız. Bir süre sonra zaman ve tarihin girileceği mesaj görüntülenir. Zaman ve tarih yazılır. Artık sabit diskin DOS bölümü oluşturulmuştur. DOS bölümünü kullanabilmek için sabit diskin DOS bölümüne FORMAT yapılması gerekmektedir. Bu aşağıdaki şekilde yapılır.

A sürücüsüne DOS disketi takılır.

A>FORMAT C:/S/V

yazılıp RETURN'a basılır ve aşağıdaki görüntü ekrana gelir.

Warning , All data on Non-removable
Disk drive C: Will be lost!
Proceed with (Y/N)

Burada sorulan soruya Y (Evet) cevabı verilecek olur ise sabit disk formatlanmaya başlanır. (Formatlama esnasında sabit diskte daha önceden bulunan bilgiler temizlenir.) Y (evet) cevabı verildikten sonra sabit disk üzerindeki kırmızı lamba yanar ve "Formatting.." mesajı görüntülenir. Sabit diskin formatlanması birkaç dakika alır. Daha sonra ekranda şu mesaj gelir.

Formatting...Format complete
System transferred
Volume label (11 characters , ENTER for none)?

Formatlama yapılırken A>FORMAT C:/S/V şeklinde yapıldığı için System transferred (Sistem transfer edildi) mesajı görüntülenir.

Formatlama yapılırken /S parametresi kullanıldığı için IBMBO.COM , IBMDOS.COM ve COMMAND.COM işletim sistemi kütüklerini de diske kopyalar. Formatlama yapılırken /V parametresi kullanıldığı için sistem transfer edildi mesajından sonra uzunluğu 11 karakteri geçmeyecek şekilde bir isim girilmesi beklenir. İsim girilir ve RETURN'a basılır. Eğer sabit diske herhangi bir isim verilmek istenmiyor ise bu kısım RETURN'a basılarak boş geçilebilir.

b) Sabit Diskin Bir Bölümünün DOS 'a Ayrılması (N (hayır) Cevabı) :

Eğer sabit diskin sadece bir bölümünü DOS 'a ayırmak isteniliyor ise ,

Do you wish to use the entrie fixed disk
for DOS (Y/N).....?[Y]

sorusuna N (hayır) cevabı verilir ve RETURN'a basılır ve aşağıdaki görüntü meydana gelir.

Total Fixed Disk space is xxxx cylinders
Maximum avaliable space is xxxx
Cylinders at xxxx

Burada [xxxx] sabit disk üzerinde kullanılabilir silindirlerin toplam sayısını gösterir.

Total Fixed Disk space is xxxx cylinders
(Toplam sabit disk sahası xxxx silindirleridir)
Maximum available space is xxxx
(Maksimum kullanılabilir saha xxxx dir)
Cylinders at xxxx
(Silindirler xxxx dadır)

Bu mesaj verildikten sonra ,

Enter partition size . . . . . : [xxxx]

Yukarıda [xxxx] içerisinde verilen sayı disk üzerinde geçerli olan bölüm büyüklüğüdür. Eğer DOS'un en geniş alanı kullanmasını isteniliyor ise RETURN tuşuna basmalıyız. Eğer belli bir kısım kullanılmak isteniyor ise kullanılacak büyüklüğü silindir tanımında yazar ve RETURN tuşuna basabiliriz. Daha sonra ekranda aşağıda cevap bekleyen mesaj görüntülenir.

Enter starting cylinder number...: [xxxx]

Yukarıda [xxxx] ile verilen sayı başlangıç silindir sayısını belirtir. Bu sayı Enter partition size ile verilmiş bulunan bölüm büyüklüğüne bağlıdır. Sabit diskteki en geniş alanın ilk silindiri bölüm için yeterli genişliktedir. Eğer DOS'un bu bölümde yer almasını istiyor ise RETURN tuşuna basılmalıdır. DOS'un ilk silindir yerine tercih edilen silindirde yer alınması isteniyor ise silindir numarası yazılıp RETURN tuşuna basılmalıdır.

Bütün bu işler yapıldıktan sonra DOS bölümü FORMAT'lanmaya hazır duruma getirilmiştir. Daha önce anlatıldığı gibi DOS bölümü FORMAT'lanır.

2. Change Active Partition : DOS'u bölümlere ayırmak istiyor isek ana menüden 2 seçimi yapılmalıdır. Bölme durumu aktif durum için A (Active) aktif olmayan durum için N (Non-active) ile gösterilmiştir. Bu bölüm sistemi çalıştıran güç anahtarı açıldığında veya sistem yeniden açıldığında (reset edildiğinde) DOS'un kontrol ettiği bölmedir.

3. Delete DOS Partition : Bu seçenek DOS bölümünü silmek için kullanılır. Ana menüen 3 yazıp RETURN'a basıldığı takdirde aşağıdaki menü ekrana gelir.

1. Delete Primary DOS partition
2. Delete Extend DOS partition

Enter choice [1]

Eğer 1 no'lu seçenek seçilir ise DOS bölümünün tamamı silinir. Eğer 2 no'lu seçenek seçilir ise ilave edilmiş olan DOS bölümü silinir.

I.11.16. DEL KOMUTU

Herhangi bir kütüğün veya kütüklerin silinebilmesi için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

[d1:] DEL [d2:]‹fileadı.fileuzantısı›

[d1:] varsayılan sürücü adı ,

[d2:] silinmek istenenen kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

‹fileadı.fileuzantısı› silinmek istenen kütük adı

DEL komutu icra ettirildiği zaman ? ve * şeklindeki global kütük karakterleri kullanılabilir.

Eğer kütük adı yerine *.* yazılacak olur ise ekranda beliren ;

Are you sure (Y/N)?

sorusuna Y (Evet) cevabı verilecek olunur ise silinmek istenen disk veya disket içerisinde bulunan tüm kütükler silinir.

Yukarıda belirtilmiş *.* yerine sadece . işareti de kullanılabilir.

Örnek I.11.16.a.

A›DEL B:*.BAS

komutu icra ettirilir ise B sürücüsünde bulunan disket içerisindeki uzantısı .BAS olan tüm kütükler silinir.

Örnek I.11.17.b.

A›DEL *.*

veya

A›DEL .

komutu icra ettirilir ise ekranda beliren ;

Are you sure (Y/N)?

sorusuna Y (Evet) cevabı verilecek olunur A sürücüsünde bulunan disket içerisindeki tüm kütükler silinir.

I.11.17. ERASE KOMUTU

Tanımlanmış olan kütük veya kütükleri silmek için kullanılan bir iç komuttur. DEL komutu ile aynı işi yapar.

Genel Yazlım Farkı :

[d1:]ERASE [d2:]‹fileadı.fileuzantısı›

[d1:] varsayılan sürücü adı ,

[d2:] silinmek istenen kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

‹fileadı.fileuzantısı› silinmek istenen kütük yada kütük adları ,

Örnek I.11.17.a.

A›ERASE ESRA.*

A sürücüsünde bulunan ve adı ESRA olan tüm kütükler silinir.

I.11.18. COPY KOMUTU

Bir veya daha fazla kütüğü belirlenmiş disk veya diskete kopyalama işini yapan bir iç komuttur.

Genel Yazılım Formatı :

1.Yazılım şekli

[d1:]COPY [d2:]‹fileadı.fileuzantısı›[/A][/B][d3:]
‹fileadı.fileuzantısı› [/A][/B][/V]

Bu yazılım şekline göre bir kütük bir disketten diske , diskten diskete , diskten diske veya disketten diskete kopyalanabilir.

2.Yazlım şekli

[d1:]COPY [d2:]‹fileadı1.fileuzantısı›[/A][/B][d3:]+
‹fileadı2.fileuzantısı›[/A][/B][d3:]+...
‹fileadı3.fileuzantısı› [/A][/B][/V]

Bu yazılım şekline göre birkaç kütük birleştirilerek diskten diske , disketten diske , diskten diskete veya disketten diskete kopyalanabilir.

/A parametresi kopyesi alınacak kütük (Kaynak disket) için kullanılır ise , kütüğe bir ASCII kütüğü olarak davranılmasını sağlar. Bu kütüğün kopyalanma işlemi

sırasında kütüğün dosya sonu işaretine (Ctrl-Z) kadar olan kısmı kopyalanır. Dosya sonu işareti (Ctrl-Z) kopyalanmaz.
/A parametresi içerisine kopya yapılacak disket (Hedef disket) için kullanılır ise , kopya edilen kütüğün sonuna dosya sonu işaretini (Ctrl-Z) yazar.
/B parametresi kopyesi alınacak kütük (Kaynak disket) için kullanılır ise , kütüğün tamamı kopyalanır. Kopyalama işlemi sırasında kütüğün dosya sonu işareti (Ctrl-Z) de kopyalanır.
/B parametresi içerisine kopya yapılacak disket (Hedef disket) için kullanılır ise , kopya edilen kütüğün sonuna dosya sonu işaretini (Ctrl-Z) yazılmaz.
/V parametresi kopya edilen kütüğün kopya edilecek olan diskete düzgün bir şekilde kopyalanıp kopyalanmadığını kontrol etmek için kullanılır. Eğer VERIFY komutu açık (ON) olarak çalıştırılmış ise /V parametresinin kullanılmasına gerek yoktur.
Kopyalama işi disk veya disket içerisinde bulunan alt directory 'ler içerisinede yapılabilir. Bir diskten diske , disketten diskete , diskten diskete veya disketten diskete kopyalama işlemi üç ayrı şekilde yapılabilir. Kütüklerin kopyalanma işlemleri esnasında global kütük karakterleri olan ? ve * işaretleri kullanılabilir.

1. Aynı isim ile kopyalama ,
2. Ayrı isim ile kopyalama ,
3. Kütükleri birleştirerek kopyalama ,

I.11.18.1. Aynı İsim Altında Kopyalama

Kopya edilecek kütük , kopya edilecek disk veya diskete aynı isim ile kopyalanır.
Genel Yazılım Formatı :

[d1:] COPY [d2:]‹fileadı.fileuzantısı› [d3:]

[d1:] varsayılan sürücü harfi adı ,
[d2:] kopyası alınacak kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,
[d3:] kopye edilecek disketin (Hedef disket) bulunduğu sürücünün harf adı ,

Örnek I.11.18.a.

A sürücüsünde bulunan disket içerisindeki GS.COB programını B sürücüsünde bulunan diskete kopye etmek için ;

A›COPY GS.COB B:

veya

B›COPY A:GS.COB

komutlarından birisi icra ettirilmelidir.

Örnek I.11.18.b.

A sürücüsünde bulunan ve uzantısı .ESR olan tüm programları C diskine kopye etmek için ;

A›COPY *.ESR C:

veya

C›COPY A:*.ESR

komutlarından birisi icra ettirilmelidir.

Örnek I.11.18.c.

A sürücüsünde bulunan disket içerisinde ki GS.COB adlı programı disket içerisindeki ESRA adlı alt directory 'ye kopyalamak için ;

A›COPY GS.COB A:\ESRA

komutu icra ettirilmelidir.

I.11.18.2. Ayrı İsim Altında Kopyalama

Kopyalanmak istenen kütük kopyalanacak disk veya diskete başka isim altında kopyalanabilir.

Genel Yazılım Formatı :

[d1:]COPY [d2:]‹fileadı.fileuzantısı›
[d3:]‹yenifileadı.yenifileuzantısı›

[d1:] Varsayılan sürücü adı harfi ,

[d2:] kopyalanacak kütüğün bulunduğu disk veya disket sürücüsü adı ,

[d3:] kopye edilecek disketin (Hedef disket) bulunduğu sürücünün harf adı ,

Bu yazılıma göre ‹fileadı.fileuzantısı› ile belirtilen kütük adı , [d3:] ile belirtilen disket sürücüsünde bulunan diskete veya diske ‹yenifileadı.yenifileuzantısı› adı ile kopye edilir.

Örnek I.11.18.d.

A sürücüsünde bulunan disket içerisindeki GS.COB adlı

programı yine A sürücüsünde bulunan diskete GUL.COB adı ile kopye edilmek istenir ise ;

A›COPY GS.COB GUL.COB

komutu icra ettirilmelidir. Bu komut icra ettirildikten sonra diskette hem GS.COB adlı program hemde GUL.COB adlı program bulunur.

Örnek I.11.18.e.

A sürücüsünde bulunan disket içerisindeki GS.COB adlı programı B sürücüsünde bulunan disket içerisindeki ESRA adlı alt directory 'ye GUL.COB adı ile kopya edilmek istenir ise ;

A›COPY GS.COB B:\ESRA GUL.COB

veya

B›COPY A:GS.COB B:\ESRA GUL.COB

komutu icra ettirilmelidir.

Örnek I.11.18.f.

A sürücüsünde bulunan bütün kütükleri C diskine kopya edebilmek için ;

A›COPY *.* C:

veya

C›COPY A:*.*

komutlarından biri icra ettirilmelidir.

I.11.18.3. Kütükleri Birleştirerek Kopyalama

Bir veya birkaç kütük birleştirilerek de kopyalama işlemi yapılabilir. Kütükler birbirleri ile + işareti ile birleştirilirler.

Genel Yazılım Formatı :

```
[d1:]COPY [d2:]‹fileadı1.fileuzantısı›[/A][/B][d3:]+
      [d3:]‹fileadı2.fileuzantısı›[/A][/B][d3:]+...
      [d4:]‹fileadı3.fileuzantısı› [/A][/B][/V]
```

[d1:] varsayılan sürücü adı harfi ,

[d2:] birleştirilecek olan birinci kütüğün bulunduğu sürücünün harf adı ,

[d3:] birleştirilecek ikinci kütüğün bulunduğu sürücünün harf adı ,

[d4:] birleştirilen kütüklerin kopyalanacağı disk veya disket sürücüsünün harf adı ,

[d2:] ve [d3:] ile belirtilen disk veya disket sürücülerinde bulunan disk veya disket içerisindeki , belirtilen kütükler birleştirilerek [d4:] ile belirtilen disk veya diskete ‹fileadı3.fileuzantısı› adı altında kopyalama işlemi gerçekleşir.

Örnek I.11.18.g.

A sürücüsünde bulunan disket içerisindeki ESRA1.ESR , ESRA2.ESR , ESRA3.ESR adlı kütükleri birleştirerek yine A sürücüsünde bulunan diskete ESRA.ESR adı altında kopyalama yapılmak istenir ise ;

```
A›COPY ESRA1.ESR+ESRA2.ESR+ESRA3.ESR ESRA.ESR
```

veya

```
B›COPY A:ESRA1.ESR+A:ESRA2.ESR+A:ESRA3.ESR A:ESRA.ESR
```

komutlarından biri icra ettirilmelidir.

Örnek I.11.18.h.

A sürücüsünde bulunan disket içerisindeki ESRA1.ESR ile B sürücüsünde bulunan disket içerisindeki ESRA2.ESR adlı kütükleri birleştirerek A sürücüsünde bulunan disket içerisindeki GUL adlı directory 'e ESRA.ESR adı altında kopya edilmek istenir ise ;

```
A›COPY ESRA1.ESR+B:ESRA2.ESR A:\GUL ESRA.ESR
```

komutu icra ettirilmelidir.

I.11.19. RENAME KOMUTU

Disk veya disket içerisinde bulunan kütüğün isminin değiştirilmesi işlemini yapan bir iç komuttur. İsimi değiştirilmek istenen kütük için global kütük karakterleri ? ve * işaretleri kullanılabilir.

Genel Yazılım Formatı :

```
[d1:] RENAME [d2:]‹fileadı› ‹yenifileadı›
```

veya

```
[d1:] REN [d2:]‹fileadı› ‹yenifileadı›
```

[d1:] varsayılan sürücü adı harfi ,

[d2:] ismi değiştirilecek olan kütüğün bulunduğu disk veya disket sürücüsünün harf adı.

İsim değiştirilme işleminin başarılı bir şekilde yapılabilmesi için , disk veya diskette kütüğe verilecek yeni isim adı altında bir kütük bulunmamalıdır. Eğer adı

değiştirilecek olan kütüğün yeni adı disk veya disketde bulunuyor ise ekranda ;

Duplicate file name or File not found

şeklinde bir uyarı mesajı görülür. Bu mesaj , ismi değiştirilmek istenen kütük ismi bulunamadığı zaman yada kütüğe verilecek yeni ismin disk veya disket içerisinde daha önceden var olması durumunda görüntülenir.

Örnek I.11.19.a.

A sürücüsünde bulunan disket içerisindeki GUL.BAK adlı programın ismini GUL.COB şeklinde değiştirilmek istenir ise;

A>REN GUL.BAK GUL.COB

veya

B>REN A:GUL.BAK GUL.COB

komutlarından birisi icra ettirilmelidir.

Örnek I.11.19.b.

A sürücüsünde bulunan ve G ile başlayan tüm kütüklerin adını H ile aşlayan kütüklere çevirmek istenilir ise ;

A>REN G*.* H*.*

komutu icra ettirilmelidir.

I.11.20. TYPE KOMUTU

Disk veya disket içerisinde bulunan bir kütüğün içeriğini standart çıkış aygıtında görüntüleyen bir iç komuttur.

Genel Yazılım Formatı :

[d1:]TYPE [d2:]<filsadı.filsuzantısı>

[d1:] varsayılan sürücü adı ,

[d2:] ekranda görüntülenmek istenen kütüğün bulunduğu disk veya disket sürücüsünün harf adı.

İçeriği görüntülenecek kütük adları için global kütük karakterleri olan ? ve * kullanılmaz.

Örnek I.11.20.a.

A sürücüsünde bulunan disket içerisindeki GS.DAT isimli kütüğün içeriğini ekranda görüntülenmesi için ;

A>TYPE GS.DAT

veya

B>TYPE A:GS.DAT

komutlarından biri icra ettirilmelidir.

Örnek I.11.20.b.

A sürücüsünde bulunan disketin ESRA adlı alt directory'sinde bulunan SEBNEM.COB adlı kütüğün içeriğini ekranda görüntüleyebilmek için ;

A>TYPE A:\ESRA SEBNEM.COB

Komutu icra ettirilmelidir.

I.11.21. BATCH FILE KOMUTU

DOS tarafından otomatik olarak yürütülmesi istenen komutların işlemesini sağlayan bir dış komuttur.

Genel Yazılım Formatı :

[d:] ‹file adı›.BAT

[d:] Prompt'un bulunduğu sürücü adı harfi veya BATCH file açılacak disketin veya diskin bulunduğu sürücünün harf adı ,

‹file adı› : oluşturulmak istenen BATCH file adı. Disk veya diskette oluşturulmak istenen BATCH file'in uzantısı .BAT olmalıdır.

Bir BATCH file çalıştırmak için BATCH file'ın adının yazılması yeterlidir. File adının uzantısı olan .BAT yazılmasına gerek yoktur.

Bir BATCH file oluşturabilmek için EDLIN.COM veya COPY CON DOS deyimlerinden birisi kullanılabilir.

Bir BATCH file içerisinden çıkılmak istenildiği zaman Ctrl-BREAK tuşuna basılır ise ekranda görülen :

Terminate batch job (Y/N)?

sorusuna Y (Evet) cevabı verilmelidir. Eğer bu soruya Y (Evet) cevabı verilecek olunur ise BATCH file içerisinden çıkılarak sistem PROMPT'una dönülür. Bu soruya N (Hayır) cevabı verilir ise BATCH file içerisindeki diğer komutlar işleme konur. Bu komutlar bittikten sonra sistem PROMPT'una dönülür.

Örnek I.11.21.a.

Adı GS olan bir BATCH file oluşturalım.

A>COPY CON GS.BAT

Yazılıp RETURN tuşuna basılır. Daha sonra BATCH file içerisinde yürütülmek istenen komutlar yazılır. Her komuttan sonra RETURN tuşuna basılmalıdır. Bu komutlar

yazıldıktan sonra ;

F6 tuşuna basılır veya Ctrl-Z

yapılır ise ekranda

1 files copied

şeklinde bir mesaj oluşur. Böylece oluşturulmak istenen GS.BAT Batch file disket içerisinde oluşturulmuş olur.

Örnek I.11.21.b.

Oluşturulmuş olan bu GS.BAT batch file'nın çalıştırılması için ;

A>GS

yazıp RETURN tuşuna basılmalıdır. Daha sonra DOS GS batch file içerisinde yazılmış olan komutları yürütür.

Batch File İçerisinde Kullanılabilecek Parametreler.

Bir batch file içerisinde kullanılabilecek parametreler %0,%1,%2,.....,%9 ifadeleridir.

%0 parametresi her zaman batch file adını ifade eder. Diyelimki batch file adı GS olsun.Bu batch file içerisinde, Batch file adı olan GS kulanılmak istenir ise GS yazmak yerine %0 yazmak yeterlidir.

%1,%2,.....,%9 parametreleri ise batch file çalıştırılır iken batch file ile birlikte verilen file adlarını ifade ederler.

Örnek I.11.21.c.

A>GS GUL ESRA IBR

yazlışında %0 parametresi GS 'yi ,

%1 parametresi GUL 'ü ,

%2 parametresi ESRA 'yı ,

%3 parametresi IBR 'yi ifade etmektedir.

Bu parametreler kullanılarak GS.BAT adlı bir batch file oluşturalım.

```
A>COPY CON GS.BAT
B:
%1
A:
%2
%3
```

Şeklinde oluşturulmuş olan GS batch file'nı aşağıdaki

şekilde çalıştıralım.

A>GS GUL ESRA IBR

GS batch file içerisinde bulunan %1 parametresinin yerine GUL , %2 parametresinin yerine ESRA , %3 parametresi yerine IBR yazılmış olsa idi hiç bir şey farketmiyecekti.

Örnek I.11.21.d.

GS batch file içerisinde bulunan komutlar şunlar olsun.

```
COBOL %1;
RUNCOB %1
TYPE %0.BAT
%0
```

Bu batch file ,

A>GS ESRA

şeklinde çalıştırılır ise batch file içerisinde bulunan komutlar icra görmeye başlar. Bu batch file içerisinde bulunan komutlardan ‹COBOL %1;› ile diskette bulunan ESRA adlı COBOL programı derlenir. Derleme işi bittikten sonra ‹RUNCOB %1› komutu ile ESRA adlı cobol programı çalıştırılır. Daha sonra ‹TYPE %0.DAT› komutu ile GS.BAT file içerisinde bulunan komutlar ekrana dökülür. Daha sonra %0 ile GS batch file tekrar çalıştırılır. Bu şekilde oluşturulmuş batch file'dan ancak Ctrl-BREAK tuşu ile çıkabiliriz. Ctrl-BREAK yapıldığı zaman ,

Terminate batch job (Y/N)?

sorusuna Y (Evet) cevabı verilmelidir. Eğer bu soruya N cevabı verilir ise batch file içerisindeki komutlar icra görmeye devam eder.

I.11.21.1 ECHO ALT KOMUTU

Batch file içerisinde bulunan komutların , PROMPT ile birlikte görüntülenmesini sağlayan veya görüntülemeyi engelleyen bir iç komuttur.

Genel Yazılım Formatı :

ECHO [ON/OFF/MESAJ]

Eğer ECHO ON kullanılmış ise Batch file içerisinde yürütülen komutlar ekranda PROMPT ile birlikte görüntülenir.

Eğer ECHO OFF kullanılmış ise Batch file içerisinde

yürütülmekte olan komutların hiçbiri ekranda gözükmez.
Eğer ECHO ile bir mesaj verilmek istenir ise batch file içerisinde , ECHO yazıp bunun yanına ekranda görülmesi istenen mesaj yazılabilir.
Batch file içerisinde ECHO komutu hiç kullanılmamış ise (Default değer) ECHO ON 'dur.
Örnek I.11.21.e.
GS batch file aşağıdaki komutları içersin.

```
ECHO OFF
COBOL %1;
RUNCOB %1
```

Bu batch file ,

```
A>GS ESRA
```

şeklinde çalıştırmak ister isek : GS batch file içerisinde bulunan komutlar icra görürken PROMPT'la beraber ekrana yazılmazlar. ESRA adlı cobol programı derlenir iken ekranda

```
A>COBOL ESRA;
```

yazısı görülmez. Aynı şekilde ESRA adlı cobol programı çalıştırılır iken ,

```
A>RUNCOB ESRA
```

yazısıda ekranda görülmez. Bu durumda ekranda PROMPT ile birlikte sadece ,

```
A>ECHO OFF
```

yazısı görülecektir.
Eğer bu batch file içeren komutlar aşağıdaki şekilde yazılmış olsa idi,

```
ECHO OFF
COBOL %1;
ECHO ON
RUNCOB %1
```

Ekranda ;

```
A>COBOL ESRA;
```

yazısı görülmez. Bunun yerine sadece ,

```
A>ECHO OFF
```

yazısı görülür. Fakat COBOL %1 ifadesinden sonra ECHO ON komutu kullanıldığı için ekranda ,

A>ECHO ON

A>RUNCOB ESRA

yazısı görülecektir.

I.11.21.2. FOR ALT KOMUTU

Batch file içerisinde kullanılan komutların yinelenerek yazılmasını sağlayan bir iç komuttur.

Genel Yazılış Formatı :

FOR %% ‹Değişken› IN (‹File adları›) DO ‹Komut›

%% ile verilen herhangi bir değişken adı , IN ile verilen file adlarını sırası ile alarak DO ile verilen DOS komutlarını icra eder.

Örnek I.11.21.f.

GS batch file içeren komutlar aşağıdaki aşağıdaki şekilde olsun.

FOR %%I IN (GUL.COB ESRA.BAS) DO DIR %%I

Bu batch file çalıştırıldığı takdirde ekranda ,

A>DIR GUL.COB

yazılarak GUL.COB adlı programın directory'si , daha sonra,

A>DIR ESRA.BAS

yazılarak ESRA.BAS adlı programın directory'si görülür.

I.11.21.3. GOTO ALT KOMUTU

GOTO komutu batch file'ın akışını GOTO ile belirtilmiş olan uygun etikete yönlendirir. Batch file kütüğüne etiket iki nokta üst üste (:) ve onu izleyen etiket ismi ile sokulur. Bir iç komuttur.

Genel Yazılım Formatı :

GOTO ‹Label(etiket ismi)›

GOTO ile verilen etiket ismi Batch file içerisinde hangi satıra dallanılacağını belirtmek içindir. Bu etiket ismi batch file içerisinde olmalı ve bu etiket isminden önce (:) olmalıdır. Eğer batch file içerisinde GOTO komutu kullanılmış ve dallanılacak etiket ismi bulunamamış ise , batch file'ın icrası Label not found şeklinde bir mesaj ile durur.

Örnek I.11.21.g.

GS.BAT batch file'ı ;

```
A>COPY CON GS.BAT
:TEKRAR
COBOL %1;
RUNCOB %1
EDLIN %1.COB
GOTO TEKRAR
```

şeklinde oluşturulmuş olsun. Bu batch file A>GS ESRA şeklinde çalıştırılacak olunur ise ; ESRA adlı cobol programı derlenip çalıştırıldıktan sonra EDLIN editörüne girilir. Editörün içerisinde ESRA.COB adlı program ile ilgili bir takım değişiklikler yapıldıktan sonra sisteme çıkıldığı anda GOTO TEKRAR komutu ile batch file içerisinde verilen TEKRAR adlı etiket ismine dallanılır ve program yeniden derlenir , çalıştırılır ve EDLIN editörüne girilir. Böylece batch file içerisinde sonsuz döngüye girilmiş olunur. Bu Batch file'dan çıkabilmek için Ctrl-Break ile programanın kırılması gerekir.

I.11.21.4. IF ALT KOMUTU

Batch file içerisinde yürütülmek istenen komutların bir şarta bağlı olarak icrasını sağlayan bir iç komuttur.

Genel Yazılım Formatı :

IF [NOT]‹Koşul› ‹Komut›

IF ile verilen koşul doğru ise ‹Komut› ile verilmiş olan komut yürütülür. Belirtilen koşul doğru değil ise IF komutunu izleyen satırda bulunan komut icra ettirilir. IF ile verilen koşul parametreleri aşağıdaki parametrelerden herhangi biri olabilir.

1. ERRORLEVEL ‹Sayı› Eğer programın çıkış kodu ERRORLEVEL ile belirtilen sayı veya daha büyüğü ise doğrudur. BACKUP ve RESTORE gibi DOS komutları ile ERRORLEVEL 'e değer verilebilir.

Örnek I.11.21.h.

GS.BAT batch file kütüğü ,

```
A:BACKUP C:\*.ESR B:/S/M
IF ERRORLEVEL 1 ECHO YEDEKLEME İŞİ GERÇEKLEŞMEDİ
```

şeklindeki komutları içersin,

GS batch file kütüğü çalıştırılacak olunur ise ;
Eğer BACKUP yapma işi başarılı bir şekilde gerçekleşti ise ERRORLEVEL'e 0 değeri , eğer BACKUP yapma işi gerçekleşmedi ise ERRORLEVEL'e 1 değeri yüklenecektir.
Ele alınan örnekte BACKUP alma işi gerçekleşmediği takdirde ekranda ECHO komutu ile 'YEDEKLEME İŞİ GERÇEKLEŞMEDİ' şeklinde bir mesaj verilir ve bir sonraki işleme geçilir.
Eğer BACKUP alma işlemi gerçekleşti ise her hangi bir mesaj vermeden bir sonraki komut icra görecektir.
2. IF ‹String1›==‹String2› ‹koşul›
Eğer String1 ile verilen ifade String2 ile verilen ifadeye özdeş ise belirtilen komut icra görür. Aksi halde IF komutundan sonraki komut icra görür. Her iki stringinde özdeş olabilmesi için karşılıklı karakterlerinin ASCII kodları aynı olmalıdır.
Örnek I.11.21.ı.
GS.BAT kütüğü ;

```
IF %1==ESRA COBOL %1;
```

şeklindeki komutları içersin,
Bu GS batch file GS ESRA şeklinde çalıştırılacak olunur ise;
ESRA adlı cobol programı derlenir fakat GS esra veya GS IBR şeklinde çalıştırılmış olsa idi ESRA adlı cobol programı derlenmeden batch file içerisinde bulunan komutlar icra görmeyecekti.
3. IF EXIST [d:]‹fileadı.fileuzantısı› ‹Komut›
[d:] file adının bulunduğu disk veya disket sürücü adı ,
file adı belirtilen sürücüde bulunur ise ‹Komut› ile yapılmak istenen işlem gerçekleştirilir. Aksi takdirde bir alt satırdan itibaren komutlar icra görmeye devam edecektir. Burada global kütük karakterleri olan ? ve * kullanılabilir.
Örnek I.11.21.i.
GS.BAT batch file kütüğü;

```
IF EXIST B:ES.COB GOTO DERLE
ECHO derlenmek istenilen kütük bulunamdı
:DERLE
```

```
ECHO program derleniyor lütfen bekleyiniz....!
COBOL B:ES;
```

şeklindeki komutları içersin ;
Bu batch file çalıştırıldığı takdirde eğer B sürücüsünde bulunan diskette ES.COB programı yok ise "derlenmek istenilen kütük bulunamadı" şeklinde bir mesaj verecekti. Eğer B sürücüsünde bulunan disket içerisinde ES.COB adlı program bulunuyor ise ; Batch file'ın icra akışı DERLE adlı etikete kayarak "program derleniyor lütfen bekleyiniz....!" şeklinde bir mesaj verip B sürücüsünde bulunan disket içerisindeki ES.COB adlı program derlenecekti.

4. NOT koşulu :

Eğer belirtilen koşul yanlış ise yapılmak istenen komutları icra ettirmek için kullanılan bir komuttur.

Örnek I.11.21.j.

GS.BAT batch file kütüğü ,

```
IF NOT EXIST A:%1 COPY B:%1 A:
GWBASIC %1
```

şeklindeki komutları içersin ;

GS.BAT batch file kütüğü ,

```
A>GS ESRA.BAS
```

şeklinde çalıştırılacak olunur ise ;

Eğer A sürücüsünde bulunan disket içerisinde ESRA.BAS isimli program bulunmuyor ise B sürücüsündeki disket içerisinden ESRA.BAS adlı program A sürücüsünde bulunan disket içerisine kopya edilerek GWBASIC %1 komutu ile ESRA.BAS adlı basic programı çalıştırılmaktadır.

I.11.21.5. PAUSE ALT KOMUTU

Sistemin işleyişini bir an için durdurup ekranda ;

```
Strike a key when ready...
```

şeklinde bir mesaj veren iç komuttur. Bu mesajda devam için herhangi bir tuşa basılması gerektiği anlatılmaktadır. Pause komutu ile ekrana istenilen bir mesaj da yazdırılabilir.

Genel Yazılım Formatı :

```
PAUSE [Mesaj]
```

PAUSE komutu mesajları görüntülemek ve komutlar arasında

disketleri değiştirebilme imkanı sağlamak için batch file içerisinde kullanılabilir. PAUSE komutu kullanıldıktan sonra batch file içerisindeki diğer komutların icra ettirilmesi için Ctrl-Break haricindeki herhangi bir tuşa basılmalıdır.

Eğer PAUSE komutu ile bir mesaj verilmek istenilir ise PAUSE [Mesaj] komutu kullanılmalıdır. Bu komut ile uzunluğu 121 byte geçmeyen bir mesaj verilebilir.

Örnek I.11.21.k.

GS.BAT batch file kütüğü ,

```
:DUR
IF  NOT  EXIST A.COB PAUSE  A sürücüsünde bulunan disket
içerisinde derlenmek istenen program bulunamadı...! disketi
değiştirin....!
GOTO DUR
```

şeklindeki komutları içersin ;

GS.BAT batch file kütüğü çalıştırılacak olunur ise ; A sürücüsünde bulunan disket içerisinde A.COB adlı program bulunamadığı takdirde ekranda "A sürücüsünde bulunan disket içerisinde derlenmek istenen program bulunamadı...! disketi değiştirin....!" şeklinde bir mesaj ile

```
Strike a key when ready...
```

mesajı belirir. Disket değiştirilip herhangi bir tuşa basılacak olunur ise GOTO DUR komutu ile programın akışı DUR adlı etiket ismine sapacaktır.

I.11.21.6. REM ALT KOMUTU

Batch file içerisinde dikkat edilecek notları görüntülemek için kullanılan bir iç komuttur.

Genel Yazılım Formatı :

```
REM [Mesaj(Açıklamalar)]
```

Batch file içerisindeki komutlar REM alt komutuna ulaştığında REM ile birlikte verilen mesaj ekranda görüntülenir. Eğer ECHO komutu kapalı (ECHO OFF) ise REM ile belirtilen mesaj ekranda görüntülenmez.

Örnek I.11.21.1.

GS.BAT batch file kütüğü,

```
COBOL %1;
REM Cobol programı derlendi
PAUSE programı çalıştırmak için bir tuşa basınız...!
RUNCOB %1
REM Cobol programı çalıştırıldı
```

şeklindeki komutları içersin. GS batch file

```
A>GS ESRA
```

şeklinde çalıştırıldığında , ESRA adlı cobol programı derlendiğinde ekranda ;

```
REM Cobol programı derlendi
Programı çalıştırmak için bir tuşa basınız...!
Strike a key when ready ...
```

şeklinde bir mesaj oluşur. Herhangi bir tuşa basılır ise ESRA adlı cobol programı RUNCOB %1 komutu ile çalıştırılıp tekrar siteme dönüldüğünde ekranda;

```
REM Cobol programı çalıştırıldı
```

şeklinde bir mesaj oluşur.

I.11.21.7. SHIFT ALT KOMUTU

SHIFT toplu işlem alt komutu , komut satırlarının 10'dan fazla değiştirilebilir parametre kullanmasını sağlar. Eğer bir komut satırında 10'dan fazla parametre kullanılmak isteniyor ise SHIFT komutu kullanılır. SHIFT komutu kullanıldığı takdirde komut satırındaki bütün parametreler, (%0'ın , %1 ile , %1'in %2 ile ...) yer değiştirmesi şeklinde sola doğru kaydırılırlar. Birbiri ardına yazılan SHIFT komutu bütün parametrelerin bir konum sola doğru kaydırılmasını sağlar. SHIFT komutu bir kez kullanıldıktan sonra ,

```
%0'ın değeri %1'e
%1'ın değeri %2'e
%2'ın değeri %3'e
%3'ın değeri %4'e
%4'ın değeri %5'e
%5'ın değeri %6'e
%6'ın değeri %7'e
%7'ın değeri %8'e
%8'ın değeri %9'e
```

%9'un değeri değeri ise komut satırında bulunan 11. parametreye denk gelir. Böylece komut satırında 11. parametre olarak kullanılan parametre %9 olarak batch file içerisinde işlem görmüştür.

I.11.21.7. AUTOEXEC.BAT KÜTÜĞÜ

İçerisinde AUTOEXEC.BAT batch file bulunan disket ile makine her açıldığında , AUTOEXEC.BAT kütüğü içerisinde bulunan komutları işleme sokulur.

Autoexec.bat kütüğü içerisinde DATE ve TIME komutları bulunmadığı sürece , DOS tarih ve zaman için cevap bekleyen iletiyi oluşturmaz.

Örnek I.11.21.m.

Sistem her açılışında , diskette bulunan ESRA adlı basic programını çalıştıracak bir AUTOEXEC.BAT file oluşturulmak istenir ise :

A>COPY CON AUTOEXEC.BAT

yazıp RETURN tuşuna basılarak diskette AUTOEXEC.BAT kütüğü açılır. Daha sonra AUTOEXEC.BAT kütüğü içerisine yazılacak komutlar sıralanır. Bunlar :

CLS
GWBASIC ESRA

olsun. Kütük içerisine yazılan komutlar bittikten sonra F6 tuşuna veya Ctrl-Z tuşuna basılarak ;

1 files copied

iletisi alındıktan sonra disket içerisinde AUTOEXEC.BAT kütüğü açılmış olur. Artık sistem bu disket ile her açılışında AUTOEXEC.BAT kütüğü içerisinde bulunan komutlar icra görür. Ele alınan örnekte CLS komutu ile ekran temizlendikten sonra , GWBASIC ESRA komutu ile diskette bulunan ESRA adlı basic programı çalıştırılır.

Örnek I.11.21.n.

Sistem her açılışında , zaman ve tarih iletilerini soracak ve ISAM komutunu bilgisayara yükleyecek AUTOEXEC.BAT file kütüğünün içerikleri şunlar olmalıdır.

DATE
TIME
ISAM

I.11.22. DISKCOPY KOMUTU

Kaynak sürücüde bulunan disketin içeriğini , hedef sürücüde bulunan diskete kopyalayan dış komuttur. Aynı zamanda hedef sürücüde bulunan disket formatlanır. Kopyalama işlemi sırasında kaynak diskette bulunan herşey , hedef sürücüde bulunan diskete kopyalanır. Eğer kaynak diskette hatalı alanlarda var ise , bu hatalı alanlar hedef sürücüde bulunan disket içerisine kopyalanır.

Genel Yazılım Formatı :

[d1:] DISKCOPY [d2:] [d3:] [/1]

[d1:] DISKCOPY komutunun bulunduğu disk veya disket sürücüsünün harf adı .

[d2:] Kaynak disketin bulunduğu disket sürücüsünün harf adı .

[d3:] Hedef disketin bulunduğu disket sürücüsünün harf adı.

[/1] , disket veya sürücü tipine bakılmaksızın sadece disketin birinci yüzünü kopyalamak için kullanılır.

Kopyalama işlemi bittikten sonra ekranda ,

Copy another (Y/N)?

şeklinde cevap bekleyen bir mesaj oluşur. Bu soruya N (Hayır) cevabı verilir ise , DISKCOPY komutunun işlemine son verilir. Eğer Y (Evet) cevabı verilir ise , kaynak ve hedef disketlerin değiştirilmesini isteyen bir mesaj belirir. Disketler değiştirildiktan sonra RETURN tuşuna basılır ise , yeni sürülmüş olan disketler kopyalanır.

Örnek I.11.22.a.

A sürücüsünde bulunan disketin tümünü B sürücüsünde bulunan diskete kopyalayabilmek için , ilk önce A sürücüsüne DISKCOPY komutunu içeren disket takılarak ;

A>DISKCOPY A: B:

komutu icra ettirilir ise , ekranda ;

Insert SOURCE diskette in drive A:
Insert TARGET diskette in drive B:
Press any key when ready . . .

A sürücüsüne kaynak disket , B sürücüsüne hedef disket takılarak herhangi bir tuşa basılır ise , disketler arasındaki kopyalama işi gerçekleşir.

Örnek I.11.22.b.

C sürücüsünde bulunan diskten , A sürücüsünde bulunan diskete DISKCOPY yapılamaz. Eğer yapılmaya kalkışılır ise ekranda ;

Invalid drive specification
Specified drive does not exist,
or is non-removable

şeklinde , diskten diskete DISKCOPY işleminin gerçekleşemiyeceğini belirten bir mesaj oluşur.

I.11.23. DIRECTORY ÇEŞİTLERİ

İki çeşit Directory tipi vardır. Bunlar ;

* ROOT (ANA) Directory ,
* ALT (SUB) Directory ,

Tek taraflı bir diskette ROOT directory en fazla 64 kütükten oluşur. Çift tarflı disketlerde ROOT directory en fazla 112 kütükten oluşmaktadır. Yüksek kapasiteli disketlerde ROOT directory en fazla 224 kütük içerebilir. Sabit diskin ROOT directory içerisinde olabilecek maksimum kütük sayısı , DOS bölümünün kapasitesine bağlıdır.

Root directory , kütük adları ile birlikte diğer alt directory adlarınıda içerirler. Alt directory adları kütük adları ile aynı formattadır. Alt directory adı en fazla 8 karakterden oluşur. Eğer istenir ise alt directory adı uzantısıda , bir nokta ve üç karakter olarak verilebilir. Bir kütük adı için geçerli olan her karakter , alt directory içinde geçerlidir. Disk veya disketler içerisine;

* alt directory MKDIR (MD) komutu ile açılır.
* alt directory içerisine CHDIR (CD) komutu ile girilir.
* alt directory RMDIR (RD) komutu ile silinir.

Simdi bunları sırası ile inceleyelim :

I.11.23.1. MKDIR KOMUTU

Belirlenmiş bir diskte veya diskette bir alt directory oluşturmak için kullanılan bir DOS komutudur. Bir iç komuttur.

Genel Yazılış formatı :

MKDIR [d:]\<alt directory adı>

veya

MD [d:]\‹alt directory adı›

[d:] Alt directory'nin açılacağı disketin veya diskin bulunduğu sürücünün adı.

‹alt directory adı› Açılmak istenen alt directory adı. Kütük adı için geçerli olan karakterler alt directory adı olarak kullanılabilir.

Örnek I.11.23.1.a.

Eğer A sürücüsünde bulunan diskette ESRA adlı bir alt directory açılmak istenir ise :

A›MD \ESRA

yazılmalıdır.

Örnek I.11.23.1.b.

Eğer A sürücüsünde bulunan diskette ESRA adlı bir alt directory ve ESRA adlı alt directory içinde EMIR adlı alt directory açılmak istenir ise :

A›MD \ESRA\EMIR

yazılmalıdır.

I.11.23.2. CHDIR KOMUTU

Bir disk veya disketteki oluşturulmuş MD veya MKDIR komutu ile açılmış olan alt directory'lere ulaşmak için kullanılan bir iç komutttur.

Genel Yazılım Formatı :

CHDIR [d:]\‹Alt directory adı›

veya

CD [d:]\‹Alt director adı›

[d:] İçerisine girilecek alt directory'nin bulunduğu disk veya disketin bulunduğu sürücünün adı.

‹Alt directory adı› İçerisine girilmek istenen alt directory adı.

Eğer A›CD veya A›CHDIR yazıldığı takdirde A sürücüsünde bulunan disketin hangi directory'de olduğunu ekrana döker.

Örnek I.11.23.2.a.

A sürücüsünde bulunan disketin hangi directory'sin de olduğunu öğrenmek için

A›CD \

veya

A›CHDIR \

yazılmalıdır. Eğer A sürücüsünde bulunan disket ESRA adlı directory'nin GUL adlı alt directory'sinde bulunuyor ise yukarıda verilen komutların birisini yazıp RETURN tuşuna basılır ise ekranda :

A:\ESRA\GUL

yazısı görülür.

Örnek I.11.23.2.b.

Disk veya diskette bulunmayan GS adlı bir alt directory'e girilmek istendiği zaman ekranda :

Invalid directory

(Yanlış directory adı) mesajı görülür.

Örnek I.11.23.2.c.

A sürücüsünde bulunan disket içerisinde bulunan ESRA adlı alt directory'nin GUL adlı alt directory'sine girilmek istenilir ise :

A>CD \ESRA\GUL

yazılmalıdır.

Örnek I.11.23.2.d.

A sürücüsünde bulunuyor iken B sürücüsünde bulunan disketin hangi alt directory'de bulunduğunu öğrenmek istiyor isek :

A>CD B:

yazılmalıdır.

I.11.23.3. RMDIR KOMUTU

Bir disk veya disket içerisinde bulunan bir alt directory silinmek istenildiğinde kullanılan bir iç komuttur.

Silinmek istenen bir alt directory'nin silinebilmesi için silinecek alt directory içerisinde bulunan tüm kütüklerin silinmiş olması gerekir.

Genel Yazılış Formatı :

[d:] RMDIR <Alt directory adı>

veya

[d:] RD <Alt directory adı>

[d:] Silinecek olan alt directory'nin bulunduğu disket veya disk sürücüsünün adı.

<Alt directory adı> Silinecek olan alt directory adı.

Eğer silinmek istenen alt directory'nin silinmesi esnasında ,

Invalid path , not directory

or directory not empty

şeklinde bir mesaj ile karşılaşılır ise directory'nin silinme işi gerçekleşmemiştir. Bunun nedenleri alt directory adı yazılır iken yazım kurallarına dikkat edilmemiş olabilir , olmayan bir alt directory adı yazılmış olabilir veya alt directory'nin içerisinde bulunan programlar silinmemiş olabilir.

Örnek I.11.23.3.a.

A sürücüsünde bulunan disket içerisindeki ESRA adlı alt directory silinmek istenildiği zaman aşağıdaki işlemler sırası ile yapılmalıdır.

A:\>CD ESRA

yazılıp alt directory'e girilir.

A:\ESRA>ERASE *.*

yazılıp RETURN 'a basıldığı takdirde ,

Are you sure (Y/N)?

sorusuna Y (evet) cevabı verilerek ESRA adlı alt directory'de bulunan tüm kütükler silinir. Daha sonra

A:\ESRA>CD .. veya A:\ESRA>CD \

yazılarak ESRA adlı alt directory'den çıkılır.

A:\>RD ESRA veya A:\>RMDIR ESRA

yazılarak RETURN tuşuna basılır ise A:\> sürücüsünde bulunan disket içerisindeki ESRA adlı alt directory silinmiş olur.

I.11.24. CHKDSK KOMUTU

Belirtilmiş sürücüde bulunan disk veya disket içerisindeki dizinleri ve File Allocation Table'ı (Kütük toplama tablosu) analiz eden dış komuttur.

Genel Yazılım Formatı :

[d1:]CHKDSK [d2:]‹fileadı.fileuzantısı›[/F][/V]

[d1:] CHKDSK komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , Analiz edilecek disk veya disket sürücüsünün harf adı ,

Eğer CHKDSK komutu ile kütük adı belirtildi ise , kütük yada kütükler tarafından işgal edilmiş bitişik olmayan alanlar rapor edilecektir.

/F ,parametresi belirtilmiş ise , CHKDSK komutu dizin veya

kütük yerleştirme tablosundaki hataları tesbit eder. Eğer /F tanımlanmamış ise , CHKDSK komutu ile , disk veya disket üzerinde yapılan düzeltmeler kaydedilmez.

/V , tüm kütükleri varsayılan veya tanımlanmış sürücüde bulunan disk yada disketin içerisindeki yolları görüntüler.

I.11.24.1. CHKDSK Komutu İle İlgili Notlar

* CHKDSK komutu icrası sırasında ekranda görülen tüm Y (Evet) , N (Hayır) şeklindeki cevap isteyen mesajları ; disk veya disket içerisinde meydana gelebilecek değişiklikleri önlemek amacı ile , Y veya N yazdıktan sonra RETURN tuşuna basılmasını bekler ,

* CHKDSK komutu disk veya disket içerisinde kaybedilmiş toplama birimleri (kümeleri) bulur ise , kaybolmuş verinin kurtarılmak istenip istenmediğini sorar. Eğer bu soruya Y (Evet) cevabı verildi ve /F parametresi kullanıldı ise , CHKDSK komutu ile kaybolmuş olan toplama kümesini ismi FILEnnnn.CHK olan kütük olarak kurtarır. Burada nnnn , 0000 ile başlayan sırasal bir sayıdır.

* CHKDSK komutunun icrasından sonra , FILEnnn.CHK şeklindeki kütüklere bakılarak , işe yarayanlar COPY komutu ile birleştirilebilirler. İşe yaramayanlar ise disk veya disket içerisinden silinebilirler.

Örnek : I.11.24.a.

B sürücüsünde adı EMİROĞLU olan disket içerisinde hatalı alan miktarlarını , hidden file sayısını , kullanılacak olan alan miktarını , öğrenebilmek için ;

A sürücüsüne CHKDSK komutunu içeren bir disket takılarak

A>CHKDSK B:

komutu icra ettirilmelidir. Bu komut icra ettirilir ise B: sürücüsünde bulunan disketin hatalalı alan miktarı , hidden file sayısı , disketin toplam kapasitesi , kullanılan file sayısı , boş olan byte miktarı ekranda aşağıdaki şekilde görülür.

```
Volume EMIROGLU  created  Jun 1, 1989 10:47a

      362496 bytes total disk space
           0 bytes in 1 hidden files
      214017 bytes in 6 user files
        1024 bytes in bad sectors
      147455 bytes avaliable on disk

      655360 bytes total memory
      178080 bytes free
```

I.11.25. COMP KOMUTU

Belirli bir disk veya disket içerisindeki kütükler ile, başka bir disk veya disket içerisindeki kütükler ile karşılaştırma işlemini yapan dış komuttur.

Genel Yazılım Formatı :

[d1:]COMP [d2:]‹fileadı.fileuzantısı›[d3:]‹fileadı.fileuzantısı

[d1:] , COMP komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , Karşılaştırılacak olan birinci kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

[d3:] , Karşılaştırılacak olan ikinci kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

I.11.25.1. COMP Komutu İle İlgili Notlar

* COMP komutu genellikle COPY işleminden sonra , kopya edilen kütüklerin özdeş olup olmadığını kontrol etmek için kullanılır.

* COMP komutu ile karşılaştırılan kütükler aynı sürücü içerisindeki disk veya disketta olabilir. Aynı zamanda ayrı alt directory içerisinde de olabilirler.

* COMP komutu ile karşılaştırma işlemi yapılır iken , global kütük karakterleri olan ? ve * karakterleri kullanılabilir.

* Karşılaştırma işlemi sırasında , iki kütük arasında birbirine uymayan bilgiler için aşağıdaki hata iletisi görüntülenir.

Compare error at OFFSET xxxxxx
File1=xx
File2=xx

Burada belirtilen file1 adı ile birinci kütük adı , file2 ile ikinci kütük adı , xx ile byte miktarı belirtilmektedir.

* Birbirine eşit olmayan 10 karşılaştırma işleminden sonra, COMP komutu daha fazla karşılaştırmanın gereksiz olduğu sonucuna varır , işlemi durdurur ve aşağıdaki mesaj verilir.

10 Mismatches-ending compare

* Eksiksiz bir karşılaştırma işleminden sonra ,

Files compare ok

mesajı verilir.

* Karşılaştırma işlemi tamamlandıktan sonra ekranda oluşan,

Compare more files (Y/N)?

şeklindeki soruya Y (Evet) cevabı verilir ise , karşılaştırılacak olan yeni kütük adların girilmesini bekleyecektir. Bu soruya N (Hayır) cevabı verilecek olunur ise karşılaştırma işlemi sona erer.

* COMP komutu bütün karşılaştırma işlemlerinde , karşılaştırılan kütüklerin en son byte'ne bakarak bunların dosya sonu işareti (Ctrl-Z) içerip içermediğini kontrol eder. Eğer kütükler dosya sonu işareti içermez ise ekranda,

EOF mark not found

şeklinde bir mesaj oluşur.

* Karşılaştırılacak olan kütüklerin büyüklükleri farklı ise karşılaştırma işlemi yapılmaz ve ekranda ,

Files are different size

şeklinde bir mesaj oluşur.

Örnek I.11.25.a.

C sürücüsünde bulunan disk içerisindeki adı DOS3.GS olan kütük ile , B sürücüsünde bulunan disket içerisindeki adı DOS3.GS olan kütüğü karşılaştırmak için ;

A sürücüsüne COMP komutunu içeren disket takılarak ,

A>COMP C:DOS3.GS B:DOS3:GS

komutu icra ettirilir ise ekranda ;

Files compare ok

şeklinde bir mesaj oluşur ise C diskindeki DOS3.GS kütüğü ile B sürücüsünde ki DOS3.GS kütükleri aynı içeriklidir. Eğer ekranda ;

Files are different size

şeklinde bir mesaj oluşmuş ise C diskindeki DOS3.GS kütüğü ile B sürücüsünde ki DOS3.GS kütükleri aynı içerikli değildir.

I.11.26. DISKCOMP KOMUTU

Hehangi iki disketin içeriğini karşılaştırmak için kullanılan dış komuttur. DISKCOMP komutu genellikle DISKCOPY işleminden sonra , iki disketin aynı olup olmadığını kontrol etmek için kullanılır. DISKCOMP komutu ile sadece disketler kontrol edilebilir. Eğer sabit sürücü harfi belirtilmiş ise hata mesajı verir.

Genel Yazılım Formatı :

[d1:] DISKCOMP [d2:] [d3:][/1][/8]

[d1:] , DISKCOMP komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , Karşılaştırılacak olan birinci disketin takılı olduğu disket sürücüsünün harf adı ,

[d3:] , Karşılaştırılacak olan ikinci disketin takılı olduğu disket sürücüsünün harf adı ,

[/1] , parametresi ile , sadece disketlerin ilk yüzünü karşılaştırmak için kullanılır.

[/8] , parametresi ile , ilk disket her iz'de (track) 9/15 kesim (sektör) içeriyor olsa bile , her iz'de sadece 8 kesimi (sektörü) karşılaştırmak için kullanılır.

I.11.26.1. DISKCOMP Komutu İle İlgili Notlar

* DISKCOMP komutu ile aynı veya değişik sürücüler belirlenebilir. Eğer aynı sürücüler belirlenmiş ise , tek sürücülü (single-drive) işlemi yapılır. Buna göre ilk önce disketlerin biri belirlenen sürücüye takılır. Birinci disketin okuma işlemi bittikten sonra , ikinci disket sürücüye takılarak karşılaştırma işlemi yapılır.

* DISKCOMP komutu her iki disketin karşılıklı kanallarını karşılaştırır. Karşılaştırılan kanallar eşit değil ise bir

hata mesajı verir.

* Disketlerin karşılaştırılma işlemi bittikten sonra ekranda ,

Compare more diskettes(Y/N)?

şeklinde cevap bekleyen bir mesaj belirir. Eğer bu soruya Y (Evet) cevabı verilir ise karşılaştırılacak olan disketler değiştirilerek , karşılaştırma işlemine devam edilir. N (Hayır) cevabı verilir ise DISKCOMP işlemi sona erer.

I.11.27. RECOVER KOMUTU

Disk veya disket üzerinde hatalı olan kütükleri yeniden kullanılır hale getirmek için kullanılan dış komuttur. Hatalı bir bölüm içeren kütüğün , hatalı bölümündeki bilgilerin çıkarılması ile bu durumdan kurtulabileceği gibi disk veya disket üzerindeki kütüklerin tümü de kurtarılabilir.

Genal Yazılım Formatı :

[d1:] RECOVER [d2:]‹fileadı.fileuzantısı›

veya

[d1:] RECOVER [d2:]

[d1:] , RECOVER komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , Kurtarılacak disk , disket veya kütüğün bulunduğu sürücünün harf adı ,

RECOVER komutu ile global kütük karakterleri olan ? ve * karakterleri kullanılabilir. Fakat ? ve * karakterleri kullanılmış ise , belirtilen ‹fileadı.fileuzantısı›'na uyan ilk kütük kurtarılır. Bunun nedeni ise RECOVER komutu , eğer file adı belirtilmiş ise sadece bir kütüğü kurtarır.

Örnek I.11.27.a.

B sürücüsünde bulunan disket içerisindeki GS.COB adlı kütüğü kurtarabilmek için , A sürücüsüne RECOVER komutunu içeren disket takılarak ,

A›RECOVER B:GS.COB

komutu icra ettirilmelidir.

Örnek I.11.27.b.

B sürücüsünde bulunan disket içerisindeki silinmiş olan tüm kütükleri kurtarabilmek için , A sürücüsüne RECOVER komutunu

içeren disket takılarak ,

A>RECOVER B:

komutu icra ettirilmelidir.

I.11.28. ASSIGN KOMUTU

DOS'a disk işlemleri için belirtilen sürücüden başka bir sürücü kullanmasını söyler. Dış komuttur.

Genel Yazılış Formatı:

[d:] ASSIGN x=y

d: ASSIGN kütüğünün bulunduğu disk veya disket sürücüsünün adı,

x,y: herhangi bir sürücü harfi adı (A>,B>,C> gibi),

olmak üzere bilgisayara ASSIGN x=y komutu icra edildikten sonra hesaplayıcı , x sürücüsünde yapılmak istenen tüm okuma yazma veya silme işlemlerini y ile belirtilen sürücüde bulunan disk veya disket üzerinde yapar.

Örnek I.11.28.a.

A>ASSIGN A=B yazıp RETURN tuşuna basacak olur isek A> prompt'unda iken A sürücüsünde bulunan disket üzerinde yapmak istenilen işlemler B sürücüsünden yapılır.

A>DIR denildiği zaman A sürücüsünde bulunan disketin listesi değilde B sürücüsünde bulunan disketin listesi ekrana dökülür. Başka bir deyişle hesaplayıcı , A sürücüsünde bulunan disketten bir program çalıştırılmak istenir ise çalıştırılmak istenen programı B sürücüsünde bulunan disket üzerinde arar. ASSIGN A=B komutunu normal hale getirebilmek için B sürücüsüne ASSIGN komutunu içeren bir disket takılır ve B>ASSIGN yazılıp RETURN tuşuna basılmalıdır.

Örnek I.11.28.b.

A>ASSIGN A=C B=C komutu icra ettirilir ise A ve B sürücüleri üzerinde yapılmak istenen tüm işlemler C sürücüsünde bulunan disk üzerinde yapılır. Bu durumda artık bilgisayarın C sürücüsünden başka işlem yapan sürücüsü kalmamıştır. Bunu normal hale getirebilmek için C sürücüsünde ASSIGN komutu olmalıdır. ASSIGN komutu yok ise sistem yeniden açılmalıdır.

I.11.29. ATTRIB KOMUTU

Bir kütüğün okunur niteliğini yanlızca okumak için ayarlamaya veya kütüğün özniteliğini görüntülemez yapmaya yarar. Dış komuttur.

Genel Yazılış Formatı :

[d:]ATTRIB [+/- R kütük adı.kütük adı uzantısı]

d: ATTRIB komutunun bulunduğu sürücü adı,

+R : tanımlanmış kütük adı ve kütük adı uzantısı ile verilmiş program ismine yanlızca okuma niteliği verir.

-R : +R yapılmış kütüğün tekrar eski haline getirilmesini sağlar. (tanımlanmış kütüğün yanlızca okuma niteliğini kaldırmak için kullanılır.)

Örnek I.11.29.a.

A>ATTRIB +R B:ESRA.BAS

yazılıp RETURN tuşuna basılır ise B sürücüsünde bulunan ESRA.BAS adlı programa yanlızca okuma niteliği verir. Bu program disketten silinmeye kalkışılır ise Access Denied şeklinde bir mesaj verir. Başka bir deyişle bu program disketten silinemez. Bu programın yanlızca okuma niteliği kaldırılmak istenir ise A>ATTRIB -R B:ESRA.BAS yazılıp RETURN tuşuna basılmalıdır.

I.11.30. BACKUP KOMUTU

Bir veya daha fazla sayıda kütüğü bir disk veya disketten başka bir disk veya diskete yedeklemeye yarayan DOS komutudur. Dış komuttur. Başka bir deyişle disketten sabit diske , sabit diskten sabit diske veya sabit diskten diskete program yedeklemek için kullanılan bir DOS komutudur.

Genel Yazılım Formatı :

[d:]BACKUP [d1:]file adı.file uzantısı [d2:]/S/M/A/D:mm-dd-yy

[d:] BACKUP adlı DOS komutunun bulunduğu sürücünün adı.

[d1:] Yedeklenmek istenen kütüklerin bulunduğu sürücü adı.

file adı.file uzantısı , Yedeklenmek istenen kütük adı veya kütük adları.

[d2:] Yedeklenecek kütüklerin kütüklerin hangi sürücüdeki diske veya diskete yükleneceğini belirten sürücü adı.

/S , Yedeklenmek istenen kütüklere ek olarak , alt directory kütüklerinide yedeklemek için kullanılır.

/M , En son yedeklemeden bu yana değiştirilmiş olan kütükleri yedeklemek içindir.

/A , Yedeklenecek disket içerisinde daha önce bulunan kütüklerin silinmesini istemiyor isek bu parametre kullanılır.

/D , Değiştirilmiş bulunan veya belirtilmiş tarihten sonra değiştirilmiş kütükleri yedeklemek içindir. Belirlenmiş tarihin formatı SELECT , COUNTRY komutlarını kullanarak seçilen ülke koduna göre verilmelidir.

Örnek I.11.30.a.

```
C>A:BACKUP C:\*.ESR B:
```

Yazılıp RETURN tuşuna basılır ise ;

```
Insert backup diskette 01 in drive B:
Warning! Files in the target drive
B:\root directory will be erased
Strike any key when ready
```

Şeklinde bir menü ekranda görülür. Bu menüde B sürücüsünde bulunan programların silinebileceği belirtiliyor. Bunun için dikkatli olunması gerekmekte olduğunu belirtiyor. Herhangi bir tuşa basıldığı takdirde C sürücüsünde bulunan ve uzantısı .ESR olan programları B sürücüsündeki diskete aşağıda görüldüğü gibi yedekler.

```
*** Backing up files to drive B: ***
Diskette number: 01
\XNENIX.ESR
\COBOL.ESR
\BASIC.ESR
\DOS.ESR
\PASCAL.ESR
```

Böylece C sürücüsünde bulunan ve uzantısı .ESR olan programlar B sürücüsündeki diskete yedeklenmiş olur. Bu yedekleme esnasında B sürücüsünde yedeklemeden önce bulunan programların hepsi silinir.

Örnek I.11.30.b.

```
A>BACKUP B: A:/A/D 6-1-1989
```

komutu ile , B sürücüsünde bulunan disket içerisindeki kütüklerden 1.Haziran 1989 tarihinden sonra değişmiş olan

kütükler , A sürücüsünde bulunan diskete yedeklenir. Yedekleme esnasında A sürücüsünde bulunan disket içerisinde bulunan kütükler silinmez.

I.11.31. RESTORE KOMUTU

BACKUP komutu ile yedeklenmiş olan kütükleri yeniden disk veya diskete yüklemek için kullanılan bir dış komuttur.

Genel Yazılım Formatı :

[d1:] RESTORE [d2:] [d3:]‹fileadı.fileuzantısı›[/S][/P]

[d1:] , RESTORE komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , BACKUP komutu ile yedeklenmiş olan kütüklerin bulunduğu disk veya disket sürücüsünün harf adı ,

[d3:] , kütüklerin yeniden yükleneceği disk veya disket sürücüsünün harf adı ,

[/S] , tanımlanmış dizin içerisindeki yedeklenmiş olan kütüklere ek olarak , alt dizinler içerisindeki yedeklenmiş olan kütükleri de yüklemek için kullanılır ,

[/P] , RESTORE komutunun , son yedeklemeden beri değişmiş olan veya "sadece okuyun" işaretli kütüklerin yüklenmesinden önce uyarmasını sağlar. DOS işletim sistemi kütükleri (IBMBIO.COM ve IBMDOS.COM) , format veya SYS komutları ile oluşturulduklarında "sadece okuyun" diye işaretlenirler.

Yükleme esnasında global kütük karakterleri olan ? ve * karakterleri kullanılabilir.

RESTORE komutu ERRORLEVEL adlı hata değişkenine şu değerleri verir.

0 Normal bitiş ,

1 Yeniden yüklenecek hiçbir kütük bulunamamıştır ,

2 Bazı kütükler , kütük bölüşüm sorunları nedeni ile yeniden yüklenememiştir.

3 Kullanıcı tarafından sona erdirilmiştir (Ctrl-Break veya ESC)

4 Hata nedeni ile sona erdirilmiştir.

Yukarıda belirtilen kodlar , IF alt komutu ile , hata düzeyini denetlemek için kullanılabilir.

Örnek I.11.31.a.

B sürücüsünde bulunan disket içerisindeki , BACKUP ile yedeklenmiş olan tüm kütükleri C diskine yeniden yüklemek için , A sürücüsüne RESTORE komutunu içeren disket takılarak ;

A>RESTORE B: C:

komutu icra ettirilmelidir.

I.11.32. EXE2BIN KOMUTU

Uzantısı .EXE olan kütükleri , uzantısı .COM veya .BIN olan kütüklere çeviren dış komuttur.

Genel Yazılım Formatı :

[d1:] EXE2BIN [d2:]‹fileadı.EXE› [d3:]‹fileadı.COM/BIN›

[d1:] , EXE2BIN komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , uzantısı .EXE olan kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

[d3:] , uzantısı .COM veya .BIN olması istenen kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

I.11.33. FIND KOMUTU

Belirlenmiş bir dizginin , tanımlanmış kütük adlarında bulunan yerlerini standart çıkyı aygıtında gösteren dış komuttur.

Genel Yazılım Formatı :

[d1:] FIND [/V][/C][/N] "Dizgi" [d2:]‹fileadı.fileuzantısı›

[d1:] , FIND komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

/V , belirlenen kütük içerisinde aranacak olan dizgiyi içermeyen tüm satırları görüntülemek içindir ,

/C , kütük içerisinde aranacak olan dizgiyi içeren satırların , kütüğün kaçıncı satırında olduğunu görüntüler,

/N , kütük içerisinde aranacak olan dizgiyi göstermeden önce , dizginin bulunduğu satır numarasını gösterir ,

"Dizgi" , kütük içerisinde aranacak olan dizgiyi belirtmek için kullanılır ,

[d2:] , içerisinden herhangi bir dizgi aranacak olan kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

Örnek I.11.33.a.

B sürücüsünde bulunan disket içerisindeki diskette bulunan GS.COB adlı kütükten "MUS" dizgisinin bulunduğu satırları görüntülemek için ; A sürücüsüne FIND komutunu içeren bir disket takılarak ;

A>FIND /C "MUS" B:GS.COB

komutu icra ettirilmelidir.

Örnek I.11.33.b.

Yukarıdaki örnekte , kütükten "MUS" dizgisini içermeyen satırları görmek istenilir ise ,

A>FIND /V "MUS" B:GS.COB

komutu icra ettrilmelidir.

I.11.34. CTTY KOMUTU

Standart giriş çıkış konsolunu bir yardımcı konsol ile değiştiren veya klavye ve ekranı standart çıkış aygıtı durumuna yeniden dönüştüren bir iç komuttur.

Genel Yazılım Formatı :

[d1:]CTTY ‹aygıt ismi›

[d1:] , varsayılan sürücünün harf adı ,

‹aygıt ismi› , CON (klavye konsolü) , PRN veya LPT1 (çıkış aygıtı) olabilir.

I.11.35. GRAFTABL KOMUTU

Hafızaya renkli (Çizgisel uyarlayıcı için) ek karakter veri tablosu yükleyen dış komuttur.

Genel Yazılım Formatı :

[d:] GRAFTABL

[d:] , GRAFTABL komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

Renkli/grafik uyarlayıcıda grafik mode içinde olduğunda yabancı dil karakterlerini görüntülemek için GRAFTABL kullanılır. Bu grafik modunda 128 den 255'e kadar olan ASCII karakterleri kullanılabilir.

I.11.36. GRAPHICS KOMUTU

Renkli/Grafik ekran uyarlayıcısı kullanıldığı zaman , printer üzerinde grafik görüntü ekranının içeriklerinin basılmasına imkan oluşturan dış komuttur.

Genel Yazılım Formatı :

[d:]GRAPHICS [Yazıcı Tipi][/R][/B]

[d:] , GRAPHICS komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[Yazıcı Tipi] , kullanılan yazıcı tipini belirtmek için kullanılır. Kullanılabilecek yazıcı tipleri şunlardır.

COLOR1 , Siyah şeritli renkli yazıcı için kullanılır.

COLOR4-RGM şeritli (kırmızı , mavi , yeşil ve siyah) yazıcı için kullanılır.

COLOR8-EMY şeritli (siyan , macenta , sarı , siyah) yazıcı için kullanılır.

COMPACT-IBM , Compact yazıcı için kullanılır.

GRAPHICS-IBM , grafik yazıcı için kullanılır.

Eğer yazıcı tipi belrtilmeyecek olunur ise varsayılan değer GRAPHICS yazıcıdır.

/R , siyahı siyah , beyazı beyaz gibi yazmak içindir. Eğer /R parametresi belirtilmeyecek olunur ise varsayılan değer, siyahın beyaz , beyazın siyah olarak yazılacağıdır.

/B , zemin saha renginin basılabilmesi içindir. Bu parametre sadece COLOR4 ve COLOR8 tipi yazıcılar için kullanılır. Eğer /B parametresi belirtilmeyecek olunur ise zemin rengi basılmayacaktır.

Bu komut bilgisyara yüklendikten sonra , ekran içeriklerini yazıcıda takılı olan kağıda yazmak için PRTSCR (Print Screen) tuşuna basılmalıdır.

I.11.37. KEYBxx KOMUTU

ROM BIOS'da yerleşik olan klavye programını , yüklenmek istenen klavye programına dönüştürmek için kullanılan dış komuttur. Komuttaki xx , DOS disketinde bulunan 6 kalavye programından birini temsil eder.

Genel Yazılım Formatı :

[d:]KEYBUK

veya

[d:]KEYBGR

veya

[d:]KEYBIT

veya

[d:]KEYBSP

veya

[d:]KEYBTR

[d:] . KEYxx komutunun bulunduğu disk veya disket sürücüsünün harf adı .

KEYBUK komutu ile United Kingdom (ingiliz alfabesi) klavye'ye yüklenir.

KEYBGR komutu ile Germany (alman alfabesi) klavye'ye yüklenir.

KEYBFR komutu ile France (fransız alfabesi) klavye'ye yüklenir.

KEYBIT komutu ile Italy (italyan alfabesi) klavye'ye yüklenir.

KEYBSP komutu ile (Spain) (ispanyol alfabesi) klavye'ye yüklenir.

KEYBTR komutu ile (Turkish) (Türk alfabesi) klavye'ye yüklenir.

Örnek I.11.37.a.

Türkçe klavye'yi yükleyebilmek için . A sürücüsüne KEYBTR.COM komutunu içeren disket takılarak ;

A>KEYBTR

komutu icra ettirilmelidir. Bu komut yüklendikten sonra CTRL-ALT-F1 tuşu ile normal klavye türüne geçilebilir. Tekrar CTRL-ALT-F3 yapılarak türkçe klaveye türüne geçilebilir.

I.11.38. MODE KOMUTU

Yazıcıya , Renkli/Grafik ekran uyarlayıcısına veya eşzamanlı olmayan iletişim uyarlayıcısına olan yolu ayarlar.

Genel Yazılım Formatı :

MODE komutunun iki çeşit yazılım şekli vardır.

1. Yazılım Şekli :

[d:]MODE LPT#[:][n],[m]

[d:] . MODE komutunun bulunduğu disk veya disket sürücüsünün harf adı .

. bu parametre ile yazıcı numarası belirtilir . # yerine yazılabilecek değerler 1 , 2 veya 3'tür.

n , bu parametre ile her satır için karakter sayısı belirlenebilir , n parametresinin alabileceği değerler 80 veya 132 'dir.

m , bu parametre ile her satıra düşecek inç miktarı belirtilir , m parametresinin alabileceği değerler 6 veya 8'dir.

MODE komutunun bu yazılım şekli ile yazıcı m ve n parametrelerine göre yazıcı ayarlanabilir. m ve n parametreleri belirtilmediği takdirde (default değerleri) , n için 80 karakter (yazıcının bir satırının uzunluğu 80 karakter) , m için 6 inçtir.

Örenk I.11.38.a.

Yazıcının bir satıra 132 karakter yazması ve bir satırın genişliğinin 8 inç olması isteniyor ise ; A sürücüsüne MODE komutunu içeren bir disket takılarak ,

A>MODE LPT1:132,8

komutu icra ettirilmelidir.

2. Yazılım Şekli :

[d:]MODE n

veya

[d:]MODE [n],m[,T]

[d:] , MODE komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

n , bu parametre ile ekran tipi belirlenebilir. alabileceği değerler 40 , 80 , BW40 , BW80 , CO40 , CO80 veya MONO'dur.

40 Renkli veya grafik monitör işleyicisi için ekranın her satırını 40 sütün olarak değiştirir.

80 Renkli veya grafik monitör işleyicisi için ekranın her satırını 80 sütün olarak değiştirir.

BW40 İşlek görüntü işleyicisini , renkli veya grafik monitör işleyicisine aktarır , görüntüyü siyah-beyaz yapar ve ekranı her satır için 40 karakter genişliğine ayarlar.

BW80 İşlek görüntü işleyicisini , renkli veya grafik monitör işleyicisine aktarır , görüntüyü siyah-beyaz yapar ve ekranı her satır için 80 karakter genişliğine ayarlar.

CO40 İşlek görüntü işleyicisini , renkli veya grafik monitör işleyicisine aktarır , görüntüyü renkli ekrana dönüştürür ve ekranı her satır için 40 karakter genişliğine ayarlar.

CO80 İşlek görüntü işleyicisini , renkli veya grafik monitör işleyicisine aktarır , görüntüyü renkli ekrana dönüştürür ve ekranı her satır için 40 karakter genişliğine ayarlar.

MONO İşlek görüntü işleyicisini Monochrome görüntü işleyicisine aktarır.
(Monochrome Görüntü işleyicisinin , görüntü genişliği daima her satır için 80 karaktere ayarlanmıştır.)

m , bu parametre ile görüntüyü sağa veya sola kaydırmak için kullanılır. R veya L dir. R ekranı sağa kaydırmak için , L ekrası sola kaydırmak için kullanılır. Okunabilirliliği ayarlamak için görüntü , 40 sütünlu bir ekran için 1 karakter , 80 sütünlu bir ekran için 2 karakter sağa veya sola kaydırılabilir.

T , görüntüyü hizalamak için kullanılacak bir deneme bilgisi kalıbı isteminde bulunur. T parametresi kullanıldı ise , ekranın iyi hizalanıp hizalanmadığı konusunda cevap bekleyen aşağıdaki mesajlardan birisi sunulur.

Dou you see the leftmost 0? (y/n)

veya

Do you see the rightmost 9? (y/n)

Eğer bu soruya Y (Evet) cevabı verilirse komut sona erer. Eğer bu soruya N (Hayır) cevabı verildi ise cevap bekleyen mesajın ardından kayma işlemi , R kullanıldı ise sağa doğru L kullanıldı ise sola doğru kayma işlemi tekrarlanacaktır.

Örnek I.11.38.b.

Ekranın her satırını 80 karaktere ayarlamak ve görüntüyü sağa kaydırmak için ; A sürücüsüne MODE komutunu içeren disket takılarak ,

A>MODE 80,R,T

komutu icra ettirilmelidir. T parametresi kullanıldığı için ekranda oluşan ;

you see the leftmost 0? (y/n)

şeklinde bir soru oluşur. Bu soruya n (hayır) cevabı verilecek olunur ise ekran 2 karakter sağa doğru kayar. Bu soruya y (Evet) cevabı verilirse komutun icrası sona erer.

I.11.39. MORE KOMUTU

Disk veya disket içerisinde bulunan verileri , kütükleri okur , sonra tüm ekranı dolduracak miktarda veriyi standart çıkış aygıtına gönderir ve ardındanda , ekranın sol alt köşesinde --More-- şeklinde bir mesaj ile durur. Bir sonraki ekran dolusu veriyi almak için herhangi bir tuşa basılmalıdır. Dış komuttur.

Genel Yazılım Formatı :

[d1:]MORE [d2:]‹fileadı.fileuzantısı›

[d1:] , MORE komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , içeriği görülecek kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

Örnek I.11.39.a.

C diskinde bulunan GS.COB adlı kütüğün içeriğini , bir ekran dolusu olacak şekilde dökebilmek için ; A sürücüsüne MORE komutunu içeren bir disket takılıp ,

A›MORE C:‹GS.COB

komutu icra ettirilmelidir. MORE komutu kullanılır iken kütükadı belirtilmeden "‹" işaretinin kullanılmasına dikkat edilmelidir.

I.11.40. PRINT KOMUTU

Bilgisayarda başka bir iş yapılır iken , yazıcıda istenilen kütüğün listeleme işlemini yapan dış komuttur.

Genel Yazılım Formatı :

[d1:]PRINT [/D][/B][/U][/M][/S][/Q][/C][/C][/T][/P]
[d2:]‹fileadı.fileuzantısı›

[d1:] , PRINT komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , dökümü alınacak kütüğün bulunduğu disk veya disket sürücüsünün harf adı ,

/D , bu parametre yazıcı aygıtını tanımlamak için kullanılır. Tanımlanmamış ise varsayılan yazıcı aygıtı PRN dir.

/B , bu parametre ara belleğin byte ölçüsünü ayarlamak içindir. Varsayılan değer 512 byte'dir. B'nin artan değeri PRINT komutunun performansını artırabilir.
/Q , bu kuyruk içerisinde sahip olunabilecek kütük sayısını belirtir. Q yerine yazılacak değerler 1 ile 32 sayısı arasındadır. Varsayılan değer 10'dur.
/S , zaman dilimini tanımlamak içindir. Varsayılan değeri 8'dir. S yerine yazılacak değerler 1 ile 255 sayıları arasında olabilir.
/U , yazıcı aygıtının kullanmasına kadar bekleyecek PRINT'in saat tıkırtıları sayısını tanımlamak için kullanılır.
/M , yazıcı aygıtında karakterlerin basılmaları için PRINT komutunun sahip olacağı saat tiktaklarının kaç tane olabileceğini tanımlamak içindir. Varsayılan değer 2 'dir. Değerlerin yazılım genişliği 1 ile 255 sayıları arasındadır.
Buraya kadar verilmiş olan parametreler , /D , /B , /Q , /S /U , /M , PRINT komutunun ilk kullanıldığı zaman tanımlanmış olamalıdırlar. Eğer tekrar tanımlanmış olunurlar ise , ekranda "Invalid Parameters" şeklinde bir mesaj oluşur.
/T , sonlandırma durumunu hazırlar. Sıraya konan tüm kütükler basım listesinden çıkarılır. Eğer herhangi bir kütüğün basımı o anda yapılamakta ise basım durur , bir iptal iletisi verilir , kağıt bir sonraki sayfaya geçer ve yazıcı alarmı çalar.
/C , iptal etme durumunu hazırlar. Çıkartılacak kütük veya kütüklerin seçimini yapma imkanı verir.
/P , yazım durumunu hazırlar. Önceki ve onun ardından gelen kütükadlarının tümü ENTER tuşuna basılana kadar , basım listesine eklenir.
Herbiri uygun parametreler ile , komut satırı üzerinde bir kütük adından fazlası girilebilir. Kütük adları yazılırken global kütük karakterleri ? ve * kullanılabilir.
Yukarıda belirtilen parametrelerin hiçbiri kullanılmadan , RETURN tuşuna basılır ise , komut satırındaki listede yer

alan kütüklerin hepsi basım için sıralanır. (/P parametresinin kullanıldığı kabul edilir.)
Eğer /D parametresi ile aygıt adı belirtilmedi ise , PRINT komutu çalıştırıldığı anda ekranda ;

Name of list device [PRN]:

şeklinde bir mesaj oluşur. Bu mesaj cıkış aygıtı belirtilmesine imkan verir. PRN , LPT1 , LPT2 , LPT3 , COM1 COM2 gibi çıkış aygıtlarından herhangi birini seçilebilir. Kütükler , giriş sıralamalarındaki düzen bozulmadan basım kuyruğuna sokulur.
Örnek I.11.40.a.
C diskinde uzantısı .COS ve .MUS olan kütükleri yazıcıya aktarmak için ; A sürücüsüne PRINT komutunu içeren disket takılarak ,

A>PRINT C:*.COS C:*.MUS

komutu icra ettirilmelidir.
Örnek I.11.40.b.
Yazıcıya yüklü olan ve uzantısı .COS olan kütükleri basım kuyruğundan çıkarmak için ;

A>PRINT C:*.COS/C

komutu icra ettirilmelidir.
Örnek I.11.40.c.
Yazıcıda yüklü olan kütüklerin basım sırasını boşaltmak için ,

A>PRINT /T

komutu icra ettirilmelidir.

I.11.41. SELECT KOMUTU

Klavye düzenini , tarih ve zamanı seçmek için imkan sağlayan dış komuttur. SELECT komutu ile , herhangi bir ülkenin tarih yazılım formatı ve klavye türü seçilebilir.
Genel Yazılım Formatı :

[d:] SELECT xxx yy

[d:] , SELECT komutunun bulunduğu disk veya disket sürücüsünün harf adı ,
xxx , ülke kodunu tanımlamak içindir.
yy , klavye kodunu tanımlamak içindir.

I.11.41.1. Ülke kodları ve Klavye Kodları

Ülke Adı	Ülke Kodu	Klavye Kodu
A.B.D.	001	US
Fransa	033	FR
İspanya	034	SP
İtalya	039	IT
İngiltere	044	UK
Almanya	049	GR

SELECT komutu seçilen ülke ve klavye kodlarını diskete kopyalamak için DISKCOPY komutunu kullanır. Ayrıca disket içerisinde COUNTRY komutunu içeren CONFIG.SYS ve KEYxx komutunu içeren AUTOEXEC.BAT kütüğü oluşturulur.

I.11.42. SORT KOMUTU

Standart çıkış aygıtındaki veriyi okuyup , bu veriyi sıraya koyarak standart çıkış aygıtında gösteren dış komuttur.

Genel Yazılım Formatı :

[d:] SORT [/R][/+n]

[d:] SORT komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

/R , bu parametre ile sıralama işlemi tersine çevrilir. Sıralama işlemi küçükten büyüğe doğru değilde , büyükten küçüğe doğru yapılır.

/+n parametresi sıralamayı n. sütünundan başlatan bir tamsayıdır. Hiçbir parametre belirtilmedi ise sıralama 1. sütündan başlar. Sıralanabilecek en büyük kütük boyu 63K'dır.

I.11.43. TREE KOMUTU

Belirtilen sürücüdeki dizin yollarının tümünü görüntüleyen ve istenildiği takdirde herbir alt dizindeki kütükleri listeleyen dış komuttur.

Genel Yazılım Formatı :

[d1:]TREE [d2:][/F]

[d1:] , TREE komutunun bulunduğu disk veya disket sürücüsünün harf adı ,

[d2:] , görüntülenmek istenen dizin yollarının bulunduğu disk veya disket sürücüsünün harf adı ,

[/F] , alt dizin içerisindeki kütüklerin adlarını da görüntülemek içindir.

Örnek I.11.43.a.

B sürücüsünde bulunan disket içerisindeki alt dizin yollarını görüntülemek için ; A sürücüsüne TREE komutunu içeren disket takılarak ,

A>TREE B:

komutu icra ettirilmelidir.

II. BÖLÜM
XENIX İŞLETİM SİSTEMİ

II.1 XENIX İŞLETİM SİSTEMİNE GİRİŞ

XENIX işletim sistemi UNIX işletim sisteminin mini bilgisayarlara uyarlamasıdır. Bütün multi-user (çok kullanıcılı) ve multi-tasking (iş paylaşımlı) işletim sistemlerinde olduğu gibi oldukça komplike bir yapıya sahiptir. Bu nedenle operatör düzeyindeki kullanıcılar sistem için hayati önem taşıyan dosyalara erişemezler ve sistemi tam anlamı ile kullanma hakları sınırlıdır. Sistemi kullanma hakkı sadece SUPERUSER'a aittir. SUPERUSER aynı zamanda root olarakta adlandırılır.

II.1.1 SUPERUSER 'in (root) GÖREVLERİ

a) Sistemi yapıyı düzenlemek,

b) Sistemi gerektiğinde normal bir şekilde açıp kapamak,

c) Sisteme yeni yazılım donanımları eklemek çıkarmak ve bunların düzenli çalışmalarını sağlamak,

d) Sisteme yeni kullanıcılar eklemek ve çıkarmak,

e) Kullanıcıları ve hazırladıkları veri dosyalarını organize etmek,

f) Sistemin düzenli olarak yedekleme ve yenileme işlemlerini yapmak,

g) Sistemin periyodik olarak bakımını yapmak , hataları bulup onları temizlemek,

h) Sistemdeki kullanıcıları yönetmek , sistemin kullanımındaki hatalarını düzeltmek , çalışma raporlarını düzenlemek,

ı) Boş disk alanlarını kontrol etmek ve periyodik olarak system log , core ve tmp gibi dosyaları temizlemek,

i) Diğer kullanıcıları yetiştirmek , sistemi daha bilinçli kullanmalarını sağlamak.

II.2. NASIL SUPERUSER (root) OLUNUR ?

Superuser sistemi tam anlamı ile kullanma yetkisine sahiptir. Superuser hakkına sahip olan kullanıcı sistemdeki bütün dosyalara erişebilir , üzerinde herhangi bir

değişiklik yapabilir ve silebilir. Bundan dolayı sistemi superuser olarak kullanan kullanıcının , sistemi çok iyi tanıması ve ve çalışırken son derece dikkatli olması gereklidir.

Root kullanıcısına sahip olmanın 3 türlü yolu vardır.

a. Sistem BOOT prosedürünü uygular iken karşımıza aşağıdaki mesaj gelir.

Type CONTROL-d to proceed with normal startup,
(or give root password for system maintenance) :

konumunda iken root 'un password'unu girdikten sonra RETURN tuşuna basılır ise tek kullanıcılı modunda root kullanıcısına girilir. Bu durumda sisteme bağlı diğer kullanıcılar sisteme giremezler.

b. Çok kullanıcılı modunda ekranda login mesajı var iken girmektir.

xenix286!login: root ‹ENTER›
password:---------- ‹ENTER›

c. Çok kullanıcılı modunda sistemde çalışıyor iken su komutunu kullanarak root kullanıcısına girmektir.

$ su ‹ENTER›
password:-------- ‹ENTER›
#

su komutu kullanıcıyı geçici olarak superuser yapar. Eğer root'un passwordu yanlış verildi ise sistem kullanıcıya sorry mesajını vererek doğru password verilmediği sürece superuser olunamıyacağını bildirir. Tekrar eski kullanıcıya geçebilmek için CONTROL-d yapılması yeterlidir.

II.3. SISTEMIN AÇMA VE KAPAMA ADIMLARI

Bu bölümde XENIX sysyteminin uygun bir şekilde nasıl açılıp kapatılacağı ve bu aşamalar esnasında karşılaşabilecek durumlar anlatılacaktır.

II.3.1. SISTEMI AÇMA

Ana makinanın güç anahtarı açıldıktan sonra sistem kendini test eder ve XENIX'in bootstrap programı otomatik olarak yüklenir. Bootstrap programı işletim sisteminin yüklenmesini sağlayan bir ön aşama programıdır. XENIX sisteminin açılması 3 aşamada oluşur.

II.3.1.1. İşletim Sisteminin Yüklenmesi

XENIX bootstrap programı yüklendiği zaman ekrana aşağıdaki mesaj gelir.

```
Boot
:
```

bu durumda RETURN tuşuna basılır ise XENIX işletim sistemi otomatik olarak harddiskten sistemin ana hafızasına yüklenmeye başlar.

Eğer harddiskte ayrıca DOS işletim sistemi var ise aşağıdaki örnekte olduğu gibi boot esnasında DOS işletim sistemini yükleyebilirsiniz.

```
BOOT
: dos ‹ENTER›
```

Eğer ? işareti girilecek olur ise eğer sistemin boot edebileceği çevre listesini alabilirsiniz.

```
Boot
: ? ‹ENTER›
Devices are:
    fd
    hd
Default device :hd(40)
```

II.3.1.2. Dosya Sisteminin Temizlenmesi

Bazen sistem gerekli komutlar verilmeden yada bir elektrik kesilmesi ile uygun olmayan bir şekilde kapanmış olabilir. Bu durumda sistem yeniden açıldığında aşağıdaki mesaj ile karşılaşırsınız.

```
System was not shutdown properly
proceed with cleaning (y or n)?
```

Bu mesaj XENIX işletim sisteminin normal çalışmasına devam edebilmesi için dosya sisteminin temizlenmesi gerektiğini bildirir. Eğer bu soruya y cevabı verilecek olursa system hasar görmüş dosyaları düzeltmeye çalışır, düzeltmek mümkün olmadığı takdirde o dosyaları siler. Yapılan bütün işlemleri ekrana yazar ve dosyaları ayrı ayrı kurtarmak isteyip istemediğimizi sorar. Bu sorulara her zaman y cevabını girilmelidir. Genellikle temizleme işlemi bittikten sonra normal çalışmaya geçilmesine karşın

aşağıdaki gibi bir mesajla da karşılaşılabilir.

** Normal System Shutdown **

** Safe to Power Off **

-or-

** Press Any Key to Reboot **

Bu mesaj ile system yeniden yüklenmek istenmektedir. Her hangi bir tuşa basılıp XENIX işletim sisteminin yeni baştan yüklenmesi sağlanabilir.

II.3.1.3 Sistemin Çalışma Ortamına Geçilmesi

Sistem boot prosedürünün son aşamasında aşağıdaki mesajı vererek sizden çalışma ortamını seçmenizi ister.

Type CONTROL-d to proceed with normal startup,
(or give root password for system maintenance) :

XENIX işletim sisteminde 2 farklı çalışma moduna sahiptir. Bunlar çalışma modu ve sistem bakım modlarıdır. Normal çalışma modu seçilmek isteniyor ise CONTROL-d tuşuna basılmalıdır. Bundan sonra sistem ekrana başlangıç mesajını gönderir , tarihi bildirir ve /etc/rc dosyasındaki komutları icra etmeye başlar. Bu komutların icrası bittikten sonra ana konsol ve terminal ekranlarına "login:" mesajı gelir. Sistem bakım modunu seçmek için eğer daha önce şifre verilmiş ise root password'unu verip RETURN tuşuna basılır. Bu durumda ekranda system maintenance moduna girildiği mesajı görüntülenir fakat /etc/rc dosyası çalıştırılmaz.

II.3.2. SİSTEMİ KAPAMA

XENIX sistemi kapatılmadan önce sistemin kapatılmaya hazırlanması gerekmektedir. Sistemi kapatabilmek için haltsys ve shutdown komutlarından yararlanabiliriz. Bu komutları yanlızca root (superuser) kullanıcısında kullanılabilir.

II.3.2.1 Shutdown Komutunun Kullanımı

Shutdown komutu genellikle sistem çok kullanıcı modunda çalışırken kullanılması gereken bir komuttur. Bu komutu çalıştırıldığı zaman sistemdeki bütün kullanıcılara sistemin kapatılacağı mesajı gönderilir ve kullanıcıların işlerini bitirebilmeleri için belirli bir süre tanınır. Bu

şekilde sistemi kapatabilmek için şu aşamalardan geçilir.

* Komut kullanılmadan önce root (superuser) kullanıcısına geçilir.
* Aşağıdaki komut verilir.

/etc/shutdown

* Sistem şu mesajı verir.

Minutes till shutdown? (0-15):

Burada sorulan soru sistemin kaç dakika sonra kapatılacağıdır. 0 ile 15 dakika arasında bir zaman girilebilir. Bu arada kullanıcı ekranlarına aşağıdaki mesaj gönderilir.

Broadcast Message from root
XENIX will now terminate

Bu mesajı alan kullanıcılar olabildiğince çabuk sistemden çıkmalıdırlar. Aksi bir durum karşısında iş yarım kalır ve ekrana aşağıdaki mesaj gelir.

** Normal System Shutdown **
** Safe to Power Off **
-or-
** Press Any Key to Reboot **

Bu mesaj alındıktan sonra sistemin güç anahtarı kapatılabilir.

II.3.2.2 Haltsys Komutunun Kullanımı

Haltsys komutu sistemi hemen kapatma konumuna getirir. Bu komutu kullanmak için aşağıdaki işlemleri yapmalıyız.

* root (superuser) kullanıcısına girilir.
* Aşağıdaki komut verilerek ,

/etc/haltsys

* Sistem hemen durur ve aşağıdaki mesajı görüntüler.

** Normal System Shutdown **
** Safe to Power Off **
-or-
** Press Any Key to Reboot **

Bu mesaj alındıktan sonra sistemin güç anahtarı kapatılabilir.

II.3.2.3 Reboot Komutunun Kullanılması

Superuser kullanıcısında iken sisteme reboot yazıldığı zaman sistem hemen kapanır.

II.4. SYSTEMIN YUKLENMESI

XENIX işletim sistemi floppy disk veya hard diskten boot edilir.

II.4.1. Boot İşleminin Floppy Diskten Yapılması

Floppy diskten boot işlemi üç adımda gerçekleşir.

a. Bilgisayarın ROM'u yada başka bir deyişle BIOS'u floppy disk sektör 0 'dan boot block bölümünü yükler.

b. Boot block floppy disketin root directory'sinden /boot dosyasını arar ve sisteme yükler.

c. /boot programı çalışmaya başlar ve en sonunda ekrana kullanıcı prompt'u gelir.

II.4.2. Boot İşlemi Harddiskten Yapılması

Harddiskten boot işlemi 5 aşamada gerçekleştirilir.

a. Bilgisayarın ROM'u hard disk sektör 0'dan masterboot block bölümünü yükler.

b. Masterboot block yine aynı sektörden hangi partition aktif ise onun boot bloğunu yükler.

c. Bir sonraki dört track'tan boot1 yüklenir.

d. boot XENIX file system'den /boot dosyasını yükler.

e. /boot programı çalışmaya başlar ve en sonunda ekrana kullanıcı prompt'u gelir.

/boot programı ve /xenix kernel programı sistemin boot edebilmesi için root file system'de mutlaka bulunmaları gerekir. Harddiskten boot işlemi sistem ROM'una bağlı olarak eğer floppy disk boş ise gerçekleşir.

Boot prompt'u ekrana aşağıdaki gibi gelir.

```
XENIX System V
Boot:
```

Eğer ? işareti girilip RETURN 'a basılır ise system boot edilebilir çevre birimlerin listesini döker. Boot işlemini devam ettirmek için şu adımlar uygulanabilir.

a. RETURN tuşuna basılır ise sistem /etc/default/boot dosyası içindeki DEFBOOTSTR 'nin işaret ettiği string'i kullanarak boot eder.

Örnek II.4.a.

Bir /etc/default/boot dosyası aşağıdadır.

```
DEFBOOTSTR=hd(40)xenix
LOADXENIX=YES
FSCKFIX=YES
MULTIUSER=YES
PANICBOOT=NO
MAPKEY=YES
SERIAL8=NO
SCRATCH=/dev/scratch
```

b. Hiç bir tuşa basmadan beklenir ise ve /etc/default/boot dosyasında LOADXENIX=YES ise bir süre sonra sistem otomatik olarak boot eder.

c. Son olarak boot işlemini devam ettirmek için device ve pathname 'i kullanıcı verebilir. Genel yazılım formatı aşağıdaki gibidir.

```
xx(m,0)filename
  veya
xx(m)filename
```

burada ;

xx = device name (hd = harddisk , fd = floppydisk)

m = minör device number (harddiskteki root file sistem için bu numara 40 , 96 tipi floppy için 52 'dir. Diğer çevre birimlerin minör number'ları için /dev directory'si içine bakılabilir.) 0 = partition içindeki ofset number'dır. Opsiyoneldir ve genellikle 0'dır. filename = standard XENIX pathname'dir. Eğer root directory'de değilse slash ile başlamalı ve full pathname 'i yazılmalıdır.

Eğer sistem boot prompt'undan sonra boot etmez ise ve /xenix kernel programının bir kopyası root'ta bulunuyorsa yada /usr/sys/conf/xenix dosyası var ise aşağıdaki gibi kullanılarak sistemin açılması sağlanabilir.

```
XENIX System V
Boot
: hd(40)/usr/sys/conf/xenix
```

II.4.3. Boot Dosyası Seçenekleri

LOADXENIX=YES : Eğer YES olarak set edilmiş ise boot

dosyası otomatik olarak belli bir süre sonra /xenix dosyasını yükler.

DEFBOOTSTR=string : Otomatik boot ederken kullanılan string'dir. Default string hd(40)xenix 'dir.

RONLYROOT=NO : Root file systemin readonly mount edilmesini sağlar. Normalde sadece install anında YES olarak set edilir.

SYSTTY=x : Boot anında x'in değerine göre system konsolü standart video adaptör yada seri port COM1 yapılır. sio yada 0 = seri port COM1 scrn yada 1 = Display adapter Eğer boot anında display adapter bulunmaz ise boot dosyası otomatik olarak COM1'i 9600 baud , 8 data bit 1 stop bit ve no parity olarak set ederek konsolü yapar.

FSCKFIX=YES : Eğer YES olarak set edilmiş ise root filesystem problemleri fsck-rr komutu ile otomatik olarak halledilir.

MULTIUSER=YES : Eğer YES olarak set edilmiş ise init komutu sistemi multiuser mode götürülebilir.

PANICBOOT=YES : Eğer YES olarak set edilmiş ise sistem panic mesajlarından birisi ile karşılaştığı zaman reboot eder.

Sistemde ikinci bir disk daha var ise birinci diskten boot edip ikinci diskin filesystem'i kullanılmak isteniyorsa /etc/default/boot dosyası içinde aşağıdaki gibi keyword kullanılabilir.

```
disk2=hd(40,0)xenix root=hd(104,0)
prompt="Using second hard disk"
```

boot prompt'u geldiğinde "disk2" girilirse xenix 1.harddiskten yüklenir ve ekrana "Using second hard disk" mesajı gelir. RETURN'e basıldığı zaman xenix ikinci diskteki root file sistemi kullanmaya başlar.

II.4.4. Boot up Esnasında Karşılaşabilecek Hata Mesajları

IO ERR : Masterboot aktif olan işletim sistemini yüklerken ortaya çıkan daha çok hardware kökenli bir arızadır.

BAD TBL : Fdisk table içerisinde bootable partition indicator'un tanımlanmış bir değer taşıması nedeni ile

ortaya çıkan software kökenli bir hatadır.

NO OS : Aktif işletim sistemi yüklenirken ortaya çıkan bir hatadır ve software kökenlidir.

Yukarıdaki hata mesajlarını masterboot ekrana verir ve sistem kilitlenir. Bu durumda yapılacak işlem hata mesajına göre işletim sistemini yeniden yüklemek yada makinanın arızalı parçasını bulup değiştirmektir.

bad magic number : Yüklenmek istenen programın executable olmaması durumunda ortaya çıkan hata mesajıdır.

can't open ‹pathname› : Verilen path'daki dosya bulunamamıştır.

Stage 1 boot failure : Bootstrap loader / boot dosyasını bulamamaktadır yada okuyamamaktadır. Boot dosyası floppy diskten tekrar kopyalanmalıdır.

not a directory : Diskin belirtilen bölgesinde geçerli bir XENIX file system'i bulunamamıştır.

zero length directory : Geçerli bir XENIX dosya sistemi olmasına karşılık belirtilen directory her hangi dosya içermemektedir. Daha çok eski versiyon System XENIX filesystem yapılarında rastlanır.

fload:read(x)=y : x byte okunmasına karşın y kadar byte okunması durumunda ortaya çıkan bir hatadır. Daha çok okunmak istenilen dosyanın bozulması durumunda ortaya çıkar.

Yukarıdaki hata mesajlarını /boot programı ekrana verir , sistem kilitlenmez fakat Boot prompt'una düşer. Bu durumda hatayı gidermenin yolu eksik , bozuk yada silinmiş olan dosyayı yeniden olması gereken yere geri kopyalamaktır.

XENIX sistem multiuser ortama ulaşıncaya kadar dört adımdan geçer. Bunlar sırası ile init,getty,login ve shell programlarıdır.

1. /etc/init PROGRAMI : Bu program sistemdeki terminalleri kullanıcıların sisteme girip çıkabilmeleri için hazırlarlar. Önce /dev/console (sistem konsülü) dosyasını okuma ve yazma modunda açar. /etc/suloğin programını çalıştırarak kullanıcıdan çalışma modlarından birisini seçmesini bekler.

Type CONTROL-d to proceed with normal startup,
(or give root password for system maintenance) :

Bu durumda kullanıcı CONTROL-d tuşuna basaarak multiuser (çok kullanıcılı) modunu seçebilir veya root (superuser) kullanıcısının passwordünü girerek system maintanance (sistem bakım modu) modunu seçebilir. /etc/init programı eğer sistem normal koşullarda kapatılmadı ise (Elektirik kesintisi veya sistemin normal kapama koşullarına uygun olmayan bir biçimde kapatıldıktan sonra tekrar açılması durumunda) /etc/init tarafındn "fsck" programı çalıştırılır. Ve hatalı kapanma esnasında bozulan dosyalar kurtarılabilir.

2. getty PROGRAMI : Terminallere login prompt'unu gönderdikten sonra kullanıcının gireceği kullanıcı adını bekler. Kullanıcı bir kullanıcı ismi girer ise getty programı bunun geçerli bir kullanıcı ismi olup olmadığını login programını çalıştırarak kontrol eder.

3. login PROGRAMI : Bu program /etc/passwd dosyasından girilen ismin doğru olup olmadığını kontrol eder. /etc/passwd dosyadında şu bilgiler bulunabilir.

a/ Kullanıcı ismi
b/ Şifrelenmiş durumda password'u
c/ Kullanıcı id numarası
d/ Grup id numarası
e/ Açıklama bölümü
f/ Home directory'si
g/ Sisteme girdiği anda çalışacak programın ismi

Örnek II.4.b.

etc/passwd dosyasına örnek

```
root:ouMUMLGYkEO5Y:0:0:Super user:/:/bin/sh
        cron:NOLOGIN:1:1:Cron daemon for periodic tasks:/:
bin:NOLOGIN:3:3:System file administration:/:
uucp::4:4:Uucp
administration:/usr/spool/uucppublic:/usr/lib/uucp/uucico
asg:NOLOGIN:6:6:Assignable device administration:/:
sysinfo:NOLOGIN:10:10:Access to system information:/:
network:NOLOGIN:12:12:Mail                and                Network
```

```
administration:/usr/spool/micnet:
lp:NOLOGIN:14:3:Print spooler administration:/usr/spool/lp:
dos:NOLOGIN:16:10:Access to Dos devices:/:
haltsys:YgjWLEo00gFVk:0:0::/etc:/etc/haltsys
ahmet::201:50::/usr/ahmet:/bin/sh
```

4. shell PROGRAMI : login işlemi doğru bir şekilde gerçekleştikten sonra daha önceden install edilmiş shell programı (komut yorumlayıcı) çağrılır. Shell programı kullanıcı prompt'unu ekrana vermeden kulanılan kullanıcının directory'sinde bulunan .profile veya .login programları var ise bunları çalıştırır. Bu dosya içinde kullanıcıya ait bir takım set işlemleri ve komutları bulunur. Kullanıcı editöre girip bunları değiştirebilir.

II.5. XENIX İŞLETİM SİSTEMİNDE DOSYA SİSTEMLERİNİN KULLANILIMI

XENIX işltim sistemi dosyaları düzenli bir şekilde hard disk yeya floppy disk gibi belleklerde saklar ve kullanıma sokar. XENIX işletim sistemi en az bir dosya sistemi içermektedir. Bu dosya sistemine root filesystem denir ve / işareti ile belirtilir. İşletim sistemi tarafından kullanılan bütün dosyalar , programlar root filesystem içerisindedir.

XENIX işletim sistemi disk veya disket üzerine ilk defa yüklenirken root filesystem disk veya disket üzerinde otomatik olarak oluşturulur.

II.5.1. FLOPPY SURUCUDE DOSYA SİSTEMİNİN OLUŞTURULMASI

Boot filesystem disket üzerinde mkdev fd komutu ile oluşturulur.

Örnek II.5.a.

48 tpi disket üzerinde root filesystem (dosya sistemi) oluşturalım.

```
# mkdev fd
```

yazılıp RETURN tuşuna basılır ise ;

```
Choies for type of floppy filesystem
        1. 48 tpi double sided , 9 sectors per track
        2. 96 tpi double sided , 15 sectors per track
        3. 135 tpi double sided , 9 sectors per track
Enter an option or enter q to quit: 1
```

Şeklinde bir menü ile karşılaşılır. 48 tpi disket üzerinde işlem yapacağımız için 1 no'lu seçenek seçilir. Eğer bu menüden q seçeneği seçilecek olur ise yapılacak işlemden vazgeçilir. 1 no'lu seçenek seçilip RETURN tuşuna basılır ise ekranda ;

```
Insert a 48ds9 floppy into drive 0.
Press <RETURN> to continue or enter q to quit:
```

şeklinde bir mesaj görülür. Floppy diske 48 tpi bir disket sürüp RETURN tuşuna basılır ise işleme devam edilir. q ile yapılacak işlemden vazgeçilebilir. İşleme devam etmek için RETURN tuşuna basılır ise ,

```
Choies for contents of floppy filesystem

        1. Filesystem only
        2. Bootable only
        3. Root filesystem only
        4. Root and Boot (only available for 96 tpi floppy)

Enter an option or enter q to quit: 1
```

Eğer işlemden vazgeçmek isteniyor ise q yazıp RETURN tuşuna basılmalıdır. Eğer diskette filesystem oluşturulacak ise 1 yazılıp RETURN 'a basılır ise ekranda ;

```
Would you like to format the floppy first? (y/n)? y
```

şeklinde beliren soruya y cevabı verilecek olur ise floppy diskte bulunan disket aşağıdaki şekilde formatlanır vede içerisinde filesystem oluşturulmuş olur.

```
formatting /dev/rf48ds9 ...
  track 39 head 1
done
isize = 80
m/n = 1 9
Filesysyem creation complete
```

Floppy diskte filesystem oluşturulduktan sonra ekranda aşağıda görülen menü tekrar ouluşur. Bu menüden q seçeneği seçilerek çıkılabilir.

```
Choies for contents of floppy filesystem

        1. Filesystem only
        2. Bootable only
        3. Root filesystem only
        4. Root and Boot (only available for 96 tpi floppy)

Enter an option or enter q to quit: q
#
```

Bu işlemlerin sonunda 48 tpi tipindeki bir disket üzerinde filesystem oluşturulmuş olundu. Bu disketi sisteme eklemek için mkdev fs komutunu yada direkt mount komutu kullanılabilir.

```
# mkdev fs
```

yazılıp RETURN tuşuna basılır ise ,

```
Enter a new directory name and press <RETURN> /mnt
```

konumunda iken directory (dizin) adı verilip RETURN tuşuna basılır ise ekranda aşağıdaki menü oluşur.

```
1. Always mount /dev/fd048 when entering multiuser mode.
2. Never mount /dev/fd048 when entering multiuser mode.
3. Prompt before mounting /dev/fd048 when entering multiuser mo
Select an option: 3
#
```

Bu menüden uygun seçim yapıldığı takdirde sistem /etc/checklist ve etc/default/filesys dosyalarını değiştirerek shell'e geri döner.

II.6. XENIX İŞLETİM SİSTEMİNDE DOSYA YAPISI

II.6.1. ls-il KOMUTU

Bu ls-il komutu içerisinde bulunulan directoy 'de bulunan dosyaların isimleri ve o dosyalarla ilgili bilgileri ekranda gösterir.

II.6.2. File Inode Number KOMUTU

XENIX işletim sisteminde disk 1024 byte'lik bloklara bölünmüştür. Dosyaların içerikleri bu bloklarda saklanır. Bir dosya bir veya daha fazla blok kaplayabilir. Bu blokların adresleri inode table'da saklanır. Öyle ise her dosyanın kendisine ait bir inode numarası vardır. İnode table'deki her bir inode için 64 byte'lik bir alan ayrılmıştır. Bu 64 byte'lik bir alan içerisinde ; dosyanın uzunluğu , kapladığı blokların adresleri , user ve group ID numaraları , açıldığı ve değiştirildiği tarihler gibi o dosyaya ait bir takım bilgiler saklanır. Bir filesystem içerisinde maksimum 65500 inode vardır. İnode numaralarından yararlanılarak , sistem dosyaları birbirine link edilirler. Dosyaların birbirine link edilebilmeleri için in deyimi kullanılır.

Örnek II.6.2.a.

file0 ve file1 dosyalarını birbirine link edilme işi aşağıda verilmiştir.

```
# in file0 file1
```

Bu işlem sonunda file0 ve file1 dosyalarının içerikleri aynı olur ve aynı inode numarasına sahip olurlar. Dosyaların birisinde yapılacak değişiklik diğerinde de gerçekleşir. Ancak dosyaların biri silindiğinde diğeri silinmez.

II.6.3. File Mode and Permissions Komutu

Permission bölümü ile dosyaya erişim haklarını belirler ve değiştirebilir. XENIX işletim sisteminde bir dosyaya erişim read , write ve execute olmak üzere üç çeşittir. Örnek bir dosyanın permission bölümüne bakıldığında en başta dosyanın tipi görülür. Bir dosya 8 farklı tipte olabilir.

1. - ordinary file (sıralı dosya)
2. b block special file (özel blok dosyası)
3. c character special file (özel karakter dosyası)
4. d directory file (dizin dosyası)
5. m shared data file (parçalanmış veri dosyası)
6. n name special file (özel isim dosyası)
7. p pipe file
8. semophore file

daha sonra gelen dokuz karakter sırası ile file owner (dosyanın oluşturucusu) , group-user ve other-user için read , write ve execute haklarını gösterir. Bunlar parametreleri ile ,

r read (okuma hakkı)

w write (yazma hakkı)

x execute (icra etme hakkı)

- no access (erişim hakkı yok)

Links : Permissionslardan sonra gelen rakam o dosyaya bağlı kaç adet dosya olduğunu belirtir.

Örnek II.6.3.a.

```
# l esra
drwr-rxr-x  15 ibrahim proje 120 jul 21:12  esra
#
```

ele alınan esra adlı directory dosyasına bağlı 15 dosya vardır.

Owner : Dosyayı oluşturan kullanıcının ismidir ve chown komutu ile değiştirilebilir.

Örnek II.6.3.b.

```
# l esra
-rwxr-x--x  1 ibrahim proje 120 jul 21:12 esra
# chown cosar esra
# l esra
-rwxr-x--x  1 cosar   proje 120 jul 21:12 esra
```

Group Owner : Dosyanın ait olduğu kullanıcı grubunun ismidir. İstenir ise chgrp komutu ile değiştirilebilir.

Örnek II.6.3.c.

```
# l esra
-rwxr-x--x  1 ibrahim proje 120 jul 21:12 esra
# chgrp group esra
# l esra
-rwxr-x--x  1 ibrahim group 120 jul 21:12 esra
```

File Size : Dosyanın byte cinsinden kapladığı alanı gösterir.

Time Of Last Modification : Dosyanın ismidir. İstenilir ise mv komutu ile dosyanın ismi değiştirilebilir.

Örnek II.6.3.d.

```
# l esra
-rwxr-x--x  1 ibrahim proje 120 jul 21:12 esra
# mv esra ibrahim
# l ibrahim
-rwxr-x--x 1 ibrahim proje 120 jul 21:12 ibrahim
```

II.6.4. mkuser KOMUTU

XENIX işletim sistemine yeni bir kullanıcı eklemek için mkuser komutu kullanılır. Bu komut /etc/password dosyası içerisine login name , password , user ve group ID gibi kullanıcının sisteme girip çalışabilmesi için gerekli bir takım bilgileri yazar. Ayrıca yeni kullanıcı için bir home directory , mail programı için gerekli bir mailbox ve kullanıcının sisteme girdiği anda çalışan .profile veya .login isimli dosyaları hazırlar. Sistemde yeni bir kullanıcı tanımlanır iken ;

mkuser komutu kullanılmadan önce superuser moduna geçilir.

```
# mkuser
```

yazılıp RETURN tuşuna basılır ise aşağıdaki mesaj ekrana gelecektir.

```
Mkuser

Add a user to system

Do you require detailed insructions? (y/n/q):
```

sorusuna y (evet) cevabı verilir ise mkuser komutunun kullanımı ile ilgili bir takım bigiler alınabilir. Aksi takdirde n (hayır) tuşuna basılarak işleme devam ettirilebilir. Yapılacak işlem yapılmaktan vazgeçilmek istenilir ise q (çık) tuşuna basılmalıdır. Programın çalışması devam ettirildiği takdirde ekranda , "yeni kullanıcı ismini giriniz" şeklinde şu soru oluşur.

```
Enter new user's login name:
```

Kullanıcı ismi yazıldıktan sonra , sistem tarafından ID numarası aşağıda belirtildiği şekilde istenir.

```
Do you wish to use the next available user id? (y/n/q):
```

Bu numara 199 sayısından büyük ve halen bir başka kullanıcı tarafından kullanılmıyor olmalıdır. Eğer bu soruya y (evet) cevabı verilir ise program otomatik olarak bir sonraki uygun kullanıcı numarasını verir. Mümkün olan kullanıcı numaraları /etc/passwd dosyasından öğrenilebilir.
Kullanıcı ismi girildikten sonra mkuser programı kullanıcı group numarası ve kullanıcı id numarasını ister. Kullanıcı group numarası isteğe bağlı bir rakamdır. Eğer herhangi bir rakam verilmez ise system kullanıcıyı ortak group numarası olan 50 'ye dahil eder. Group numarası kullanıcının sistemdeki ortak dosya ve kataloglar ile erişim izinlerini belirler. Group numaraları 50'den 30000'e kadar herhangi bir rakam olabilir. Eğer farklı bir Group ID numarası verilecek ise uygun olan numara /etc/group dosyasından öğrenilebilir.

```
Do you want to use the default group? (y/n/q): n
Existing groups are:

        Group "group" (50)ibrahim emiroglu

        Group "proje" (51) ahmet

Do you want to use one of these groups? (y/n/q): n
Please enter number for new group:muhasebe
Please enter number for new group.
Or press ENTER for default number: 55
```

Grup ismi ve Grup ID numarası girildikten sonra oluşturulmuş kullanıcı için şifre girilmesi istenecektir.

```
Please enter at least 5 characters for the password.
Enter password:
```

Bütün bunlardan sonra mkuser programı yüklendikten sonra yapılan işlerin tasdik edilmesi istenir. Eğer bu soruya y (evet) cevabı verilir ise , kullanıcı verilen bilgiler ışığında sistemde oluşturulur.

II.6.5. rmuser KOMUTU

XENIX işletim sisteminden bir kullanıcı çıkartabilmek için

kullanılan bir komuttur. rmuser komutu ile sisteme ait kullanıcı isimleri (root , sys , sysinfo , cron , uucp) silinemez. Bir kullanıcının sistemden çıkartılabilmesi için , kullanıcının directory'sinin içi boş olmalı ve mailbox'u silinmiş olmalıdır. Sistemden bir kullanıcının çıkarılması için , sisteme super user olarak girilir. Daha sonra kulanıcının directory'sine girilir ve directory'deki tüm önemli dosyaların yedeği alınır. Directory'deki tüm dosyalar aşağıdaki şekilde silinerek boşaltılır.

```
# rm *
```

komutu ile directory'deki tüm dosyalar silinir. Kullanıcının mailbox kütüğüde silinmelidir. Bu aşağıdaki şekilde yapılır.

```
#rm /usr/spool/mail/kullanıcı ismi
```

kullanıcının directory'sinden çıkılarak rmuser komutu ile kullanıcı sistemden silinir.

Örnek II.6.5.a.

esra adlı bir kullanıcıyı sistemden çıkartabilmek için , süper-user olarak sisteme girilir ve

```
#cd /usr/esra
```

ile esra adlı alt directory'e girilir. Bu directory'deki önemli programların yedeği alınarak ,

```
#rm *
```

ile directory içerisindeki dosyalar silinir.

```
#cd/usr
```

komutu ile kullanıcının directory'sinden çıkılır ve kullanıcının mailbox'u

```
#rm /usr/spool/mail/esra
```

ile silinir ve ,

```
#rmuser
```

komutu çalıştırılır. Daha sonra ekranda ,

It will ask you for the username, and then delete the corrospon
password file entry, the user's mail box, and home directory.
The following files, If they exist in the user's home directory,
will be removed:

.profile .login .cshrc .mailrc
.exrc .newsrc .plan .project
.UNET.accounts

Before removing a user you should check that the corresponding mail box is empty, and all the files owned by the user have eit been deleted or transferred to some other user. The program wil check this for you, and refuse to remove the user if the check fails.
Press ENTER when you are ready:

Enter login name of user to be removed:esra

Removing user esra from the system. CONFIRM? (y/n/q): y
User esra removed from the system

Do you want to remove another user? (y/n/q): n
#
böylece esra adlı kullanıcı sistemden çıkarılmış olur.

II.6.6. passwd KOMUTU

XENIX işletim sisteminde kullanıcılara ait şifreleri değiştirmek için kullanılan bir komuttur.Sisteme super-user olarak girildiği takdirde , sistemdeki tüm kullanıcıların şifreleri değiştirilebilir.
Genel Yazılım Formatı :

#passwd

root kullanıcısının şifresini değiştirebilmek için , root kullanıcısının şifresi mutlaka bilinmelidir.
Örnek II.6.7.a.
Sistemde bulunan emir adlı kullanıcının şifresini değiştirebilmek için ,
sisteme super-user olarak girilir ve

#passwd emir

komutu icra ettirilir ise , kullanıcının şifresi 5 karakterden fazla olmak koşulu ile değiştirilir. "New password" ile yeni şifre girildikten sonra , yeni şifrenin yazılımında herhangi bir hata olup olmadığını anlamak için girilen yeni şifrenin , "Re-enter new password" ile tekrar yazılması istenir. Eğer "New password" ile yazılan şifre "Re-enter password" yazılan şifre aynı ise kullanıcının şifresi değiştirilmiş olur. Eğer her iki şifre aynı değil

ise şifre değiştirme işlemi gerçekleşmez.

```
Enter new password (minumum of 5 characters)
Please use a combination of upper and lowercase letters and numbers
New password:
Re-enter new password:
```

II.6.7. pwadmin KOMUTU

Kullanıcı şifrelerini , verilen zaman aralığında değiştirmek için kullanılan bir komuttur. Bu komut ile 0 ile 63 hafta arasında bir değiştirme zamanı tanımlanabilir.

Örnek II.6.8.a.

esra adlı kullanıcının şifresini minumum 4 hafta , maksimum 9 hafta içerisinde şifresinin değiştirilmesi isteniyor ise;

```
#pwadmin -min 4 -max 9 esra
```

komutu icra ettirilmelidir. Eğer kullanıcının şifresi 4 ile 9 hafta arasında değiştirilmez ise , 9. hafta bittiğinde şifrenin değiştirilmesini ototmatik olarak isteyecektir.

minumum ve maksimum değerleri öğrenmek için ;

```
#pwadmin -d esra
```

komutu icra ettirilmelidir. Bu komutun icrası sonucu ekranda ;

```
Minumum weeks: 4.
Maximum weeks: 9.
```

şeklinde bir mesaj oluşur.

Kullanıcının acilen şifresini değiştirmek için ;

```
#pwadmin -f esra
```

komutu icra ettirilmelidir.

Periyodik şifre uygulamasını kaldırmak için ;

```
#pwadmin -n esra
```

komutu icra ettirilmelidir.

II.7. XENIX İşletim Sisteminde DOS Komutlarının Kullanımı

1. doscat : DOS disketindeki dosyaların içeriklerini ekrana döker.
2. doscp : DOS ortamından XENIX ortamına dosya transferi için kullanılır.
3. dosdir : DOS ortamındaki directory listesini verir.
4. dosformat : DOS 2.0 formatlı disket oluşturur.
5. dosls : DOS directory'sini XENIX'in ls komutu

formatında listeler.

6. dosmkdir : DOS disketinde directory oluşturur.
7. dosrm : DOS disk'inde dosya siler.
8. dosrmdir : DOS disk'inde directory siler.

II.8. XENIX İŞLETİM SİSTEMİNDE YEDEKLEME ve YENİLEME KOMUTLARI

XENIX işletim sisteminde , DOS işletim sistemindeki BACKUP ve RESTORE komutlarına karşılık sysadmin komutu vardır. Günlük yada periyodik yedekleme için zaman kazandırıcı bir algoritma uygulayan bu program dört ana bölümden oluşur. Bunun birinci bölümü yedekleme yönteminin seçimi , ikinci bölüm yedekleme yapılacak alanın seçimi (tüm yada özel dosyalar) , üçüncü bölüm yedekleme ortamının seçimini (teyp veya disket sürücü seçimi) , dördüncü bölüm ise yenileme işleminin seçimidir.

Yedekleme uygulaması için yapılacak en kolay yol önce sistemin komple yedeklenmesi , takip edilen günlerde ise sadece güncelleştirilen veya yeni oluşturulan dosyaların yedeklenmesidir. XENIX işkletim sisteminde seviyeli bir yedekleme serisi kullanılır. Toplam 10 seviye vardır. Bunlar ;

0. Seviye : Sistemdeki bütün dosyalar yedeklenir.
1. Seviye : Seviye 0'dan sonraki güncelleştirilmiş (değiştirilmiş) olan dosyalar yedeklenir.
2. Seviye : Seviye 0 ve 1 'den sonra değiştirilmiş olan dosyalar yedeklenir.
3. Seviye : Seviye 0 , 1 ve 2 'den sonra değiştirilmiş olan dosyalar yedeklenir.

. .
. .
. .

9. Seviye : Seviye 0 , 1 . . . , 8 'den sonra değiştirilmiş olan dosyalar yedeklenir.

Herbir seviye bir önceki sevileyi yedeklemeden bu yana olan değişiklikleri yedekler.

Bir önceki yedekleme seviyeleri /etc/ddate dosyası içinde saklanır. Böylece yeniden yedekleme işlemi gerektiğinde bir önceki başarılı yedekleme işleminin tarihi bilinir ve o

tarihten sonraki değiştirilmiş yada oluşturulmuş olan dosyalar yedeklenir. Yedekleme işlemi 0. seviyede yapılmış ise /etc/ddate dosyasına bakılmaksızın tüm sistemin yedeklemesi yeniden yapılır.

II.8.1. sysadmin Komutunun Kullanılması

XENIX işletim sisteminde sysadmin komutu kullanılarak seviye 0 ve 9'da tüm sistemin veya özel dosyaların yedekleme ve yenilemesi yapılabilir. Bu aşağıdaki şekilde yapılır.

```
# sysadmin
```

komutu icra ettirilir. Daha sonra ekranda ;

```
Filesystem Maintenance Options
        1. Do a daily backup (level 9)
        2. Do a periodic backup (level 0)
        3. Get a backup listing
        4. Restore backed up file(s)
        5. Restore an entrie filesystem

Enter an option or enter q to quit:
```

Yukarıda verilen menüde 1. seçenek seçilir ise günlük yedekleme , 2. seçenek ise periyodik yedekleme işlemini yapar. 1. ve 2. seçeneklerden herhangi birisine girildiği zaman ekranda aşağıdaki menü oluşur.

```
      1. / The root filesystem
      2. Other
Select the filesystem that you wish to backup,
or enter q to return to the main menu:
```

İhtiyaca uygun seçenek seçildiği zaman sysadmin komutu hangi sürücüye yedekleme yapılacağını aşağıdaki şekilde sorar.

```
      1. Floppy Drive 0 (48dsdd)
      2. Floppy Drive 1 (48dsdd)
      3. Floppy Drive 0 (96dshd)
      4. Floppy Drive 1 (96dshd)
      5. Floppy Drive 0 (96dsdd)
      6. Floppy Drive 1 (98dsdd)
      7. Floppy Drive 0 (3.5 in dsdd)
```

8. Floppy Drive 1 (3.5 in dsdd)
9. Floppy Drive 0 (3.5 in dshd)
10. Floppy Drive 1 (3.5 in dshd)
11. Cartridge Drive (300 ft tape)
12. Cartridge Drive (450 ft tape)
13. Cartridge Drive (600 ft tape)
14. Mini-Cartridge Drive (10MB)
15. Mini-Cartridge Drive (40MB)
16. Other

Select an archive device,
or enter q to return to the main menu:

Yukarıda belirtilen menuden yedekleme yapılacak sürücü tipi belirlendikten sonra yedekleme işlemi gerçekleşir.

sysadmin komutu bu iki menüyü hazırlar iken , /etc/default/filesys ve /etc/default/archive dosyalarından yararlanır. Yedekleme yapılacak sürücü tipi belirlendikten sonra , sürücünün formatlı olup olmadığını sorar , eğer disket formatlı değil ise formatlanır ve yedekleme işlemi başlar.

Ana menüde belirtilen 3. seçenek yedekleme listesini verir.

Ana menüde belirtilen 4. ve 5. seçenekler yedeklenmiş dosyaların yenilenmesinidir. 4. veya 5. seçenekler seçildiğinde ekranda ;

1. Restore file(s) to their original location(s)
2. Restore file(s) to another location

Enter an option, or enter q to return to the main menu:

şeklinde bir menü oluşur.

1. seçenek seçildiği takdirde dosyaları eski yerlerine yenileme işlemini yapar.

1. seçenek seçildiği takdirde dosyaları başka bir yere yenileme işlemini yapar.

II.8.2. tar KOMUTU

Bu komut disket backup (yedekleme) veya restore (yenileme) işlemi için kullanılan bir komuttur.

Genel Yazılım Formatı :

```
# tar [tercih] [sürücü tipi] [file adı]
```

[tercih] , ile c , f , t , v , x , v , w , f , d , 0-9

yazılabilir.

c , ismi yazılmış file'ların içeriği diskete yedeklenir. Daha önce diskette bulunan kütükler silinir.

f , tar komutu bu tercihin arkasından aygıt adı bekler.

t , dosyaları ekranda gösterir. Kopyalama yapmaz.

v , dosyalar kopyalanırken ekranda gösterilir.

x , ismi belirtilmiş file adları restore (yenileme) edilir. Yenileme yapılır iken kullanılır.

w , yedeklenecek yada yenilenecek file'lar için kullanıcıdan "y" veya "n" cevabını bekler. Tar komutunu karşılıklı soru-cevap şeklinde çalıştırır.

d , tarihli yedekleme için kullanılır.

0-9 , tar komutunun tanıdığı sürücü cihazlarıdır. Bunun için /etc/default/archive dosyasına bakılmalıdır.

Örnek II.8.2.a.

#tar cv2f /dev/rfd048 gs.cob

komutunun icrası sonucu gs.cob adlı dosya 360K'lık bir diskete kopyalanır.

II.8.3. cpio KOMUTU

Diskten diskete veya disketten diske kopyalama işlemi yapmak için kullanılan bir komuttur.

Genel Yazılım Formatı :

cpio -o [tercih] [sürücü tipi]

bu yazılım şekline göre diskten diskete belirtilen dosyalar kopyalanır.

cpio -i [tercih] [sürücü tipi]

bu yazılım şekline göre disketten diske belirtilen dosyalar kopyalanır.

[tercih] , ile o , i , B , d , c , t , u , v belirtilebilir.

o , yedekleme için kullanılır ,

i , yenileme işleminde kullanılır ,

B , blok olarak kopyalar ,

d , directory oluşturur ,

c , kopyanın başına bir ön bilgi yazar ,

t , kopya yapmaz , dosyaları sadece ekranda gösterir.

u , kopyalama yapılır iken eğer aynı dosyadan var ise

üzerine yazılmasını sağlar ,
v , kopya anında dosyaları ekranda gösterir.
Örnek II.8.3.a.
360K'lık bir disket içerisindeki tüm dosyaları disk'e kopyalayabilmek için ,

cpio -icvB < /dev/rf048

komutu icra ettirilmelidir.
Örnek II.8.3.b.
diskte bulunan gs.cob adlı dosyayı 360K'lık bir diskete kopyalayabilmek için ,

ls gs.cob | cpio -ocvB > /dev/rfd048

komutu icra ettirilmelidir.

II.9. vi EDİTÖRÜ

Vi editörü metin satırlarının pencere yada ekran olarak adlandırılan belli bir kısmını ekrana getirerek çalışır. Kullanıcı vi içerisinde iken , ekranda bulunan metin üzerinde istediği değişiklikleri yapabilir.
vi komutu ile birlikte çağrılacak kütük aşağıdaki şekilde çağrılmalıdır.

[d:] vi <kütükadı>

[d:] , ile kullanıcının bulunduğu shell belirtilmektedir.
Örnek II.9.a.
Super-user konumunda iken , gs.cob adlı bir kütük açılmak istenildiği zaman ,

vi gs.cob

komutu icra ettirilmelidir.

II.9.1. vi Editöründe Ekran Hareketleri

ESC tuşuna basılarak ekran hareketleri aşağıdaki şekilde yapılır.
Ctrl-F : Ekran ileri ,
Ctrl-B : Ekran geri ,
Ctrl-D : Yarım ekran ileri ,
Ctrl-U : Yarım ekran yukarı ,
Bu komutlar kullanıldıktan sonra tekrar ESC tuşuna basılarak metin yazma moduna geçilir.

II.9.2. vi Editöründe Kürsor Hareketleri

ESC tuşuna basılarak kürsor hareketleri aşağıdaki şekilde yapılır.

H : Kürsor'u ekranın en üst satırına ,

M : Kürsor'u ekranın orta satırına ,

L : Kürsor'u ekranın en son satırına ,

Bir metin kütüğüne giriş yapmak a veya i komutları ile sağlanır. Kürsorun bulunduğu konumdan itibaren bu tuşlarla araya metin girişi yapılır. ESC tuşu ile bu işleme son verilir.

Bir metin kütüğünde herhangi bir kelimeyi silmek için , kürsor silinecek olan kelimenin başına getirilerek dw tuşuna basılmalıdır.

Bir metin kütüğünde herhangi bir kelimeyi değiştirmek için, kürsor değiştirilecek olan kelimenin başına getirilerek cw tuşuna basılmalıdır.

Bir metin kütüğünde belirli bir yeri kopyalamak için y veya YY komutu kullanılır. YY veya y komutundan önce kopya edilecek satır sayısı verilir. YY veya y komutunun ardından P komutu ile belirlenen metin kütüğü kopyalanır. ESC ile bu işleme son verilir.

Kürsor'un üzerinde bulunduğu karakteri değiştirmek için r veya R komutlarından herhangi birisi kullanılır.

Karakter silme işlemi x veya X komutu ile yapılır.

x : kürsor ile silinecek karakterin üzerine gidilir ve x komutu ile kürsorun üzerinde bulunduğu karakter silinir. Birden fazla karakter silmek için silinecek karakter sayısı kadar x tuşuna basılır. Burada 3x ile xxx aynı işlemi yapar.

X : x komutunun tersine silme işlemi kürsorun bulunduğu yerden sola doğru yapılır. Diğer işlemler x komutu ile aynıdır.

Kürsorun üzerinde bulunduğu satırı silmek için dd komutu kullanılır. 1 dd ile 1 adet satır silinir. 5 dd ile 5 satır silinir.

Kürsorun bulunduğu satırın sağından itibaren satırın tamamını silmek için D komutu kullanılır.

Herhangi bir satır üzerinde yapılan değişiklikler u veya U komutu ile iptal edilir.

U : Bulunan satır üzerindeki yapılan tüm değişiklikleri iptal etmek için kullanılır.

u : yapılmış olan en son değişikliği iptal etmek için kullanılır.

I : Satırın başına bilgi eklemek için kullanılır.

A : Satırın sonuna bilgi eklemek için kullanılır.

Ctrl-G komutu kürsor'un bulunduğu satır ve sütünu belirtmek için kullanılır. Ctrl-G ile herhangi bir sayı verilmiş ise kürsor bulunduğu konumdan itibaren belirtilen sayı kadar ileri gider.

Yukarıda verilmiş olan tüm komutlar ESC tuşuna basıldıktan sonra sağlanır.

II.9.3. vi Editöründen Çıkış

Vi editöründen çıkış iki şekilde yapılır.

Metin üzerinde yapılmış olan değişiklikler yüklenmek istenir ise , :x (iki nokta üstüste x) komutu kullanılmalıdır.

Metin üzerinde yapılmış olan değişiklikler yüklenmek istenmediği takdirde , :q! (iki nokta üstüste q!) komutu kullanılmalıdır.

III. BÖLÜM

BASIC PROGRAMLAMA DİLİ

III.1. BASIC PROGRAMLAMA DİLİNE GİRİŞ

III.1.1. Interpritter Olan Basic Türleri

Interpritter olan BASIC.COM , BASICA.COM , GWASIC.EXE'lerde PROMPT 'ta iken program uzantısı BASIC , BASICA , GWBASIC yazıldıktan sonra ENTER ' e basılarak yazılım ortamına geçilir.

Örnek III.1.a.

```
A>BASIC
A>BASICA
A>GWBASIC
```

ortama geçildiğinde ekranın üst kısmında kullanılan BASIC dilinin versiyonu ve bize memory'de ayrılan alan belirtilmektedir.

Bu yazılım GWBASIC için aşağıda verilmiştir.

```
GW-BASIC 2.02
(C) Copyright Microsoft 1983,1984

Compatibility Software GW-BASIC V2.02
Copyright (c) 1984 by Phoenix Software Associates Ltd.
Copyright (c) 1985 Commodore Electronics Ltd. All Rights Reserved

62194 Bytes free
Ok
```

Bu tür BASIC türlerinde her satıra bir numara vermek gerekir verilmediği takdirde o andaki komut icra edilir.

Bu tür BASIC türlerinde bazı F tuşlarının görevleri aşağıdaki gibidir.

F1 : LIST (hafızada bulunan programı listeler).

F2 : RUN (hafızada bulunan programı çalıştırır).

F3 : LOAD (diskette vaya diskteki herhangi bir programı çağırmamızı sağlar).

F4 : SAVE (Hafızada bulunan programı diskete veya diske yükler).

III.1.2. Compiler Yöntemi İle Çalışan Basic Türleri

Bunlar TURBO BASIC (TB.EXE) , QUICK BASIC (QB.EXE) dir. Bunlar promptta iken hangi Basic kullanılıyor ise file adı yazılarak program yazma ortamına girilir.

Örnek III.1.b.

Turbo Basic editörüne girebilmek için ; A sürücüsüne TB.EXE komutunu içeren disket takılarak ,

A>TB

komutu icra ettirilmelidir.

Örnek III.1.c.

Quick Basic editörüne girebilmek için ; A sürücüsüne QB.EXE komutunu içeren disket takılarak ,

A>QB

komutu icra ettirilmelidir.

Bunlardan birinin içine girildiğinde karşımıza menü gelir. Menüde gerekli seçimler yapılarak program yazma ortamına gireriz. Bu tür basic 'ler de program yazma işleminde satır numarası vermeye gerek yoktur. Her komut ENTER 'le geçilebilir. Programın çalıştırılabilmesi için : Turbo basic 'de (ALT-R) tuşuna Quick Basic'de F10 tusuna basıldıktan sonra karşımıza çıkan menüden gerekli seçimler yapılmalıdır.

III.2. BASIC'TE KULLANILAN OPERATÖRLER

Basic programlama dilinde aşağıdaki tanımlanmış olan operatörler kullanılabilir.

+ : TOPLAMA İŞLEMİ.
- : ÇIKARMA İŞLEMİ.
* : ÇARPMA İŞLEMİ.
/ : BÖLME İŞLEMİ.
^ : ÜST ALMA İŞLEMİ.
\ : BİR BÖLME İŞLEMİNİN SONUCUNU TAM SAYIYA ÇEVİRİR.
MOD : KALAN BULMA.

BASIC'TE KARŞILAŞTIRMA OPERATÖRLERİ

= : EŞİTTİR.
<> : EŞİT DEĞİL.
< : KÜÇÜK.
> : BÜYÜK.

<= : KUÇUK EŞIT.

>= : BUYUK EŞIT.

III.3. MANTIKSAL OPERATÖRLER

NOT : DEĞIL ANLAMINDADIR.

OR : VEYA ANLAMINDADIR.

AND : VE ANLAMINDADIR.

Örnek III.3.a.

10 IF A<5 AND A>1 THEN 50

Eğer A sayısı 1-5 arasında ise 50 no'lu satıra git.

10 IF A>5 OR B<1 THEN 50

Eğer A sayısı 5'den büyük veya B sayısı 1'den küçük ise 50 no'lu satıra git.

III.4. BASIC PROGRAM YAPISI

Basic programlama dilinde yazılan her satıra satır numarası verilir. Programı sona erdirmek istediğimiz zaman programın son satırına END yazılır.

III.5. BASIC PROGRAMLAMA DILINDE DEĞIŞKEN (VERI) TIPLERI

BASIC programlama dilinde her harfe bir değişken olarak atanmakla beraber, her harfin yanına bir veya birkaç rakam yazılarak da değisken tanımlamak mümkündür. Bir değişkenin karakter uzunluğu maksimum 8 karakter olmalıdır.

Örnek III.5.a.

A , B , A1 , A2 , Z , AHMET , gibi tanımlanmış olan değişkenler geçerli değişkenlerdir.

Örnek III.5.b.

A-B , AHMET ALI , MEMURKAYIT , gibi değişkenler geçerli olmayan değişkenlerdir.

BASIC programlama dilinde 3 tip değişken vardır.

III.5.1. Reel Değişkenler

Gerçel sayılaran oluşan (ondalıklı sayılar) değişkenlerdir.

Örnek III.5.c.

A=8.0 , B=17.957

gibi.

III.5.2. Tamsayı Değişkenler

içerikleri tam sayılardan oluşan değişkenlerdir. (değişken tanımına ek olarak % işaretinin değişkene eklenmesi ile olur.)

Örnek III.5.d.

A%=15 , B%=17 , C%=50

gibi .

III.5.3. Karakter Tipi Değişkenler

Gerçel sayı değişkenlerinin yanına $ işareti ilave edilerek bilgiler sayısal olmaktan kurtarılmaktadır. Bu tip değişkenlerle 4 işlem yapılamaz.

Örnek III.5.e.

G$="1 HAZIRAN 1989" , GS$="Bir gerçek ortaya çıktı"

burada G$ ve GS$ adlı değişkenler karakter tipinde tanımlanmışlardır.

III.6. BASIC PROGRAMLAMA DILINDE DIZILER

Aynı tipteki elemanların yanyana sıralanışı ile elde edilen bir bilgi kümesidir. BASIC programlama dilinde bir dizinin eleman sayısı 10'dan fazla ise diziye eleman sayısı kadar boyut açmak gerekir. Bu işlem DIM deyimi ile gerçekleştirilir. DIM deyimi programda bir kez ve programın en başında kullanılmalıdır.

Kullanıcının diziler ve matrisler ile karşılaşabileceği hatalar şunlardır.

Subscript out of range

Hata numarası 9'dur. Anlamı dizi kullanılmış fakat yeterince boyut açılmamış.

Örnek III.6.a.

```
10 CLS
20 DIM A(100),B(47),C$(15),D$(100)
```

Bu yazılıma göre ;

A isimli sayısal dizinin maksimum eleman sayısı , 100'dür.
B isimli sayısal dizinin maksimum eleman sayısı , 47'dir.
C$ isimli karakter dizinin maksimum eleman sayısı ,100'dür.
D$ isimli karakter dizinin maksimum eleman sayısı ,100'dür.

Matrislerde de boyut açma işlemi aynı şekilde yapılır.

Örnek III.6.b.

```
10 CLS
20 DIM A(10,10),B(20,15),C$(10,17)
```

Tanımlamasına göre A matrisi 10 satır 10 sütun'dan , B matrisi 20 satır 15 sütundan , C$ matrisi 10 satır 17

sütundan oluşmaktadır.

III.7. BASİC PROGRAMLAMA DİLİNDE GİRİŞ-ÇIKIŞ İŞLEMLERİ

Basic'de bilgisayar belleğine bilgi girişi ya program içerisine atama deyimleri ile veya READ , INPUT deyimleri ile olur.

Bilgi çıkışı ise PRINT deyimi ile gerçekleştirilir.

III.7.1. INPUT DEYİMİ

INPUT deyimi klavyeden değişkenlere bilgi atamak için kullanılır.Genel Formatı aşağıdaki gibi yazılır.

‹Satır No› INPUT ‹Okutulacak Değişkenler›

Örnek III.7.a.

```
10 INPUT "A=",A
20 INPUT "B=",B
```

program parçacığı çalıştırılır ise ekrada A ve B sayısının değerlerini girilmesini bekleyen ileti ekranda aşağıdaki gibi olurdu.

"A="

A sayısına verilmek istenen değer yazılıp RETURN tuşuna basılır ise ekranda :

"B="

şeklinde bir mesaj oluşur ve B sayısının değeri girilir.

III.7.2. READ DEYİMİ

READ deyimi ile DATA vererek bilgi okutmak istiyor isek READ deyimi ile birlikte mutlaka DATA deyimini kullanmak gerekmektedir. DATA deyimi ile beraber bilgiye atamak istediğimiz değişkenlerin değerlerini vermeliyiz.

Genel Formatı aşağıda yazıldığı gibidir.

‹Satır No› READ ‹Okutulacak Değişkenler›

‹Satır No› DATA ‹Değişkenlerin Değerleri›

Örnek III.7.b.

```
10 READ GS$,G$
20 DATA "1.HAZİRAN 1989","PERŞEMBE'dir"
```

Program parçacığı çalıştırıldığı takdirde READ deyimi ile verilen GS$ değişkenine "1.HAZİRAN 1989" , G$ değişkenine ise "PERŞEMBE'dir" değerleri atanmış olur. Kullanıcının READ deyimini kullanırken karşılaşabileceği hatalar şunlardır.

Out Of DATA in ‹Satır no›

hata numarası 4'tür. Anlamı belirtilen satırda kullanılan READ deyimi için yeterli DATA yazılmamıştır.

III.7.3. PRINT DEYİMİ

PRINT deyimi ile bir değişken değerinin veya sabit bir değerin ekrana çıkış işlemi yapılır.

Genel yazılış formatı aşağıdaki gibidir.

‹Satır No› PRINT ‹Yazılacak Değişkenler›

Örnek III.7.c.

```
10 A=5:B=12
20 C=A+B
30 PRINT "TOPLAM=";C
40 END
```

Program parçacığının çalışması sonucu ekranda TOPLAM=17 yazısı görülür.

III.7.4. LPRINT DEYİMİ

LPRINT deyimi ile bir değişken değeri veya sabit bir değerin printerden çıkışı sağlanır.

Örnek III.7.d.

```
10 A=5:B=12
20 C=A+B
30 LPRINT "TOPLAM=";C
40 END
```

Program parçacığının çalışması sonucu yazıcıya takılmış olan kağıda TOPLAM=17 yazısı yazılır.

III.7.5. LOCATE DEYİMİ

Kürsor'ü ekranın belirli bir satır ve sütününa getirmek için kullanılan bir komuttur.

Genel Yazılım Formatı :

‹Satır no› LOCATE X,Y

Burada X ile , kürsorun ekranın hangi satırında olacağı belirtilmektedir. 1 ile 25 sayıları arasında herhangi bir tam sayı olabilir. Ekranın 25. satırını kullanabilmek için KEY OFF veya Ctrl-T komutunun icra ettirilmesi gerekir.

Y ile , kürsorun ekranın hangi sütününda olacağı belirtilmektedir. 1 ile 80 sayıları arasında herhangi bir tam sayı olabilir.

Örnek III.7.e.

Ekranın 10. satır 17. sütününda A sayısının değerini okutabilmek için ,

```
10 CLS
20 LOCATE 10,17:INPUT "a=",A
```

program parçacığı icra ettirilmelidir.

III.7.6. INPUT$ DEYİMİ

Çalışmakta olan bir programı herhangi bir veya birkaç tuşa basıncaya kadar bekletmek için kullanılan bir komuttur.

Genel Yazılım Formatı :

‹Satır no› ‹Karakter tipi değişken›=INPUT$(‹tam sayı›)

Genel formda belirtilen tamsayı , INPUT$ komutu ile durdurulmakta olan programın kaç tuşa basıncaya kadar durması gerektiğini bildirmek için kullanılır.

Örnek III.7.f.

```
10 CLS
20 A$=INPUT$(3)
30 PRINT "Üç tuşa basıldı"
40 END
```

programı çalıştırıldığı takdirde , ekran CLS komutu ile temizlendikten sonra , herhangi üç tuşa basılıncaya kadar program durur. Başka bir deyişle 30 no'lu satırda bulunan PRINT komutunun icra görebilmesi için herhangi üç tuşa basılmalıdır.

III.7.7. INKEY$ DEYİMİ

Çalışmakta olan program parçacığını herhangi bir tuşa basıncaya kadar durdurmak için kullanılan bir deyimdir.

Genel Yazılım Formatı :

‹Satır no› ‹Karakter tipi değişken›=INKEY$

Örnek III.7.g.

```
10 A$=INKEY$:IF A$="" THEN 10
20 PRINT "herhani bir tuşa basıldı....! "
30 PRINT "Basılan tuş ";A$; "tuşudur."
40 END
```

programı çalıştırıldığı takdirde 20 no'lu satırdaki PRINT komutunun icra görmesi için herhangi bir tuşa basılmalıdır.

III.8. BASIC PROGRAMLAMA DILINDE KULLANILAN KONTROL KOMUTLARI

Kontrol deyimlerinden birisini kullanırken ortaya çıkabilecek hatalar şunlardır.

Undefined line number in ‹Satır no›

hata numarası 8'dir. Anlamı ; hesaplayıcı herhangi bir kontrol komutu ile belirtilen satır numarasından , programda olmayan bir satır numarasına gönderilmek istenmiştir.

III.8.1. GOTO DEYİMİ

Bir işlemin hangi numaralı deyime sapacağını gösterir. Genel yazılış formatı aşağıdaki gibidir.

‹Satır No› GOTO ‹Sapılacak Satır No›

Örnek III.8.a.

```
10  CLS
20  INPUT A,B
30  GOTO 100
40  END
100 C=A+B
110 PRINT "C=";C
```

Program parçacığının çalışması sonucu 10 no'lu satırda ekran temizlenip 20 no'lu satır da A ve B değerleri ekrandan okutulduktan sonra GOTO 100 deyimi ile programın akışı 100 no'lu satıra kayar. C=A+B işlemini yaptıktan sonra C'nin değerini ekrana yazar.

III.8.2. IF-THEN-ELSE DEYİMİ

Bir karşılaştırma işlemi sonucunda işlemin bu karşılaştırma koşuluna göre dallanma veya sapma veya atlama sağlamaktadır.

Genel yazılış formatı aşağıdaki gibidir.

‹Satır No› IF ‹Karşılaştırma› THEN ‹Deyim No› ELSE ‹Deyim No›

Örnek III.8.b.

```
10 CLS
20 INPUT A,B
30 IF A<B THEN 40 ELSE 60
40 PRINT "A SAYISI B SAYISINDAN KUÇUKTUR"
50 GOTO 70
60 PRINT "A SAYISI B SAYISINA EŞITTIR VEYA BÜYÜKTÜR"
70 END
```

III.8.3. ON ... GOTO DEYİMİ

Herhangi bir nümerik değişkenin birden çok farklı durumları için programın herhangi bir satırına sapma yapmak için kullanılan bir deyimdir.

Genel yazılış formatı aşağıda ki gibidir.

‹Satır No› ON ‹Nümerik Değişken› GOTO ‹Sapılacak Olan satırlar›

Örnek III.8.c.

```
10 CLS
20 PRINT "---- M E N U ----"
30 PRINT "‹1›.KAYIT YAPMA"
40 PRINT "‹2›.KAYIT GÖRME"
50 PRINT "‹3›.KAYIT DEĞIŞIKLIK"
60 PRINT "‹4›.KAYIT İPTAL"
70 PRINT "‹5›.LİSTELEME"
80 PRINT "‹6›.PROGRAM SONU"
90 INPUT "SEÇİMİNİZ...‹1›.‹2›.‹3›.‹4›.‹5›.‹6›...",A
100 IF A‹1 OR A›6 THEN 90
110 ON A GOTO 120,200,300,400,500,600
120 REM ********** KAYIT YAPMA **********
130 GOTO 90
200 REM ********** KAYIT GÖRME **********
210 GOTO 90
300 REM ********** KAYIT DEĞIŞIKLIK **********
310 GOTO 90
400 REM ********** KAYIT İPTAL **********
410 GOTO 90
500 REM ********** LİSTELEME **********
510 GOTO 90
600 REM ********** PROGRAM SONU **********
610 END
```

Programının çalışması sonucu 90 no'lu satırda okutulan A değeri için ;

A 'nın değeri 1 ise : 120 nolu satıra git.

A 'nın değeri 2 ise : 200 nolu satıra git.

A 'nın değeri 3 ise : 300 nolu satıra git.

A 'nın değeri 4 ise : 400 nolu satıra git.

A 'nın değeri 5 ise : 500 nolu satıra git.

A 'nın değeri 6 ise : 600 nolu satıra git.

III.8.4. FOR-NEXT DEYİMİ

Basic programlama dilinde bir işi bir çok kez yapmak istiyor isek FOR-NEXT deyimi kullanılır.

Genel yazılış formatı aşağıdaki gibidir.

‹Satır No› FOR ‹İndis Değişken› = ‹Başlangıç Değeri› TO ‹Bitiş Değeri› STEP ‹Artım Miktarı›

Bu yazılış şekline göre FOR ile birlikte verilen indis değişkeni nümerik bir değişken olmalıdır. İndis değişkenin başlangıç değeri ile verilen değerden başlayarak her adımda artma sayısı kadar artması ile NEXT deyimini görünceye kadar işlem yapar.

Her FOR deyiminin mutlaka bir tane NEXT 'i olmalıdır.

Örnek III.8.d.

1'den 100'e kadar olan sayıların aritmetik ortalamasını bulacak bir bilgisayar programı yapınız.

```
10 CLS
20 FOR I=1 TO 100
30 T=T+I
40 NEXT I
50 ORT=T/100
60 PRINT "ORTALAMA=";ORT
70 END
```

Programında I değişkeninin başlangıç değeri 1'den başlayarak 100 oluncaya kadar FOR ile NEXT deyimi arasında işlem yapılır.

Kullanıcının FOR-NEXT deyimi ile ilgili karşılaşabileceği hatalar şunlardır.

NEXT without FOR in ‹Satır no›

Hata numarası 1'dir. Anlamı belirtilen satırda kullanılan NEXT deyiminin FOR deyimi yoktur. (FOR deyimi kullanılmadan NEXT deyimi kullanılmış.)

FOR without NEXT in ‹Satır no›

Hata numarası 26'dır. Anlamı belirtilen satırda kullanılan FOR deyiminin NEXT deyimi yoktur. (NEXT deyimi kullanılmadan FOR deyimi kullanılmış.)

Örnek III.8.e.

LAGRANGE enterpolasyon yöntemi ile bir fonksiyonun belirli

noktalarındaki değerlerini hesaplayacak programı yapınız.

```
10 REM **** LAGRANGE ENTERPOLASYON YÖNTEMİ ****
20 CLS
30 INPUT "Verilecek nokta sayısı .....: ",N
40 DIM X(N),Y(N)
50 FOR I=1 TO N
60 LOCATE 2,1:PRINT "X(";I;")=";:INPUT X(I)
70 LOCATE 3,1:PRINT  X(I);"  değerine  karşılık   gelen
fonksiyon değeri..";
80 INPUT Y(I)
90 NEXT I
100 LOCATE 4,1:INPUT "fonksiyonun hangi  noktadaki  değeri
isteniliyor..:",A
110 FOR K=1 TO N
120 P=1:R=1
130 FOR I=1 TO N
140 IF I=K THEN 160
150 P=P*(X(K)-X(I)):R=R*(A-X(I))
160 NEXT I
170 T=T+R*Y(K)/P
180 NEXT K
190 LOCATE 5,1:PRINT  "fonksiyonun  ";A;
200 PRINT "noktasındaki değeri = ";T
210 END
```

Örnek III.8.f.

Okutulan herhangi bir kare matrisin determinantını hesaplayacak bilgisayar programını yapınız.

```
10 REM **** DETERMINANT HESABI ****
20 CLS
30 INPUT "   Determinantın Boyutunu giriniz   ",R
40 DIM Z(R+1,R+1)
50 FOR I=1 TO R:FOR J=1 TO R:
60 READ Z(I,J)
70 NEXT J:NEXT I
80 DATA 1,2,3,4,-2,0,0,4,5,1,1,-2,1,0,-4,1
90 CLS
100 FOR I=1 TO R:FOR J=1 TO R
```

```
110 PRINT USING"#####";Z(I,J);:NEXT J:PRINT :NEXT I
120 P=1
130 FOR I=P+1 TO R+1
140 IF Z(P,P)<>0 THEN 190
150 FOR J=P TO R
160 K=-Z(P,J):Z(P,J)=Z(I,J):Z(I,J)=K
170 NEXT J:NEXT I
180 C=0:LOCATE 12+R,37+4*R:PRINT C:END
190 C=1:L=1 :P=1
200 FOR K=1 TO R
210 IF Z(K,K)=0 THEN 130
220 FOR N=K+1 TO R
230 T=-Z(N,L)/Z(K,L)
240 FOR M=P TO R
250 Z(N,M)=Z(N,M)+T*Z(K,M)
260 NEXT M:NEXT N:P=P+1
270 IF P>R THEN 290
280 C=C*Z(K,L):
290 L=L+1
300 NEXT K
310 C=C*Z(R,R)
320 PRINT :PRINT "DETERMINANTIN DEGERI = ";C
330 END
```

Örnek III.8.g.

Binom katsayılarını hesaplayacak bilgisayar programını yapınız.

```
10 REM ********** BINOM KATSAYILARININ HESABI ***************
20 CLS
30 INPUT " KACINCI MERTEBEYE KADAR BINOM KATSAYILARININ
HESABINI ISTIYORSUNUZ....! ",N
40 DIM A(N,N+1)
50 FAK=1
60 FOR I=1 TO N
70 FAK=FAK*I
80 K=0
90 FOR J=1 TO I+1
100 L=K:GOSUB 1000:G=FAK1
```

```
110 L=I-K:GOSUB 1000:GS=FAK1
120 A(I,J)=FAK/(G*GS)
130 K=K+1
140 NEXT J:NEXT I
150 FOR I=1 TO N:FOR J=1 TO I+1
160 PRINT USING"#####";A(I,J);
170 NEXT J:PRINT :NEXT I
180 END
1000 REM ********** FAKTORYEL HESABI ****************
1010 FAK1=1
1020 FOR M=1 TO L
1030 FAK1=FAK1*M
1040 NEXT M
1050 RETURN
```

Örnek III.8.h.

N elemanlı bir dizinin elemanlarını küçükten büyüğe doğru sıralayacak bilgisayar programını yapınız.

```
10 REM ** DIZININ ELEMANLARINI SIRALAMA PROGRAMI *****
20 CLS
30 INPUT "ELEMAN SAYISINI GIRINIZ.... : ",N
40 DIM A(N)
50 FOR I=1 TO N
60 INPUT A(I)
70 NEXT I
80 FOR I=1 TO N-1
90 EK=A(I):GS=I
100 FOR J=I+1 TO N
110 IF EK>A(J) THEN GS=J
120 NEXT J
130 C=A(I):A(I)=A(GS):A(GS)=C
140 NEXT I
150 FOR I=1 TO N
160 PRINT A(I)
170 NEXT I
180 END
```

Örnek III.8.ı.

Herhangi 2 matrisin çarpımını bulacak bir bilgisayar

programı yapınız.

```
10 REM **** MATRISLERIN CARPIMI ******
20 CLS
30 INPUT "A MATRISININ BOYUTUNU GIRINIZ (SATIR,SUTUN) : ",N1,M1
40 INPUT "B MATRISININ BOYUTUNU GIRINIZ (SATIR,SUTUN) : ",N2,M2
50 DIM A(N1,M1),B(N2,M2),C(N1,M2)
60 IF M1<>N2 THEN PRINT "MATRISLER CARPILAMAZ.....":END
70 FOR I=1 TO N1:FOR J=1 TO M1
80 INPUT A(I,J)
90 NEXT J:NEXT I
100 FOR I=1 TO N2:FOR J=1 TO M2
110 INPUT B(I,J)
120 NEXT J:NEXT I
130 FOR I=1 TO N1
140 FOR J=1 TO M2
150 T=O
160 FOR K=1 TO N2
170 T=T+A(I,K)*B(K,J)
180 NEXT K
190 C(I,J)=T
200 NEXT J
210 NEXT I
220 FOR I=1 TO N1:FOR J=1 TO M2
230 PRINT USING"####";C(I,J);
240 NEXT J:PRINT :NEXT I
250 END
```

III.9. BASIC PROGRAMLAMA DİLİNDE FONKSİYONLAR

Basic programlama dilinde fonksiyonlar DEF deyimi ile belirtilirler.

Örnek III.9.a.

```
10 DEF FN F(X)=X*X+1
```

yazılımında X^2+1 fonksiyonu tanımlanmıştır. Bu fonksiyonun herhangi bir noktadaki değerini hesaplamak için FN F(nokta) deyimi kullanılır. FN F(0) yazıldığı takdirde fonksiyonun 0'daki değerini verir.

Örnek III.9.b.

X*X-1 fonksiyonunun okutulan herhangi bir nokta civarındaki kökünü bulacak programı yapınız.

```
10 REM *** BELİRLİ BİR NOKTA CİVARINDAKİ KÖK BULMA *****
20 CLS
30 DEF FN F(X)=X*X+1
40 DEF FN TUREV(X)=2*X
50   INPUT "HANGİ NOKTA CİVARINDAKİ KÖKUN BULUNMASINI
İSTİYORSUNUZ..",X0
60 X=X0
70 INPUT "MAKSİMUM İTARASYON SAYISI ... : ",ITAR
80 INPUT "EPSİLON SAYISI....: ",EPS
90 FOR I=1 TO ITAR
100 X1=X0-FN F(X0)/FN TUREV(X0)
110 X0=X1
120 IF FN F(X0)<=EPS THEN 150
130 NEXT I
140  IF FN F(X0)>0 THEN PRINT "FONKSİYONUN BELİRTİLEN NOKTA
CİVARINDA KÖKU YOKTUR":GOTO 170
150 PRINT "ÇÖZÜME ";I;".INCI İTARSAYON SONUCU ULAŞILDI...!"
160  PRINT "FONKSİYONUN ";X;" NOKTASI CİVARINDAKİ GERÇEK
KÖKÜ..: ";X0
170 END
```

III.10. BASIC PROGRAMLAMA DİLİNDE ALTPROGRAMLAR

Basic'de bir alt program GOSUB kontrol komutu ile gerçekleştirilir. GOSUB deyimi ile gidilen herhangi bir alt programdan dönebilmek için RETURN deyimini kullanmak gereklidir. Bir basic programında GOSUB deyimi kullanıldığı zaman GOSUB ile gönderilmek istenen satıra gidip oradan RETURN deyimini görünceye kadar işlem yapar. RETURN deyimini gördüğü anda hangi satırdan bu altprograma geldi ise o satırın bir sonraki deyiminden itibaren işlem yapmaya devam eder.

Örnek III.10.a.

```
10 CLS
15 DIM A(100)
20 FOR I=1 TO 100
```

```
30 INPUT A(I)
40 IF A(I)>=0 THEN GOSUB 1000:GOTO 60
50 GOSUB 2000
60 NEXT I
70 END
1000 PRINT "OKUTULAN SAYI POZİTİF BİR SAYIDIR...."
1010 RETURN
2000 PRINT "OKUTULAN SAYI NEGATİF BİR SAYIDIR...."
2010 RETURN
```

Yukaridaki ornekte okutulan A dizisinin elemani pozitif ise programın akışı 1000 nolu satıra sapacak ve ekrana "OKUTULAN SAYI POZİTİF BİR SAYIDIR...." mesajını verdikten sonra RETURN deyimini gördükten sonra geri dönecek GOTO 60 deyimi ile 60 nolu satıra gidecektir.

Okutulan A dizisinin eleman sayısı negatif ise programın akışı 2000 nolu satıra sapacak ve ekrana "OKUTULAN SAYI NEGATİFTİR...." mesajını verdikten sonra geri dönüp 60 nolu satırdan itibaren işlem yapmaya devam edecektir.

Örnek III.10.b.

```
10 CLS
20 GOSUB 100
30 END
100 PRINT "1.HAZİRAN 1989 TARİHİ PERŞEMBE GUNUNE RASTLAMAKTADIR"
110 RETURN
```

Program çalıştırldığı takdirde 10 nolu satırda ekranı temizleyip 20 nolu satırda GOSUB 100 deyimini gördüğü anda 100 nolu satıra gidip ekrana "1.HAZİRAN 1989 TARİHİ PERŞEMBE GUNUNE RASTLAMAKTADIR."yazacak ve RETURN deyimi ile geri dönüp 30 nolu satırda programı sona erdirecektir.

Basic programlama dilinde yapılmak istenen bir alt programın mutlaka bir RETURN deyimi ile sona ermesi gerekmektedir.

Örnek III.10.c.

Cramer yöntemi ile bir lineer denklem sisteminin çözümünü yapacak bir bilgisayar programı yapınız.

```
10 REM **** CRAMER SISTEMI ILE DENKLEM SISTEMININ COZUMU ****
20 CLS
```

```
30 INPUT "MATRISIN BOYUTUNU GIRINIZ.......: ",R1
40 DIM A(R1,R1),Z(R1+1,R1+1),X(R1),B(R1)
50 PRINT "KATSAYILAR MATRISINI GIRINIZ....!"
60 FOR I=1 TO R1:FOR J=1 TO R1
70 INPUT A(I,J)
80 Z(I,J)=A(I,J)
90 NEXT J:NEXT I
100 PRINT :PRINT "ESITLIGIN SAG TARAFINI GIRINIZ....!"
110 FOR I=1 TO R1
120 INPUT B(I)
130 NEXT I
140 R=R1:GOSUB 310
150  IF C=0 THEN PRINT "BU DENKLEM SISTEMI LINEER  OLMADIGI
ICIN CRAMER ILE COZULEMEZ....! ":END
160 DET=C
170 FOR E=1 TO R1
180 FOR U=1 TO R1:FOR G=1 TO R1
190 IF E=U THEN Z(G,U)=B(G):GOTO 210
200 Z(G,U)=A(G,U)
210 NEXT G:NEXT U
220 R=R1:GOSUB 310
230 X(E)=C/DET
240 NEXT E
250 PRINT "DENKLEM SISTEMININ KOKLERI SIRASI ILE ....! "
260 PRINT
270 FOR I=1 TO R1
280 PRINT "X(";I;")=";X(I)
290 NEXT I
300 END
310 REM ********** DETERMINANT HESABI **********
320 P=1
330 FOR I=P+1 TO R+1
340 IF Z(P,P)<>0 THEN 390
350 FOR J=P TO R
360 K=-Z(P,J):Z(P,J)=Z(I,J):Z(I,J)=K
370 NEXT J:NEXT I
380 C=0:GOTO 520
```

```
390 C=1:L=1 :P=1
400 FOR K=1 TO R
410 IF Z(K,K)=0 THEN 330
420 FOR N=K+1 TO R
430 T=-Z(N,L)/Z(K,L)
440 FOR M=P TO R
450 Z(N,M)=Z(N,M)+T*Z(K,M)
460 NEXT M:NEXT N:P=P+1
470 IF P>R THEN 490
480 C=C*Z(K,L):
490 L=L+1
500 NEXT K
510 C=C*Z(R,R)
520 RETURN
```

III.10.1. Basic Programlama Dilinde Bir Altprogramla İlgili Olabilecek Hatalar Şunlar Olabilir

RETURN without GOSUB in ‹Satır no›

Hata numarası 3'tür. Anlamı ise Satır no ile belirtilen satırda RETURN deyimi görülmüş fakat bu RETURN deyimine GOSUB deyimi ile gelmemiş. (GOSUB kullanılmadan RETURN kullanılmış.)

III.11. BASIC PROGRAMLAMA DİLİNDE SAHALARIN KARAKTER SEVİYESİNDE İNCELENMESİ

III.11.1. MID$ DEYİMİ

Herhangi bir karakter tipindeki bir değişkenin , herhangi bir karakterinden itibaren , istenildiği kadar karakter sayısını , başka bir karakter tipindeki değişkene atamak için kullanılan bir deyimdir.

B$=MID$(A$,I,J) yazılımına göre A$ adlı değişkenin I. karakterinden itibaren J tanesini alıp B$ adlı değişkene atar.

Örnek III.11.a.

```
10 G$="1.HAZIRAN 1989"
20 GS$=MID$(G$,1,9)
30 LOCATE 10,17:PRINT GS$
40 END
```

Programının çalıştırılması sonucu ; G$ adlı değişkenin 1.

karakterinden itibaren 9 karakterini alıp GS$ adlı değişkene atar ve 30 nolu satırda GS$ adlı değişkenin değerini ekranın 10.satır 17.sütununa yazar.

Örnek III.11.b.

```
1
1 H
1 H A
1 H A Z
1 H A Z İ
1 H A Z İ R
1 H A Z İ R A
1 H A Z İ R A N
```

Ekrana yukarıdaki şekilde bir yazı yazacak basic programı yapınız.

```
10 CLS
20 GS$="1 H A Z İ R A N"
30 FOR I=1 TO 15 STEP 2
40 PRINT MID$(GS$,1,I)
50 NEXT I
60 END
```

III.11.2. LEFT$ DEYİMİ

Herhangi bir karakter tipindeki bir değişkenin , en solundaki karakterinden itibaren , istenildiği kadar karakter sayısını , başka bir karakter tipindeki değişkene atamak çin kullanılan bir deyimdir.

B$=LEFT$(A$,J) yazılımına göre A$ adlı değişkenin en solundaki karakterinden itibaren J tanesini alıp B$ adlı değişkene atar.

Örnek III.11.c.

```
10 CLS
20 G$="1 HAZİRAN 1989 PERŞEMBE"
30 GS$=LEFT$(G$,15)
40 LOCATE 10,17:PRINT GS$
50 END
```

Programının çalışması sonucu G$ adlı değişkenin soldan itibaren 15 karakterini alıp GS$ adlı değişkene atayıp ekranın 10.satır 17.sütününa GS$ adlı değişkeni yazar.

III.11.3. RIGHT$ DEYİMİ

Herhangi bir karakter tipindeki bir değişkenin , en sağındaki karakterinden itibaren , istenildiği kadar karakter sayısını , başka bir karakter tipindeki değişkene atamak için kullanılan bir deyimdir.

B$=RIGHT$(A$,J) yazılımına göre A$ adlı değişkenin en sağındaki karakterinden itibaren J tanesini alıp B$ adlı değişkene atar.

Örnek III.11.d.

```
10 CLS
20 G$="1 HAZİRAN 1989 PERŞEMBE"
30 GS$=RIGHT$(G$,8)
40 LOCATE 10,17:PRINT GS$
50 END
```

Programının çalışması sonucu G$ adlı değişkenin sağdan itibaren 8 karakterini alıp GS$ adlı değişkene atayıp ekranın 10.satır 17.sütününa GS$ adlı değişkeni yazar.

III.11.4. LEN DEYİMİ

Herhangi bir karakter tipteki değişkenin karakter sayısını veren deyimdir.

Örnek III.11.e.

```
10 CLS
20 G$="1 HAZİRAN 1989 PERŞEMBE"
30 GS=LEN(G$)
40  LOCATE 10,17:PRINT "G$ Adlı Değişkende Bulunan Karakter
Sayısı: ";GS
50 END
```

Programının çalışması sonucu G$ adlı değişkende bulunan karakter sayısını GS ile belirtilen nümerik bir değişkene atayıp 10.satır 17.sütününa GS adlı değişkeni yazar.

III.11.5. VAL DEYİMİ

Alfanümerik olan bir değişkeni nümerik bir değişkene çevirmek için kullanılan bir deyimdir.

Örnek III.11.f.

```
10 CLS
20 GS$="8247"
30 GS=VAL(GS$)
```

```
40 PRINT "GS=";GS
50 END
```

Programının çalışması sonucu GS$ adlı alfanümerik bir değişkenin değerini nümerik bir değişkene çevirerek GS adlı değişkene atar ve ekrana nümerik olan bu sayıyı yazar.

III.11.6. STR$ DEYİMİ

Nümerik bir değişkeni string (alfanümerik) bir değişkene çevirmek için kullanılan bir deyimdir.

Örnek III.11.g.

```
10 CLS
20 GS=8247
30 GS$=STR$(GS)
40 PRINT "GS$=";GS$
50 END
```

Programının çalışması sonucu GS adlı nümerik bir değişkenin değerini alfanümerik bir değişkene çevirerek GS$ adlı değişkene atar ve ekrana alfanümerik olan bu değeri yazar.

III.11.7. ASC DEYİMİ

Herhangi bir karakterin ASCII kodunu veren bir komuttur.

Örnek III.11.h.

```
10 A=ASC("A")
20 PRINT "A Karakterinin ASCII kodu ";A;" dır."
30 END
```

program parçacığının çalışması sonucu ekranda "A" karakterinin ASCII kodu yazılır.

III.11.8. SPACE$ DEYİMİ

Belirtilen değişkene , belirtilen tamsayı kadar boşluk atar. SPACE$ deyimi sadece PRINT deyimi ilede kullanılabilir.

Örnek III.11.ı.

```
10 PRINT SPACE$(80)
```

programı çalıştırılacak olunur ise , kürsorun bulunduğu konumdan itibaren 80 adet boşluk karakteri yazar.

Örnek III.11.i.

Ekranın 11. ve 17. satırlarına boşluk atmak için : aşağıdaki program çalıştırılmalıdır.

```
10 A$=SPACE$(80)
20 LOCATE 11,1:PRINT A$
30 LOCATE 17,1:PRINT A$
40 END
```

III.11.9. CHR$ DEYİMİ

ASCII kodları ile belirtilen karakterleri , karakter tipi değişkenlere atamak için veya ekrana yazdırmak için kullanılan bir deyimdir.

Örnek III.11.j.

ASCII kodu 65 olan karakteri GS$ adlı değişkene atayabilmek için ;

```
10 GS$=CHR$(65)
```

program parçacığı çalıştırılmalıdır.

IIII.11.10 STRING$ DEYİMİ

Belirtilen karakter tipi değişkene , belirtilen tam sayı kadar belirtilen karakterden yan yana yazar.

Örnek III.11.k.

A$ adlı değişkene "—" karakterinden yan yana 80 tane atayabilmek için aşağıdaki program parçacığı yapılmalıdır.

```
10 CLS
20 A$=STRING$(80,"—")
30 PRINT A$
40 END
```

veya

```
10 CLS
20 A$=STRING$(80,CHR$(196))
30 PRINT A$
40 END
```

veya

```
10 CLS
20 A$=STRING$(80,196)
30 PRINT A$
40 END
```

Bu program parçacıklarından herhangi birisi çalıştırıldığı takdirde , ekranda kürsorun bulunduğu yere 80 tane "—" karakterinden yazar.

III.12. BASIC PROGRAMLAMA DİLİNDE DOSYA TİPLERİ

Basic programlama dilinde iki ayrı tipte dosya açılım şekli vardır. Bunlar SEQUENTIAL (SIRALI ERİŞİMLİ) dosyalar ve RANDOM (RASTGELE ERİŞİMLİ) dosyalardır.

Basic programlama dilinde hangi tipten olur ise olsun aynı anda 3 'den fazla dosya açık bulundurulamaz. Eğer 3'den fazla dosya açık bulundurulmak isteniyor ise hesaplayıcıya BASIC , BASICA veya GWBASIC yüklenir iken aşağıdaki şekilde bir bildiri sunulmalıdır.

A›BASIC /F:‹Aynı anda açık bulundurulacak dosya sayısı›

veya

A›BASICA /F:‹Aynı anda açık bulundurulacak dosya sayısı›

yada

A›GWBASIC /F:‹Aynı anda açık bulundurulacak dosya sayısı›

Burada /F: parametresi ile yapılacak Basic programında aynı anda açık bulundurulmak istenen dosya sayısı verilebilir.

Örnek III.12.a.

Eğer GWBASIC bilgisayara ,

A›GWBASIC /F:5

şeklinde yüklenir ise : Yapılacak olan dosya programında aynı anda 5 adet dosya açık bulundurulabilir ve bu beş dosya üzerinde giriş çıkış işlemleri yapılabilirdi.

Basic programlama dilinde açılmış olan bir dosyanın bir kayıdının maksimum kayıt uzunluğu 256 'dan fazla olamaz. Eğer dosyanın bir kayıdının uzunluğunun 256'dan fazla olması gerekiyor ise , bilgisayara BASIC , BASICA veya GWBASIC yüklenir iken aşağıdaki şekilde bir bildiri sunulmalıdır.

A›BASIC /S:‹Dosyanın 1 kayıdının uzunluğu›

veya

A›BASICA /S:‹Dosyanın bir kayıdının uzunluğu›

yada

A›GWBASIC /F:‹Dosyanın bir kayıdının uzunluğu›

Burada /S:‹Dosyanın bir kayıdının uzunluğu› parametresi ile yapılacak dosya programının bir kayıdının uzunluğu verilebilir. Bir kayıdın uzunluğu 256 'dan az ise BASIC dilleri bilgisayara yüklenir iken bu bildiriyi yazmaya

gerek yoktur. Eğer bir kayıt uzunluğu 256'dan fazla ise Basic dilleri bilgisayara yüklenirken bu bildiri ile yüklenmelidir.

Örnek III.12.b.

Bir kayıdının uzunluğu 317 olan bir dosya açabilmek için Basic dillerinden birisi bilgisayara yüklenir iken ;

A>GWBASIC /S:317

şeklinde bir bildiri sunulmalıdır.

Örnek III.12.c.

Bilgisayara GWBASIC yüklenir iken ;

A>GWBASIC /F:5/S:317

şeklinde bir bildiri sunulur ise açılacak maksimum dosya sayısı 5 ve herhangi bir dosyanın bir kayıdının maksimum uzunluğu 317 olacaktır.

III.12.1. SQUENTIAL (SIRALI ERIŞIMLI) DOSYALAR

III.12.1.1. OPEN KOMUTU

Açılmak istenen dosyayı açmak için kullanılan bir deyimdir. Open kelimesinden sonra tırnak içerisinde , açılacak olan dosyanın tipi ,dosya numarası ve dosyanın disketteki adı belirtilmelidir.

Örnek III.12.1.a.

10 OPEN "O",#1,"B:GS.DAT"

örnekte "O" ile dosyanın açılış modu,

#1 ile dosyanın numarası ,

"B:GS.DAT" ile dosyanın hangi disk veya diskette açılacağı ile dosyanın disk veya disketteki adı belirtilmektedir. Bu örnekte dosya B sürücüsünde bulunan disket üzerinde açılmaktadır ve dosyanın disketteki adı GS.DAT 'dır.

III.12.1.2. CLOSE DEYİMİ

Açılmış olan dosyayı kapatmak için kullanılan bir deyimdir. Açık bulunan dosyalardan sadece bir tanesi kapatılmak istendiği zaman CLOSE deyimi ile beraber kapatılmak istenen dosyanın dosya numarası da belirtilmelidir. Açık bulunan tüm dosyaların kapatılması istenildiği zaman sadece CLOSE yazılması yeterlidir.

Örnek III.12.1.b.

Açık bulunan #1,#2,#3 no'lu dosyalardan sadece #1 no'lu

dosyayı kapatmak için CLOSE #1 veya CLOSE 1 yazmak yeterlidir. Bu şekilde kapatılmış olan bir dosyada açık bulunan #2 ve #3 no'lu dosyalar kapatılmaz. Eğer tüm dosyalar kapatılmak istenildiği zaman sadece CLOSE deyiminin yazılması yeterli olacaktı.

III.12.1.3. SQUENTIAL (SIRALI) Dosyaların Açılım Şekilleri

Basic programlama dilinde SEQUENTIAL dosyaların üç ayrı şekilde açılım modu vardır. Bunlar OUTPUT modu , INPUT modu ve APPEND modu dur.

Output Modu : Bu mod dosyaya bilgi yazmak için kullanılır. Aynı zamanda disk veya diskette , SEQUENTIAL tip dosya ilk defa açılıyor ise mutlaka Output modunda açılmalıdır.

Output modunda açılmış olan bir dosya da daha önce yapılmış olan kayıtların tamamı silinir.

Genel Yazılım Formatı Aşağıdaki Gibidir :

‹Satır No› OPEN "O",‹Dosya No›,‹Dosyanın Disketteki Adı›

OPEN : Açılmak istenen dosyayı disk veya diskette açar.

"O" : Açılacak dosyanın OUTPUT modda açıldığını belirtmek içindir.

‹Dosya No› : Eğer birkaç tane dosya kullanılıyor ise , bu dosyaları birbirinden ayırmak için kullanılır. Dosya numarası 1 ile 15 sayıları arasında herhangi bir tam sayıdır. Kullanılır iken bu dosya numarasının önüne # işareti de getirilebilir.

‹Dosyanın Disketteki Adı › : Açılacak olan dosyanın disk veya disket üzerindeki adını belirtmek için kullanılır.

Örnek III.12.1.c.

B sürücüsünde bulunan disket üzerine adı "ESRA.DAT" olan ve OUTPUT modda açılmış bir dosya açalım.

```
10 OPEN "O",#1,"B:ESRA.DAT"
```

Örnekte "O" ile dosyanın output modda açıldığını , #1 ile dosya numarasının 1 olduğunu , "B:ESRA.DAT" ile B sürücüsünde bulunan disket üzerinde adı ESRA.DAT olan bir dosyanın açıldığı belirtilmektedir.

Input Modu : Bu mod dosyadan bilgi okumak için kullanılır.

Disk veya diskette daha önce Output mod ile açılmış dosyadan bilgi okunmak istendiği zaman kullanılır. Daha önce Output ile açılmamış dosya , Input modunda açılmaya çalışılır ise ekranda aşağıdaki hata mesajını belirten ileti oluşur.

file not found ‹Satır No›

Böyle bir hata ile karşılaşıldığı zaman ERR hata değişkenine 53 değeri yüklenmektedir. Bu hata aranılan dosyanın bulunamadığını belirtmektedir.

Burada Input modunda açılmak istenilen dosyanın bulunamadığı belirtilmektedir.

Genel Yazılım Formatı Aşağıdaki Gibidir.

‹Satır No› OPEN "I",‹Dosya No›,‹Dosyanın Disketteki Adı›

"I" : Açılacak dosyanın INPUT modda açıldığını belirtmek içindir.

Diğer parametreler ise Output modda belirtildiği gibidir.

Örnek III.12.1.d.

Daha önce Output modunda açılmış olan ismi "ESRA.DAT" olan bir dosyayı INPUT modunda açabilmek için ,

```
10 OPEN "I",#1,"ESRA.DAT"
```

şeklinde bir program yapılması yeterlidir.

Örnek III.12.1.e.

Daha önce diskette Output modda açılmamış "COS.DAT" adlı dosyanın Input modunda açılabilmesi için bir program yapalım.

```
10 CLS
20 ON ERROR GOTO 1000
30 OPEN "I",#1,"COS.DAT"
40 END
1000 IF ERR=53 THEN 1100 ELSE 40
1100 LOCATE 10,17: PRINT "DOSYA OUTPUT MODDA AÇILDIKTAN
SONRA INPUT MODDA AÇILACAKTIR....!"
1110 OPEN "O",#1,"COS.DAT":CLOSE #1
1120 GOTO 30
```

Yazılan bu programın 20 no'lu satırında kullanılan ON ERROR GOTO deyimi ile eğer program akışı içerisinde herhangi bir hata meydana gelir ise , programın akışı 1000 no'lu satıra kayacaktır. 1000 no'lu satırda hata numaralarını kontrol edilip gerekli mesajlar verildikten sonra GOTO deyimi ile

programın akışı istenilen satıra yönlendirilebilir.

Append Modu : Append modu SEQUENTIAL tipte bir dosyaya kayıt ilavesinde bulunmak için açılan moddur. A modu ile açılmış olan bir SEQUENTIAL tipte dosyada , yazıcı kafa son kayıttan bir sonraki kayıt üzerine konumlanır. Append modda açılmış olan bir dosyanın daha önce Output modda açılmış olmasına gerek yoktur. Eğer dosya disk veya disket üzerinde ilk defa , A modunda açılıyor ise yazıcı kafa dosyanın birinci kayıdı üzerine konumlanır.

Genel Yazılım Formatı :

‹Satır No› OPEN "A",‹Dosya No›,‹Dosyanın Disketteki Adı›

"A" : Açılacak dosyanın INPUT modda açıldığını belirtmek içindir.

Diğer parametreler ise Output modunda belirtildiği gibidir.

SEQUENTIAL Tip Dosyalarda Erişim Şekli : SEQUENTIAL tip dosyalarda erişim şekli sıralıdır. Yani herhangi bir kayıda ulaşabilmek için , o kayıda ulaşıncaya kadar olan tüm kayıtlar okunmalıdır. Diyelimki dosya içerisinde bulunan 17. kayıdı okumak istiyoruz. Bu kayıdı okuyabilmek için bundan önceki yapılmış olan 16 kayıt okunmalıdır.

SEQUENTIAL Tip Dosyalarda Kayıt Yapma Deyimleri : SEQUENTIAL tip dosyalarda kayıt yapma deyimleri iki şekilde olur. Bunlar sırası ile PRINT ve WRITE deyimleridir. Bunların yazılış şekilleri aşağıda belirtilmiştir.

1. PRINT #‹Dosya No›,‹Kayıt Değişkenleri›

‹Dosya No› : Kayıt yapılacak dosyanın dosya numarası ,

Bu şekilde bir dosyaya bilgi yazılmak istenir ise , dosya değişkenleri birbirinden "," ile ayrılmalıdır.

Örnek III.12.1.f.

```
10 CLS
20 OPEN "O",#1,"ESRA.DAT"
30 LOCATE 10,17:INPUT "ADI ..........: ",A$
40 LOCATE 11,17:INPUT "SOYADI .......: ",B$
50 LOCATE 12,17:INPUT "TELEFON NO ...: ",C$
60 PRINT #1,A$;",";B$;",";C$
70 LOCATE 17,17:INPUT "KAYIT YAPMAYA DEVAM IÇIN (E/e)
YAZ",E$
```

```
80 IF E$="E" OR E$="e" THEN CLS:GOTO 30
90 CLOSE #1
100 END
```

Şeklinde yapılmış olan bir program , istenildiği kadar kişinin adını , soyadını ve telefon numarasını adı ESRA.DAT olan dosyaya yazar. Bu programı sona erdirebilmek için 70 nolu satırda okutulmakta olan E$ adlı karakter tipindeki değişkene E veya e 'den başka bir değer atanmalıdır.

2. WRITE #‹Dosya No›,‹Kayıt Değişkenleri›

Sequential tipte açılmış olan dosyaya bilgi yazdırmak için kullanılan bir deyimdir. Bu şekilde dosyaya bilgi yazdırılmak isteniyor ise dosya değişkenlerini ,

```
WRITE #1,A$,B$
```

şeklinde yan yana yazabiliriz.

Örnek III.12.1.g.

Yukarıdaki örnek WRITE deyimi kullanılarak aşağıdaki şekilde düzenleyebiliriz.

```
10 CLS
20 OPEN "O",#1,"ESRA.DAT"
30 LOCATE 10,17:INPUT "ADI ..........: ",A$
40 LOCATE 11,17:INPUT "SOYADI .......: ",B$
50 LOCATE 12,17:INPUT "TELEFON NO ...: ",C$
60 WRITE #1,A$,B$,C$
70  LOCATE  17,17:INPUT  "KAYIT YAPMAYA  DEVAM  İÇİN  (E/e)
YAZ",E$
80 IF E$="E" OR E$="e" THEN CLS:GOTO 30
90 CLOSE #1
100 END
```

Bu örnekte istenildiği kadar kayıt WRITE deyimi kullanılarak ESRA.DAT adlı dosyaya kayıt edilmektedir.

SEQUENTIAL Tip Dosyalarda Kayıt Okuma Deyimleri :

Sequential tip dosyalarda kayıt okunabilmesi için

```
INPUT #‹Dosya No›
```

deyimi kullanılır. Burada Dosya No ile okunacak dosyanın dosya numarası belirtilmektedir.

Sequential olarak açılmış bir dosyadan okutulacak değişkenlerin adlarının , dosyaya bilgi yazarken kullanılan

değişkenler ile aynı olmasına gerek yoktur. Diyelim ki dosyaya bilgi yazarken kullanılan değişkenler AD$,SOYAD$,TEL$ olsun. Aynı dosyadan bilgi okumak istenildiğinde kullanılacak değişkenlerin isimleri AD$,SOYAD$,TEL$ olmak zorunda değildir. Bunlar yerine A$,B$,C$ gibi farklı değişkenlerde kullanılabilir.
Sequential dosyalarda kayıt okunur iken dosya sonuna ulaşılıp ulaşılmadığını EOF (End Of File) deyimi ile kontrol edebiliriz.

III.12.1.4. EOF DEYİMİ

Sequential dosyalarda dosya sonuna ulaşılıp ulaşılmadığını kontrol etmek için kullanılan bir deyimdir. IF , WHILE kontrol komutları ile birlikte kullanılır.
Genel Yazılım Formatı :

IF EOF(Dosya No) THEN ‹Sapılacak Satır No›

Dosya No : Dosya sonuna ulaşılıp ulaşılmadığı öğrenilmek istenen dosyanın dosya numarasıdır.
Eğer dosya sonuna ulaşıldı ise parogramın akışı THEN deyiminden sonra belirtilecek satır numarasına sapar.
Eğer EOF deyimini WHILE kontrol komutu ile kullanılması

‹Satır No› WHILE NOT EOF(Dosya No)
‹Satır No› ‹Dosya Sonuna Ulaşılmadı ise Yapılacak işlemler›
‹Satır No› WEND

şeklinde olur. Burada ise NOT EOF(Dosya No) deyimi ile , okunmakta olan dosyanın dosya sonuna gelinmedi ise WHILE ile WEND deyimlerinin arasında bulunan satırlar icra görecektir. Dosya sonuna ulaşıldı ise , programın akışı WEND deyiminden sonra yazılmış olan satırlara sapar.
Örnek III.12.1.h.
Örnek III.12.1.ğ. de açılmış olan ESRA.DAT adlı dosyadan, isime göre arama yapabilecekj bir program yapınız.

```
10 CLS
20 LOCATE 10,15:INPUT "ARANILAN KIŞININ ADI....:",AD$
30 LOACTE 11,15:INPUT "ARANILAN KIŞININ SOYADI.:",SOYAD$
40 OPEN "I",#1,"GS.DAT"
50 IF EOF(1) THEN 130
60 INPUT #1,AD1$,SOYAD1$,TEL$
```

```
70 IF AD$=AD1$ AND SOYAD$=SOYAD1$ THEN 90
80 GOTO 50
90 LOCATE 12,15:PRINT"ARANILAN KIŞİNİN TEL.NO'SU...:";TEL$
100 LOCATE 19,15:INPUT"DEVAM ETMEK İSTİYORMUSUNUZ..(E/H)",E$
110 IF E$="E" OR E$="e" THEN CLOSE #1:GOTO 10
120 END
130 LOCATE 15,15:PRINT"ARANILAN KIŞI KAYITLARDA YOKTUR..!"
140 GOTO 100
```

Bu örneğin 50 no'lu satırında kullanılan EOF(1) deyimi ile 1 no'lu dosyanın dosya sonuna gelinip gelinmediğini araştırılmaktadır. Eğer 1 no'lu dosyanın sonuna gelindiği takdirde EOF(1) fonksiyonu TRUE (Doğru) değerini üretecektir ve dosya sonuna gelindiği için belirtilen satıra sapacaktır.

III.12.1.5. SEQUENTIAL DOSYALARDA KAYIT DÜZELTME

Sequential dosyalarda kayıt düzeltmek için önce geçici bir dosya açılır. Düzeltilecek kayıda ulaşıncaya kadar olan tüm kayıtlar asıl dosyadan okunarak geçici dosyaya kaydedilir. Düzeltilmek istenen kayıda ulaşılınca bu kayıt üzerindeki yapılacak değişiklikler yapıldıktan sonra geçici dosyaya kaydedilir. Daha sonra düzeltilecek kayıttan sonraki kayıtlar da asıl dosyadan okunarak geçici dosyaya yazılır. Bundan sonra asıl dosya kapatılarak KILL deyimi ile silinir. Daha sonra geçici dosyanın adı NAME deyimi ile asıl dosyanın adı olarak değiştirilir.

III.12.1.5.1. KILL DEYİMİ

Silinmek istenilen bir dosyayı disketten silmek için kullanılan bir deyimdir.

Örnek III.12.1.1.

KILL"GS.DAT yazılıp RETURN 'a basılır ise GS.DAT adlı dosya disk veya disketten silinir. KILL deyimini bir program içerisinde satır numarası vererek kullanabiliriz. Açık bulunan bir dosya disketten KILL deyimi ile silinmeye çalışılır ise ekranda ,

File already open

şeklinde bir hata bildiren mesaj görülür. Bu hatanın kod numarası 55 'dir. Anlamı belirtilen dosyanın halen açık

olduğunu belirtmekte ve bu dosyanın silinemeyeceğini veya tekrar açılamayacağını belirtir.

III.12.1.5.2. NAME DEYİMİ

İstenilen dosyanın adının değiştirilmesi için kullanılan bir deyimdir.

Örnek III.12.1.i.

NAME "GECICI.DAT" AS "GS.DAT"

örnekte GECICI.DAT olan dosyanın adı GS.DAT şeklinde değiştirilmektedir. Bu değişiklik işinin yapılabilmesi için disk veya diskette GS.DAT adında herhangi bir dosya bulunmamalıdır. Eğer diskette GS.DAT adlı bir dosya mevcut olsa idi ekranda ,

File already exists

şeklinde bir hata mesajı bildiren bir mesaj görülecekti. Bu hata mesajının numarası 58 'dir. Anlamı adı değiştirilmek istenen dosyanın yeni adının diskette daha önceden var olduğunu belirtilmektedir. Örnekte diskette daha önceden GS.DAT adında bir dosya olsa idi , kullanıcı bu hata mesajı ile karşılaşacaktı.

III.12.1.6. SEQUENTIAL DOSYALARDA KAYIT SİLME

Sequential dosyalarda kayıt silmek için önce geçici bir dosya açılır. Silinecek olan kayıda ulaşıncaya kadar olan tüm kayıtlar asıl dosyadan okunarak geçici dosyaya kaydedilir. Silinmek istenen kayıda ulaşılınca bu kayıt geçici dosyaya kayıt edilmez ve silinecek kayıttan sonraki kayıtlar da asıl dosyadan okunarak geçici dosyaya yazılır. Bundan sonra asıl dosya kapatılarak KILL deyimi ile silinir. Daha sonra geçici dosyanın adı NAME deyimi ile asıl dosyanın adı olarak değiştirilir. Böylece dosyadan istenilen kayıt silinmiş olur.

Örnek III.12.1.j.

Kayıt değişkenleri , ADI SOYADI (AD$) , MESLEĞİ (M$) , ADRESİ (ADR$) ve TELRFONU (T$) olan ve disketteki adı "URAL.DAT" olan dosyada , dosya açma , kayıt yapma , kayıt görme işlemleri aşağıdaki program tarafından yapılmaktadır.

```
5 REM ***** Bu Program 1988-1989 Öğretim Yılında          ***
6 REM ***** YILDIZ ÜNİVERSİTESİ MATEMATİK MÜHENDİSLİĞİ ***
7 REM ***** I. Sınıf Öğrencilerinden A.Gülnur AKYOL     ***
8 REM ***** Tarafından Yapılmıştır.                      ***
9 DIM A$(100),B$(100),C$(100),D$(1000)
10 CLS:CLOSE #1
15 KEY OFF
20 LOCATE 10,10:COLOR 0,7:PRINT"     ANA MENU     ":COLOR 7,0
30 LOCATE 11,10:PRINT"1- DOSYA AÇMA"
40 LOCATE 12,10:PRINT"2- KAYIT YAPMA"
50 LOCATE 13,10:PRINT"3- KAYIT GÖRME"
60 LOCATE 14,10:PRINT"4- PROGRAM SONU"
70 LOCATE 16,20: COLOR 9,0:  PRINT  "SEÇİMİNİZ 1,2,3,4,[.]"
:COLOR 7,0
80 LOCATE 16,39:A$=INKEY$:IF A$="" THEN 80
90 IF VAL(A$)<1 OR VAL(A$)>4 THEN 70
100 ON VAL(A$) GOTO 110,500,1000,1999
110 'DOSYA ACMA
120 CLS
130 LOCATE 15,10:COLOR 9,8
135 PRINT "DOSYAYI AÇMAK İSTİYORMUSUNUZ? (E/H)":COLOR 7,0
140 A$=INKEY$:IF A$ ="" THEN 140
150 IF A$="E" OR A$="e" THEN 160 ELSE 10
160 OPEN "O",#1,"URAL.DAT"
170 CLOSE#1:GOTO 10
500 'KAYIT YAPMA
505 CLS
510 OPEN "I",#1, "URAL.DAT"
511 OPEN "O",#2, "GECICI.DAT"
512 IF EOF(1) THEN 520
513 INPUT #1,A$,B$,C$,D$
514 WRITE #2,A$,B$,C$,D$
515 GOTO 512
520 LOCATE 10,10:PRINT"ADI SOYADI    :"
530 LOCATE 11,10:PRINT"MESLEĞI       :"
540 LOCATE 12,10:PRINT"ADRES         :"
550 LOCATE 13,10:PRINT"TELEFON       :"
```

```
560 LOCATE 10,25:LINE INPUT AD$
570 LOCATE 11,25:LINE INPUT M$
580 LOCATE 12,25:LINE INPUT ADR$
590 LOCATE 13,25:LINE INPUT T$
600 WRITE #2,AD$,M$,ADR$,T$
610 LOCATE 23,10:PRINT "KAYIT YAPMAYA DEVAMMI  (E/H)";
620 LOCATE 23,40:LINE INPUT A$
630 IF A$="E" OR A$="e" THEN CLS:GOTO 520
640 CLOSE #1,#2:KILL"URAL.DAT"
645 NAME "GECICI.DAT" AS "URAL.DAT":GOTO 10
1000 'KAYIT GORME
1010 CLS
1020 LOCATE 10,10:COLOR 0,7:PRINT"  KAYIT GÖRME MENUSU   "
:COLOR 7,0
1030 LOCATE 11,10:PRINT"1- MESLEĞİNE GÖRE              "
1040 LOCATE 12,10:PRINT"2- ADI SOYADINA GÖRE           "
1050 LOCATE 13,10:PRINT"3- SEMTİNE GÖRE                "
1060 LOCATE 14,10:PRINT"4- ANA MENUYE DÖNUŞ            "
1070 LOCATE 16,20 :COLOR 31,0:PRINT"SEÇİMİNİZ 1,2,3,4, [.]"
:COLOR 7,0
1080 LOCATE 16,40 :A$=INKEY$:IF A$="" THEN 1080
1090 IF VAL(A$)<1 OR VAL(A$)>4 THEN 1070
1100 ON VAL(A$) GOTO 1200,1500,1700,10
1200 'MESLEĞİNE GÖRE
1205 CLS
1210 GOSUB 5000
1220 LOCATE 10,10:COLOR 0,7:INPUT "HANGİ MESLEK LİSTESİNİ
İSTİYORSUNUZ?",F$:COLOR 7,0:CLS
1230 IF EOF(1) THEN 1280
1240 INPUT #2,A$,B$,C$,D$
1250 IF F$=B$ THEN 1260 ELSE 1230
1260 I=I+1:A$(I)=A$:C$(I)=C$:D$(I)=D$
1270 GOTO 1230
1280 IF I=0 THEN 1290 ELSE 1300
1290 LOCATE 21,10:COLOR 0,7:PRINT"BU MESLEKTE KAYITLI KİŞİ
YOKTUR": FOR I1=1 TO 10000:NEXT I1:COLOR 7,0:LOCATE 21,10:
PRINT SPACE$(70):GOTO 1020
```

```
1300 CLS:GOSUB 6000
1310 FOR I=8 TO 21
1320 LOCATE I,6:PRINT A$(I-7)
1330 LOCATE I,30:PRINT C$(I-7)
1340 LOCATE I,50:PRINT D$(I-7)
1350 NEXT I
1360 LOCATE 23,10:PRINT "DEVAM İÇİN BİR TUŞA BASIN"
1370 A$=INKEY$:IF A$="" THEN 1370
1380 CLOSE #1:GOTO 1000
1500 'ADINA GÖRE
1510 CLS
1520 GOSUB 5000
1530 LOCATE 10,10:COLOR 0,7
1535 INPUT "HANGİ ŞAHSI İSTİYORSUNUZ";L$:COLOR 7,0
1537 LOCATE 10,10:PRINT SPACE$(70)
1540 IF EOF(1) THEN 1605
1560 INPUT#1,A$,B$,C$,D$
1570 LOCATE 2,20:PRINT "ADI SOYADI ";:COLOR 16,3:PRINT L$
:COLOR 7,0
1575 IF L$=A$ THEN 1576 ELSE 1540
1576 E=1:BAS=4:SON=22:GOSUB 10000
1580 LOCATE 10,10:PRINT"MESLEGİ.....: ";B$
1590 LOCATE 11,10:PRINT"SEMTİ.....: ";C$
1600 LOCATE 12,10:PRINT"TELEFON.....: ";D$
1605 IF E=0 THEN LOCATE 21,10:PRINT"BU İSİMDE BİRİ  KAYITLI
DEGİLDİR"
1610 LOCATE 22,10:PRINT"DEVAM İÇİN BİR TUŞA BASIN"
1620 A$=INKEY$:IF A$="" THEN 1620
1630 GOTO 1000
1700 'SEMTİNE GORE
1710 BAS=4:SON=22:GOSUB 10000
1720 GOSUB 5000
1730  LOCATE 10,10:COLOR 0,7:INPUT"HANGİ SEMTTE  OTURANLARI
İSTİYORSUNUZ";M$:COLOR 7,0
1735 IF EOF(1) THEN 1780
1740 INPUT #1,A$,B$,C$,D$
1750 IF M$=C$ THEN 1760 ELSE 1735
```

```
1760 S=S+1:A$(S)=A$:B$(S)=B$:D$(S)=D$
1770 GOTO 1735
1780 IF S=0 THEN 1790 ELSE 1800
1790 LOCATE 11,10:COLOR 0,7:PRINT "BU SEMTTE OTURAN KAYITLI
KIŞI YOKTUR":COLOR 7,0:GOTO 1000
1800 CLS:GOSUB 6000
1805 LOCATE 6,30:PRINT "MESLEĞI"
1810 FOR I2=8 TO 21
1820 LOCATE I2,6:PRINT A$(I2-7)
1830 LOCATE I2,30:PRINT B$(I2-7)
1840 LOCATE I2,50:PRINT D$(I2-7)
1850 NEXT I2
1860 LOCATE 23,10:PRINT "DEVAM IÇIN BIR TUŞA BASINIZ"
1870 A$=INKEY$:IF A$="" THEN 1870
1880 GOTO 1000
1999 CLS
2000 LOCATE 10,30:PRINT "┌──────────────────────┐"
2010 LOCATE 11,30:PRINT "│ PROGRAM SONU         │"
2020 LOCATE 12,30:PRINT "└──────────────────────┘"
2030 END
5000 'OKUMAK IÇIN DOSYANIN AÇILMASI
5010 CLOSE:OPEN "I",#1,"URAL.DAT"
5020 RETURN
6000 LOCATE 5,5:PRINT"┌──────────────────────┬──────────────────────
──────┬──────────────────────┐"
6010 LOCATE 6,5:PRINT"│   ADI SOYADI         │   ADRES
   │   TELEFON            │"
6020 LOCATE 7,5:PRINT"├──────────────────────┼──────────────────────
──────┼──────────────────────┤"
6030 FOR I=8 TO 21
6040 LOCATE I,5:PRINT"│                      │
   │                      │"
6050 NEXT I
6060 LOCATE 22,5:PRINT"└──────────────────────┴──────────────────────
──────┴──────────────────────┘"
6070 RETURN
10000 FOR I=BAS TO SON:LOCATE I,1:PRINT SPACE$(80):NEXT :RETURN
```

III.12.2. RANDOM (RASTGELE) ERİŞİMLİ DOSYALAR

Random tipi dosyaların açılım modu R 'dir.

Genel Yazılım Formatı :

‹Satır No›OPEN "R",‹Dosya No›,‹Dosya Adı›,‹Bir Kayıt Uzunluğu›

"R" : dosyanın açılım modunun RANDOM (Rastgele) olduğu belirtilmektedir.

‹Dosya No›: Dosya numarasını belirtmek için kullanılır. Genelde dosya numarasının önüne # işareti konulmaktadır.

‹Dosya Adı›: Dosyanın disketteki adını belirtmek için kullanılır.

‹Bir Kayıt Uzunluğu› : Bunun ile dosyanın bir kayıdının toplam uzunluğu bildirilir.

Dosyanın bir kayıdının toplam uzunluğu , dosya açılırken belirtilen sayıdan büyük olabilir fakat küçük olamaz. Eğer bir kayıdın uzunluğu belirtilen sayıdan küçük olur ise ekranda ,

FIELD overflow in ‹Satır No›

şeklinde hata bildiren mesaj görülür. Bu hatanın kod numarası 20 'dir. Bu hatanın anlamı , Satır No ile belirtilen satırda yeterli alan tanımlaması yapılmamıştır. Başka bir deyişle dosya açılırken belirtilen kayıt uzunluğu , dosya değişkenlerinin toplam uzunluğundan küçüktür.

Dosya OPEN deyimi ile yukarıda belirtilen şekilde açıldıktan sonra FIELD komutu ile dosya değişkenleri tanımlanmalıdır.

III.12.2.1. FIELD KOMUTU

Dosya değişkenleri uzunlukları ile FIELD komutu ile belirtilmektedir.

Genel Yazılım Formatı :

‹Satır No›FIELD ‹Dosya No›,‹Dosya değişkenleri ve uzunlukları›

yukarıda belirtilen dosya değişkenleri ve uzunlukları ile dosyaya yazılacak değişkenler uzunlukları ile belirtilir.

Örnek III.12.2.a.

```
10 CLS
20 OPEN "R",#1,"GS.DAT",47
30 FIELD #1,11 AS A$,17 AS B$,19 AS C$
```

yazılımıda 10 no'lu satırdaki OPEN "R",#1,"GS.DAT",47 ile açılacak olan dosyanın RANDOM tipi olduğunu , "GS.DAT" ile dosyanın disketteki adı ve 47 ile bir kayıdın uzunluğu belirtilmektedir.

20 no'lu satırdaki FIELD deyimi ile dosya değişkenleri ve her değişkenin uzunluğu belirtilmektedir. Ele alınan örnekte A$,B$,C$ adlı değişkenlerin uzunlukları ,

11 AS A$ ile A$ adlı dosya değişkeninin uzunluğu 11 ,

17 AS B$ ile B$ adlı dosya değişkeninin uzunluğu 17 ,

19 AS C$ ile C$ adlı dosya değişkeninin uzunluğu 19 ,

olarak belirtilmektedir.

III.12.2.2. DOSYAYA BİLGİ KAYDETME

Dosyaya bilgi kaydetmek istenildiği zaman , dosyaya yazılacak dosya değişkenleri LSET veya RSET komutları ile dosyaya yerleştirilerek PUT deyimi ile dosyaya yazılırlar.

III.12.2.2.1. LSET DEYİMİ

Dosyaya yazılmak istenen değişkenler , sola yanaşık olarak yazılması isteniyor ise LSET deyimi kullanılır.

Örnek III.12.2.b.

A$ adlı dosya değişkeninin uzunluğu 15 olsun. Dosyaya "MATEMATİK" kelimesi yazılmak istenir ise eğer bu kelime dosyaya ,

```
LSET A$="MATEMATİK"
```

şeklinde yerleştirilir ise , yerleştirme işlemi aşağıdaki şekilde yapılır.

M	A	T	E	M	A	T	İ	K						

III.12.2.2.2. RSET DEYİMİ

Dosyaya yazılmak istenen değişkenler , sağa yanaşık olarak yazılması isteniyor ise RSET deyimi kullanılır.

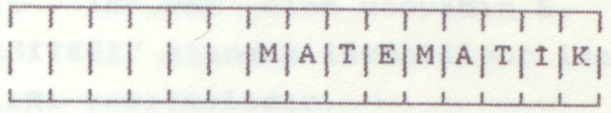
Örnek III.12.2.c.

A$ adlı dosya değişkeninin uzunluğu 15 olsun. Dosyaya

"MATEMATIK" kelimesi yazılmak istenir ise eğer bu kelime dosyaya ,

RSET A$="MATEMATIK"

şeklinde yerleştirilir ise , yerleştirme işlemi aşağıdaki şekilde yapılır.



Örnekte görüldüğü gibi yerleştirme işlemi dosya değişkenine ayrılan alanın en sağından itibaren gerçekleşir.

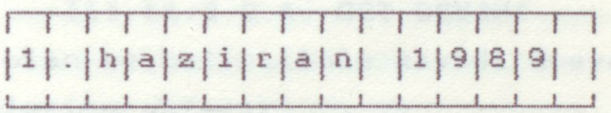
Örnek III.12.2.d.

Dosya değişkeninin adı GS$, uzunluğu 15 dosyaya yazılacak değişkenin değeri "1.haziran 1989 Perşembe" olsun.

Yerleştirme işlemi ;

RSET GS$="1.haziran 1989 Perşembe"

şeklinde yapılacak olur ise dosyanın içerisindeki görünüm aşağıdaki gibi olur.



Örnekte görüldüğü gibi eğer dosya değişkenine ayrılan alan , dosyaya yazılacak olan alanın uzunluğundan daha küçük ise yerleştirme işlemi yine sağdan itibaren yapılırsa ; fazla olan kısım , dosyaya yazılacak olan değişkenin sağından itibaren kesilir. Ele alınan örnekte dosya değişkenine ayrılan alan 15 ve dosyaya yazılacak olan değişkenin uzunluğu 23 olduğundan ve dosyaya yerleştirme işlemi RSET ile yapıldığından bu değişkenin soldan itibaren 15 karakteri dosyaya yazılır.

III.12.2.2.3. PUT DEYİMİ

Dosya değişkenlerine LSET veya RSET deyimleri ile yerleştirilen bilgileri , dosyanın belirtilen kaydına yazmak için kullnılır.

Genel Yazılım Formatı :

‹Satır No› PUT ‹Dosya No›,‹Kayıt No›

‹Kayıt No› ile belirtilen değişken tamsayı değişken

olmalıdır.
‹Dosya No› ile temsil edilen dosyanın , ‹Kayıt No› ile belirtilmiş olan kaydına dosya değişkenlerinin o anki değerlerini yazar.
Örnek III.12.2.d.
Disketteki adı "IBR.DAT" olan dosyanın 5. kayıdı olarak "YILDIZ ÜNİVERSİTESİ" alınmak isteniliyor ise , aşağıdaki program parçacığı yapılmalıdır.

```
10 OPEN "R",#1,"IBR.DAT",19
20 FIELD #1,19 AS A$
30 LSET A$="YILDIZ ÜNİVERSİTESİ"
40 PUT #1,5
50 END
```

yapılan bu programın 40 no'lu satırında #1 ile temsil edilen "IBR.DAT" adlı dosyanın 5. kayıdı olarak "YILDIZ ÜNİVERSİTESİ" alınmıştır. Eğer yapılmak istenen kayıt 5. değilde 17. kayıt olarak yapılmak istenilse idi ; 40 no'lu satırda PUT #1,17 yazılmalı idi.

III.12.2.2.4. GET DEYİMİ

Dosyada var olan ve belirtilmiş kayıdı dosyadan okuyarak dosya değişkenlerine aktarır.
Genel Yazılım Formatı :

‹Satır No› GET ‹Dosya No›,‹Kayıt No›

‹Kayıt No› ile belirtilen değişken tamsayı değişken olmalıdır.
‹Dosya No› ile temsil edilen dosyanın , ‹Kayıt No› ile belirtilmiş olan kayıdını dosyadan okuyarak dosya değişkenlerine yazar.
Örnek III.12.2.e.
Disketteki adı "IBR.DAT" olan dosyanın 5. kayıdı okuyup dosya değişkenlerine aktarılmak istenir ise , aşağıdaki program parçacığı yapılmalıdır.

```
10 OPEN "R",#1,"IBR.DAT",19
20 FIELD #1,19 AS A$
30 GET #1,5
40 PRINT A$
50 END
```

yapılan bu programın 30 no'lu satırında #1 ile temsil edilen "IBR.DAT" adlı dosyanın 5. kayıdı okunarak dosya değişkeni olan A$ adlı değişkene aktararak , 40 no'lu satırda dosyanın 5. kayıdı ekrana yazılır. Eğer okunmak istenen kayıt 5. değilde 17. kayıt olsa idi ; 30 no'lu satırda GET #1,17 yazılmalı idi.

III.12.2.2.5. LOF DEYİMİ

Dosyada bulunan kayıtların toplam byte miktarını bir tamsayı olarak verir. Eğer bu sayı dosyanın bir kayıdının uzunluğuna bölünür ise , dosyada bulunan kayıt sayısı bulunur.

Örnek III.12.2.f.

```
10 OPEN "R",#1,"IBR.DAT",19
20 FIELD #1,19 AS A$
30 K=LOF(1)
40 L=K/19
50 END
```

Program parçacığında 30 no'lu satırda LOF(1) ile 1 no'lu dosyanın kapladığı byte miktarı K adlı tamsayı değişkene aktarılmakta , 40 no'lu satırda ise , K sayısı dosyanın bir kayıt uzunluğu olan 19'a bölünerek dosyadaki toplam kayıt sayısı L değişkenine aktarılmaktadır.

Örnek III.12.2.g.

Disketteki adı "ERZ.DAT" olan random (rastgele) erişimli olarak düzenlenmiş olan dosyalama programının , dosya değişkenleri AD$ (15) , SAD$ (15) , SEMT$ (10) , TEL$ (7) dir. Bu şekilde düzenlenmiş olan programın kayıt yapma , kayıt görme , kayıt düzeltme , kayıt silme bölümleri aşağıdaki programda yapılmıştır.

```
10 OPEN "R",#1,"ERZ.DAT",47
20 FIELD #1,15 AS AD$,15 AS SAD$,10 AS SEMT$,7 AS TEL$
30 K=LOF(1)/47
40 CLS
50 LOCATE 10,20:PRINT "****** A N A   M E N U ******"
60 LOCATE 11,20:PRINT "<1>... KAYIT YAPMA ........."
70 LOCATE 12,20:PRINT "<2>... KAYIT GÖRME ........."
80 LOCATE 13,20:PRINT "<3>... KAYIT DUZELTME ......."
90 LOCATE 14,20:PRINT "<4>... KAYIT SILME ........."
```

```
100 LOCATE 15,20:PRINT "<5>... PROGRAM SONU ....."
110 LOCATE 17,25:PRINT "Seçiminiz <1>.<2>.<3>.<4>.<5>....[.]"
120 LOCATE 17,59:COLOR 17,1:PRINT ".";:COLOR 7,0
130 A$=INKEY$:IF A$="" THEN 130
140 ON VAL(A$) GOTO 150,250,500,750,990
150 'KAYIT YAPMA BÖLUMU
155 CLS
160 GOSUB 1000
170 GOSUB 2000
175 GOSUB 2500
180 K=K+1:PUT #1,K
185 COLOR 0,7
190 LOCATE 20,17:PRINT "KAYIT YAPMAYA DEVAM MI ? (E/H)"
195 COLOR 7,0
200 A$=INPUT$(1)
210 IF A$="E" OR A$="e" THEN 160 ELSE 40
250 'KAYIT GÖRME BÖLUMU
255 CLS
260 COLOR 0,7
270 LOCATE 20,17:PRINT"Görüntülenecek Kayıt No'yu Giriniz..:";
280 COLOR 7,0
290 LINE INPUT K$
300 IF VAL(K$)<1 OR VAL(K$)>K THEN GOSUB 3000:GOTO 250
310 GET #1,VAL(K$)
320 GOSUB 1000
330 GOSUB 4000
340 COLOR 0,7
350 LOCATE 20,17:PRINT"Baska Kayıt Görüntülenecek mi? (E/H)";
360 COLOR 7,0
370 A$=INPUT$(1)
380 IF A$="E" OR A$="e" THEN 250 ELSE 40
500 'KAYIT DUZELTME BÖLUMU
510 CLS
520 COLOR 0,7
530 LOCATE 20,17:PRINT"Düzeltilecek Kayıt No'yu Giriniz..:";
540 COLOR 7,0
550 LINE INPUT K$
```

```
560 IF VAL(K$)<1 OR VAL(K$)>K THEN GOSUB 3000:GOTO 500
570 GET #1,VAL(K$)
580 GOSUB 1000
590 GOSUB 4000
595 COLOR 0,7
600 LOCATE 20,17:PRINT "Düzeltilecek Kayıt Bu mu (E/H)    ";
610 COLOR 7,0
620 A$=INPUT$(1)
630 IF A$="E" OR A$="e" THEN 635 ELSE 650
635 GOSUB 1000:GOSUB 2000:GOSUB 2500
637 PUT #1,VAL(K$)
650 COLOR 0,7
660 LOCATE 20,17:PRINT "Baska Kayıt Düzeltilecek mi? (E/H)";
670 COLOR 7,0
680 A$=INPUT$(1)
690 IF A$="E" OR A$="e" THEN 250 ELSE 40
750 'KAYIT SİLME BÖLÜMÜ
760 CLS
770 COLOR 0,7
780 LOCATE 20,17:PRINT "Silinecek Kayıt No'yu Giriniz..: ";
790 COLOR 7,0
800 LINE INPUT K$
810 IF VAL(K$)<1 OR VAL(K$)>K THEN GOSUB 3000:GOTO 750
820 GET #1,VAL(K$)
830 GOSUB 1000
840 GOSUB 4000
855 COLOR 0,7
860 LOCATE 20,17:PRINT "Silinecek Kayıt Bu mu (E/H)      ";
870 COLOR 7,0
880 A$=INPUT$(1)
890 IF A$="E" OR A$="e" THEN 895 ELSE 920
895 GOSUB 1035
900 LSET AD$=" ":LSET SAD$=" ":LSET SEMT$=" ":LSET TEL$=" "
905 LSET AD$=" ":LSET SAD$=" ":LSET SEMT$=" ":LSET TEL$=" "
910 PUT #1,VAL(K$)
920 COLOR 0,7
930 LOCATE 20,17:PRINT "Baska Kayıt Silinecek  mi? (E/H) ";
```

```
940 COLOR 7,0
950 A$=INPUT$(1)
960 IF A$="E" OR A$="e" THEN 750 ELSE 40
990 'PROGRAM SONU
995 END
1000 'EKRANA GÖRÜNTÜ
1005 LOCATE 10,17:PRINT "ADI ..........: "
1010 LOCATE 12,17:PRINT "SOYADI ........: "
1020 LOCATE 14,17:PRINT "SEMT ..........: "
1030 LOCATE 16,17:PRINT "TELEFON .......: "
1035 COLOR 9,1
1040 LOCATE 10,34:PRINT SPACE$(15)
1050 LOCATE 12,34:PRINT SPACE$(15)
1060 LOCATE 14,34:PRINT SPACE$(10)
1070 LOCATE 16,34:PRINT SPACE$(7)
1080 COLOR 7,0
1090 RETURN
2000 'EKRANDAN BİLGİ OKUMA
2005 COLOR 9,1
2010 LOCATE 10,34:LINE INPUT A$
2020 LOCATE 12,34:LINE INPUT B$
2030 LOCATE 14,34:LINE INPUT C$
2040 LOCATE 16,34:LINE INPUT D$
2050 COLOR 7,0
2060 RETURN
2500 'DOSYA DEĞİŞKENLERİNİ YERLEŞTİRME
2510 LSET AD$=A$:LSET SAD$=B$:LSET SEMT$=C$:LSET TEL$=D$
2520 RETURN
3000 'KONTROL MESAJI
3005 COLOR 0,7
3010 FOR I=1 TO 2500:NEXT I
3015 COLOR 7,0
3020 RETURN
4000 'EKRANA BİLGİ YAZMA
4005 COLOR 9,1
4010 LOCATE 10,34:PRINT AD$
4020 LOCATE 12,34:PRINT SAD$
```

```
4030 LOCATE 14,34:PRINT SEMT$
4040 LOCATE 16,34:PRINT TEL$
4050 COLOR 7,0
4060 RETURN
```

Örnek III.12.2.h.

Aşağıda verilmiş olan örnek program Yıldız Üniversitesi Matematik Mühendisliği ile ilgili iç hat , dış hat ve şehirlerarası telefon numaralarını ilgili dosyalarda saklayan bir programdır.

```
1 REM ************ YILDIZ ÜNİVERSİTESİ ********************
2 REM ************ MATEMATİK MÜHENDİSLİĞİ ******************
3 REM ******** ELEKTRONİK HESAP I-2 DÖNEM ÖDEVİ *************
4 REM ***************** ÖDEVİ HAZIRLAYANLAR ******************
5 REM ***************** 8861017 A.Gülnur AKYOL ***************
6 REM ***************** 8861002 Nilgün GÜLER   ***************
7 REM ***** Yöneten * Arş.Gör. İBRAHİM EMİROĞLU ************
8 REM ***** Konu     * TELEFON REHBERİ ***********************
10 KEY OFF:CLS:DIM G$(15,4),KIM$(300,4),DURUM(300)
15 W$(0)="OKEY ":W$(1)="iPTAL"
20  FOR  I=1  TO 10:KEY I,CHR$(223+I):NEXT  I:KEY  9,"COLOR
7,0"+CHR$(13)
30 KEY 10,"CLS:RUN"+CHR$(13)
40 CLS
50 GOSUB 3000
60 LOCATE 8,25:PRINT "***  ";:COLOR 19,8:PRINT "A N A   M E
N U  ";:COLOR 7,0:PRINT "***";
70 LOCATE 10,28:PRINT "[1]..... TELEFON KAYIT"
80 LOCATE 12,28:PRINT "[2]..... LISTELEME"
90 LOCATE 14,28:PRINT "[3]..... KAYIT DUZELTME"
100 LOCATE 16,28:PRINT "[4]..... PROGRAM SONU"
110 LOCATE 19,37:PRINT "SECIMINIZ....[   ]"
120 LOCATE 19,52:COLOR 19,8 :PRINT".":COLOR 7,0
130 LOCATE 19,52:LINE INPUT A$
140 IF VAL(A$)>4 OR VAL(A$)<1 THEN 120
200 ON VAL(A$) GOTO 205,2600,500,10500
205 GOSUB 3200'EKRAN SIL
210 GOSUB 3500'EKRAN CIZ
```

```
215 GOSUB 3630:GOSUB 3750
220 GOSUB 4000
225 I1=7:L=7
230 GOSUB 4100
235 IF I1>9 THEN A=2 ELSE A=1
240 SC$="050805":IN$=MID$(STR$(I1),A,2)+"1001"
245 ZZ$=G$(I1-6,1):GOSUB 9000:G$(I1-6,1)=ZP$
250 IF FF>0 AND FF<4  THEN 275
255 IF VAL(ZP$)<1 OR VAL(ZP$)>3 THEN G$(I1-6,1)="":GOTO 240
260 IF ZC=-1 AND I1=7 THEN 240
265 IF ZC=-1 THEN I1=I1-1:GOTO 360
270 IF ZC=-2 THEN I1=7:GOTO 235
275 IF ZC=2 THEN I1=L:GOTO 235
280 IF FF>3 THEN 240
285 ON FF GOTO 1000,1100,2400
290 SC$="051412":IN$=MID$(STR$(I1),A,2)+"1412"
295 ZZ$=G$(I1-6,2):GOSUB 9000:G$(I1-6,2)=ZP$
300 IF ZC=-1 THEN  240
305 IF FF>3 THEN 290
310 ON FF GOTO 1000,1100,2400
315 IF ZC=-2 THEN I1=7:GOTO 235
320 IF ZC=2 THEN I1=L:GOTO 235
325 SC$="052721":IN$=MID$(STR$(I1),A,2)+"2721"
330 ZZ$=G$(I1-6,3):GOSUB 9000:G$(I1-6,3)=ZP$
335 IF ZC=-1 THEN  290
340 IF ZC=-2 THEN I1=7:GOTO 235
345 IF ZC=2 THEN I1=L:GOTO 235
350 IF FF>3 THEN 330
355 ON FF GOTO 1000,1100,2400
360 IF I1>9 THEN A=2 ELSE A=1
365 SC$="054912":IN$=MID$(STR$(I1),A,2)+"4912"
370 ZZ$=G$(I1-6,4):GOSUB 9000:G$(I1-6,4)=ZP$
375 IF ZC=-1 THEN  325
380 IF ZC=-2 THEN I1=7:GOTO 235
385 IF ZC=2 THEN I1=L:GOTO 235
390 IF FF>3 THEN 360
395 ON FF GOTO 1000,1100,2400
```

```
400 IF L=I1 THEN L=L+1
405 I1=I1+1
410 IF I1>21 THEN GOSUB 5000 :GOSUB 3800:I1=7:L=21
415 GOTO 235
500 REM***KAYIT DUZELTME***
510 GOSUB 3200
520  LOCATE 8,25:PRINT "***  ";:COLOR 19,8:PRINT "KAYIT
DUZELTME MENUSU";:COLOR 7,0:PRINT "  ***"
530 GOSUB 6000
565 GOSUB 3150
570 LOCATE 15,37:PRINT "SECIMINIZ....[   ]"
575 LOCATE 15,52:COLOR 19,8:PRINT ".":COLOR 7,0
576 LOCATE 15,52:LINE INPUT A$
577 IF VAL(A$)>4 OR VAL(A$)<1 THEN 500
578 IF VAL(A$)=4 THEN 1000
579 FOR S=1 TO 300:KIM$(S,1)="":KIM$(S,2)="" :KIM$(S,3)=""
:KIM$(S,4)="":NEXT S
600 J=VAL(A$)
603 LOCATE 24,8:PRINT SPACE$(70);
604 LOCATE  24,12: COLOR  19,8 :PRINT  "BILGILER   YUKLENIP
SIRALACAK LUTFEN BEKLEYINIZ!";:COLOR 7,0
605 GOSUB 7000'BILGI YUKLEME
606 GOSUB 7200 'SIRALAMA
610 GOSUB 3500'EKRAN CIZ
615 GOSUB 8000
620 SAYFA=1:N=7:L=7
630 GOSUB 4000
635 N=(SAYFA-1)*15+7
640 GOSUB 4100:GOSUB 4200
650 I1=7
660 IF I1>9 THEN A=2 ELSE A=1
670 SC$="051412":IN$=MID$(STR$(I1),A,2)+"1412"
680 ZZ$=KIM$(N-6,2):GOSUB 9000:KIM$(N-6,2)=ZP$
690 IF ZC=-1 AND I1=7 AND SAYFA=1 THEN 670
691 IF ZC=-1 AND I1=7 THEN SAYFA=SAYFA-1:I1=21:GOSUB 4100:
GOSUB 4200:N=N-1:GOTO 660
692 IF ZC=-1 THEN I1=I1-1:N=N-1:GOTO 755
```

```
695 IF FF>3 THEN 670
700 ON FF GOTO 1000,1200,500
705 IF ZC=-2 AND SAYFA=1  THEN I1=7:GOTO 660
707 IF ZC=-2 THEN SAYFA=SAYFA-1:GOTO 635
710 IF ZC=2 THEN SAYFA=SAYFA+1:GOTO 820
715 SC$="052721":IN$=MID$(STR$(I1),A,2)+"2721"
720 ZZ$=KIM$(N-6,3):GOSUB 9000:KIM$(N-6,3)=ZP$
730 IF ZC=-1 THEN  660
735 IF ZC=-2 AND SAYFA=1  THEN I1=7:GOTO 660
737 IF ZC=-2 THEN SAYFA=SAYFA-1:GOTO 635
740 IF ZC=2 THEN SAYFA=SAYFA+1:GOTO 820
745 IF FF>3 THEN 760
750 ON FF GOTO 1000,1200,500
755 IF I1>9 THEN A=2 ELSE A=1
760 SC$="054912":IN$=MID$(STR$(I1),A,2)+"4912"
765 ZZ$=KIM$(N-6,4):GOSUB 9000:KIM$(N-6,4)=ZP$
770 IF ZC=-1 THEN 715
775 IF ZC=-2 AND SAYFA=1  THEN I1=7:GOTO 660
777 IF ZC=-2 THEN SAYFA=SAYFA-1:GOTO 635
780 IF ZC=2 THEN SAYFA=SAYFA+1:GOTO 820
785 IF FF>3 THEN 360
790 ON FF GOTO 1000,1200,500
795 IF L=N THEN L=L+1
798 IF N=SAYI+6  THEN 805
800 I1=I1+1:N=N+1
805 IF I1>21 THEN SAYFA=SAYFA+1:GOTO 820
810 GOTO 660
820 N=(SAYFA-1)*15+7
825 IF N>SAYI+6 THEN N=SAYI+6:I1=SAYI-(SAYFA-2)*15+6
:SAYFA=SAYFA-1:GOTO 660
830 GOTO 635
1000 '*****ANA MENUYE DONUS*****
1010 CLS:GOTO 40
1100 'KAYIT YAPMA
1105 GOSUB 8500
1110 FOR I2=1 TO L-6
1120 HAT$="EMRE"+G$(I2,1)+".DAT"
```

```
1125 IF VAL(G$(I2,1))<1 OR VAL(G$(I2,1))>3 THEN 1180
1130 OPEN "R",#1,HAT$,46
1135 FIELD #1,1 AS HATK$,12 AS UN$,21 AS AD$, 12 AS TEL$
1140 LSET HATK$=G$(I2,1):LSET UN$=G$(I2,2)
1150 LSET AD$=G$(I2,3):LSET TEL$=G$(I2,4)
1155 J=LOF(1)/46+1
1160 PUT#1,J
1170 CLOSE#1
1180 NEXT I2
1185 GOSUB 2500
1190 GOTO 210
1200 'YENI KAYIT YAPMA
1205 GOSUB 8500
1210 HAT$="EMRE"+MID$(STR$(J),2,1)+".DAT"
1215 FOR GUL=1 TO SAYI:DURUM(GUL)=0:NEXT GUL
1220 GOSUB 3850
1240 GOTO  620
1500 END
2400 GOSUB 2500
2410 GOTO 210
2500 'KAYIT IPTAL
2510 FOR S=1 TO 15:G$(S,1)="":G$(S,2)="":G$(S,3)="":G$(S,4)1=""
NEXT S
2520 RETURN
2600 REM***LISTELEME***
2605 GOSUB 3200
2610 GOSUB 6000
2611 GOSUB 3150
2615 LOCATE 8,28:PRINT "***  ";:COLOR 19,8:PRINT "LISTELEME
MENUSU";:COLOR 7,0:PRINT "  ***"
2616 LOCATE 15,37:PRINT "SEÇIMINIZ....[   ]"
2620 LOCATE 15,52:COLOR 19,8:PRINT ".":COLOR 7,0
2630 LOCATE 15,52:LINE INPUT A$
2635 IF VAL(A$)>4 OR VAL(A$)<1 THEN 2600
2636 IF VAL(A$)=4 THEN 1000
2640 J=VAL(A$)
2650 LOCATE 24,8:PRINT SPACE$(70);
```

```
2670 LOCATE 24,12:COLOR 19,8: PRINT  "BILGILER   YUKLENIP
SIRALANACAK LUTFEN BEKLEYINIZ!";:COLOR 7,0
2680 GOSUB 7000'BILGI YUKLEME
2685 IF SAYI=0 THEN SAYI=1
2690 GOSUB 7200'SIRALAMA
2700 GOSUB 4300'L. EKRAN CIZME
2705 GOSUB 8600
2710 SAYFA=1:N=7:I1=7
2720 GOSUB 6500
2730 GOSUB 6850
2735 GOSUB 6750
2740 Z$=INKEY$:IF Z$="" THEN 2740
2750 IF MID$(Z$,2,1)="H" THEN 2830
2760 IF MID$(Z$,2,1)="P" THEN 2860
2770 IF MID$(Z$,2,1)="I" THEN 2891
2780 IF MID$(Z$,2,1)="Q" THEN 2893
2800 FF=ABS(ASC(Z$)-223)
2810 ON FF GOTO 2600,2821,2822,3400,4500
2820 GOTO 2740
2821 DURUM(N-6)=0:LOCATE I1,17:COLOR 0,7:BEEP:PRINT  "OKEY
":COLOR 7,0:GOTO 27400
2822 DURUM(N-6)=1:LOCATE I1,17:COLOR 0,7:BEEP:PRINT "iPTAL":
COLOR 7,0:GOTO 2740
2830 IF I1=7 AND SAYFA=1 THEN 2730
2840 IF I1=7 THEN SAYFA=SAYFA-1:N=N-1:I1=21:GOSUB 3300:
GOTO 2720
2850 GOSUB 6850:I1=I1-1:N=N-1:GOTO 2735
2860 GOSUB 6850
2870 I1=I1+1:N=N+1
2875 IF N>SAYI+6 THEN  N=SAYI+6:I1=SAYI-(SAYFA-1)*15+6:
GOTO 2730
2880 IF I1>21 THEN SAYFA=SAYFA+1:I1=7:GOSUB 3300:GOTO 2720
2890 GOTO 2735
2891  IF SAYFA=1 THEN 2892 ELSE  SAYFA=SAYFA-1:GOSUB  3300:
N=(SAYFA-1)*15+7:I1=7:GOTO 2720
2892 GOSUB 6850:I1=7:N=(SAYFA-1)*15+7:GOTO 2735
2893 SAYFA=SAYFA+1:IF SAYI<(SAYFA-1)*15 THEN SAYFA=SAYFA-1:
N=SAYI+6:I1=SAYI-(SAYFA-1)*15+6 :GOTO 2720
```

```
2894 GOSUB 3300:N=(SAYFA-1)*15+7:I1=7:GOTO 2720
3000 A$="┌"+STRING$(78,"─")+"┐"
3010 LOCATE 1,1:PRINT A$
3020 A$="│"+STRING$(78," ")+"│"
3030 LOCATE 2,1:PRINT A$
3040 A$="└"+STRING$(78,"─")+"┘"
3050 LOCATE 3,1:PRINT A$
3060 LOCATE 2,2:COLOR 0,7:PRINT  " YILDIZ  UNIVERSITESI
MATEMATIK MUHENDISLIGI I":COLOR 7,0
3070 LOCATE 2,49:PRINT "HAZIRLAYANLAR:"
3080 LOCATE 2,64:COLOR 0,7:PRINT "G.AKYOL-N.GULER ":COLOR 7,0
3090 A$="┌"+STRING$(78,"─")+"┐"
3100 LOCATE 23,1:PRINT A$;
3110 A$="│"+STRING$(78," ")+"│"
3120 LOCATE 24,1:PRINT A$;
3130 A$="└"+STRING$(78,"─")+"┘"
3140 LOCATE 25,1:PRINT A$;
3150 LOCATE 24,2:COLOR 0,7:PRINT "MESAJ:";:COLOR 7,0:COLOR 19,8:
PRINT " 1,2,3,4 RAKAMLARINDAN BIRINE YAZARAK ENTER  'E BASINIZ..
COLOR 7,0
3160 RETURN
3200 FOR I=4 TO 22
3210 LOCATE I,1:PRINT SPACE$(80)
3220 NEXT I
3230 LOCATE 24,8:PRINT SPACE$(70);
3240 RETURN
3300 REM ***EKRAN SILME2***
3310 FOR S=7 TO 21
3320 LOCATE S,12:PRINT SPACE$(4)
3330 LOCATE S,17:PRINT SPACE$(5)
3340 LOCATE S,23:PRINT SPACE$(12)
3350 LOCATE S,36:PRINT SPACE$(21)
3360 LOCATE S,58:PRINT SPACE$(12)
3370 NEXT S
3380 RETURN
```

```
3400 REM****KAYIT****
3410 GOSUB 3850
3420 GOTO 2705
3500 'EKRAN CIZME
3510 A$="┌──────┬"+STRING$(12,"─")+"┬"+STRING$(21,"─")+"┬"+
STRING$(12,"─")+"┐"
3520 LOCATE 4,7:PRINT A$;
3530 A$="│"+STRING$(5," ")+"│"+STRING$(12," ")+"│"+
STRING$(21," ")+"│"+STRING$(12," ")+"│"
3540 LOCATE 5,7: PRINT A$;
3550 A$="├"+STRING$(5,"─")+"┼"+STRING$(12,"─")+"┼"+
STRING$(21,"─")+"┼"+STRING$(12,"─")+"┤"
3560 LOCATE 6,7: PRINT A$;
3570 A$="│"+STRING$(5," " )+"│"+STRING$(12," ")+"│"+
STRING$(21," ")+"│"+STRING$(12," ")+"│"
3580 FOR I=7 TO 21
3590 LOCATE I,7: PRINT A$;
3600 NEXT I
3610 A$="└"+STRING$(5,"─" )+"┴"+STRING$(12,"─")+"┴"+
STRING$(21,"─")+"┴"+STRING$(12,"─")+"┘"
3620 LOCATE 22,7: PRINT A$;
3625 RETURN
3630 LOCATE 10,65:A$="┌"+STRING$(14,"─")+"┐"
3640 PRINT A$;
3650 LOCATE 11,65:A$="│"+STRING$(14," ")+"│"
3660 PRINT A$;
3670 LOCATE 12,65:A$="├"+STRING$(14,"─")+"┤"
3680 PRINT A$;
3690 A$="│"+STRING$(14," ")+"│"
3700 FOR I=13 TO 17
3710 LOCATE I,65:PRINT A$;
3720 NEXT I
3730 A$="└"+STRING$(14,"─")+"┘"
3740 LOCATE 18,65:PRINT A$;
3745 RETURN
3750 LOCATE 11,66:COLOR 0,7:PRINT " HAT KODLARI ";:COLOR 7,0
3760 LOCATE 14,66:PRINT "[1] IC HAT"
```

```
3770 LOCATE 15,66:PRINT "[2] DIŞ HAT"
3780 LOCATE 16,66:PRINT "[3] ŞEHİR DIŞI"
3790 LOCATE 24,8:PRINT SPACE$(70);
3800 LOCATE 24,10:COLOR 0,7:PRINT"‹F1›";:COLOR 7,0:
LOCATE 24,14:PRINT" ANA MENÜ'YE DÖNÜŞ";
3810 LOCATE 24,37:COLOR 0,7:PRINT"‹F2›";:COLOR 7,0:
PRINT" KAYIT ";
3820 LOCATE 24,51:COLOR 0,7:PRINT"‹F3›";:COLOR 7,0:
PRINT" VAZGEÇ ";
3830 RETURN
3850 'YENİ KAYIT YAPMA
3860 GOSUB 8500
3880 KILL HAT$
3890 OPEN "R",#1,HAT$,46
3900 FIELD #1,1 AS HATK$,12 AS UN$,21 AS AD$, 12 AS TEL$
3905 G=0
3910 FOR I2=1 TO SAYI
3915 IF DURUM(I2)=1 THEN 3950
3920 LSET HATK$=KIM$(I2,1):LSET UN$=KIM$(I2,2)
3930 LSET AD$=KIM$(I2,3):LSET TEL$=KIM$(I2,4)
3940 G=G+1:PUT #1,G
3950 NEXT I2
3960 CLOSE #1
3970 LOCATE 24,8:PRINT SPACE$(70);:GOSUB 8000
3980 RETURN
4000 LOCATE 5,8:PRINT "HAT K";
4010 LOCATE 5,17:PRINT "ÜNVAN";
4020 LOCATE 5,31:PRINT "ADI SOYADI";
4030 LOCATE 5,52:PRINT "TEL-NO";
4040 RETURN
4100 '  EKRAN SİLME (K)
4110 FOR I=7 TO 21
4120 LOCATE I,8:PRINT SPACE$(5)
4130 LOCATE I,14:PRINT SPACE$(12)
4140 LOCATE I,27:PRINT SPACE$(21)
4150 LOCATE I,49:PRINT SPACE$(12)
4160 NEXT I
4170 RETURN
```

```
4200 'EKRANA BILGI AKTARMA
4210 I=6
4220 FOR K=(SAYFA-1)*15+1 TO SAYFA*15
4230 I=I+1
4240 LOCATE I,10:PRINT KIM$(K,1)
4250 LOCATE I,14:PRINT KIM$(K,2)
4260 LOCATE I,27:PRINT KIM$(K,3)
4270 LOCATE I,49:PRINT KIM$(K,4)
4280 NEXT K
4290 RETURN
4300 'LISTELEME EKRANI
4310 A$="┌"+STRING$(4,"─")+"┬"+STRING$(5,"─")+"┬"+
STRING$(12,"─")+"┬"+STRING$(21,"─")+"┬"+STRING$(12,"─")+"┐"
4320 LOCATE 4,11:PRINT A$;
4330 A$="│"+STRING$(4," ")+"│"+STRING$(5," ")+"│"+
STRING$(12," ")+"│"+STRING$(21," ")+"│"+STRING$(12," ")+"│"
4340 LOCATE 5,11:PRINT A$;
4350 A$="├"+STRING$(4,"─")+"┼"+STRING$(5,"─")+"┼"+
STRING$(12,"─")+"┼"+STRING$(21,"─")+"┼"+STRING$(12,"─")+"┤"
4360 LOCATE 6,11:PRINT A$;
4370 A$="│"+STRING$(4," ")+"│"+STRING$(5," ")+"│"+
STRING$(12," ")+"│"+STRING$(21," ")+"│"+STRING$(12," ")+"│"
4380 FOR U=7 TO 21
4385 LOCATE U,11:PRINT A$;
4390 NEXT U
4400 A$="└"+STRING$(4,"─")+"┴"+STRING$(5,"─")+"┴"+
STRING$(12,"─")+"┴"+STRING$(21,"─")+"┴"+STRING$(12,"─")+"┘"
4410 LOCATE 22,11:PRINT A$;
4420 LOCATE 5,12:PRINT "SIRA";
4430 LOCATE 5,17:PRINT "DURUM";
4440 LOCATE 5,26:PRINT "UNVAN";
4450 LOCATE 5,41:PRINT "ADI SOYADI";
4460 LOCATE 5,61:PRINT "TEL-NO";
4470 RETURN
```

```
4500 REM***YAZICI***
4510 GOSUB 5100'BAŞLIK
4520 FOR R=1 TO SAYI
4530 LPRINT TAB(2);"*";
4535 LPRINT USING" ### ";R;
4537 LPRINT "  *";TAB(17);KIM$(R,2);TAB(33);"*      ";
KIM$(R,3);TAB(60);"*   ";KIM$(R,4);"*"
4538 LPRINT
4540 NEXT R
4550 END
5000 LOCATE 24,8:PRINT SPACE$(70);
5010 LOCATE 24,11:PRINT "KAYIT YAPMAK İÇİN ";
5020 LOCATE 24,30:COLOR 16,7:PRINT "‹F2›";:COLOR 7,0
5030 LOCATE 24,36:PRINT "'YE BASINIZ";
5040 FOR O=1 TO 10000:NEXT
5050 LOCATE 24,8:PRINT SPACE$(70);
5060 RETURN
5100 REM**** BAŞLIK ****
5110 LPRINT CHR$(27);"G";
5120 LPRINT CHR$(27);"-";CHR$(1)
5130 LPRINT CHR$(27);"W";CHR$(1)
5140 LPRINT TAB(9);"YILDIZ   ÜNİVERSİTESİ"
5150 LPRINT
5160 LPRINT TAB(5);"MATEMATİK  MÜHENDİSLİĞİ  BÖLÜMÜ"
5170 LPRINT
5180 LPRINT TAB(11);"TELEFON REHBERİ"
5190 LPRINT CHR$(27);"-";CHR$(0):LPRINT CHR$(27);"W";CHR$(0)
5192 LPRINT TAB(25);"İÇ HAT TELEFONLAR"
5195 LPRINT STRING$(80,"—")
5200 LPRINT CHR$(27);"-";CHR$(1)
5210 LPRINT TAB(2);"*SIRA NO*";TAB(17);"ÜNVAN";TAB(33);*
"*    ADI SOYADI";TAB(60);"*   TELEFON      *"
5215 LPRINT CHR$(27);"-";CHR$(0)
5220 RETURN
6000 'MENU YAPMA
6010 LOCATE 10,28:PRINT "[1].....İÇ HAT TELEFONLAR"
6020 LOCATE 11,28:PRINT "[2].....DIŞ HAT TELEFONLAR"
```

```
6030 LOCATE 12,28:PRINT "[3].....ŞEHİR DIŞI TELEFONLAR"
6040 LOCATE 13,28:PRINT "[4].....ANA MENUYE DÖNÜŞ"
6050 RETURN
6500 'l.ekranına BİLGİ AKTARMA
6510 I=6
6515 'N=(SAYFA-1)*15+7
6516 FOR K=(SAYFA-1)*15+1 TO SAYFA*15
6520 I=I+1
6525 IF K>SAYI THEN 6590
6530 LOCATE I,12:PRINT USING"### ";K
6540 LOCATE I,17:PRINT W$(DURUM(K))
6550 LOCATE I,23:PRINT KIM$(K,2)
6560 LOCATE I,36:PRINT KIM$(K,3)
6570 LOCATE I,58:PRINT KIM$(K,4)
6580 NEXT K
6590 RETURN
6750 'SATIR BOYAMA 1
6760 LOCATE I1,12:COLOR 0,7:PRINT USING"### "; N-6 :COLOR 7,0
6770 LOCATE I1,17:COLOR 0,7:PRINT W$(DURUM(N-6)):COLOR 7,0
6780 LOCATE I1,23:COLOR 0,7:PRINT KIM$(N-6,2):COLOR 7,0
6790 LOCATE I1,36:COLOR 0,7:PRINT KIM$(N-6,3):COLOR 7,0
6800 LOCATE I1,58:COLOR 0,7:PRINT KIM$(N-6,4):COLOR  7,0
6810 RETURN
6850 'SATIR SİLME 1
6860 LOCATE I1,12:COLOR 7,0:PRINT USING"### "; N-6
6870 LOCATE I1,17:COLOR 7,0:PRINT W$(DURUM(N-6))
6880 LOCATE I1,23:COLOR 7,0:PRINT KIM$(N-6,2)
6890 LOCATE I1,36:COLOR 7,0:PRINT KIM$(N-6,3)
6900 LOCATE I1,58:COLOR 7,0:PRINT KIM$(N-6,4)
6910 RETURN
7000 ' DOSYA AÇMA
7010 HAT$="EMRE"+MID$(STR$(J),2,1)+".DAT"
7020 OPEN "R",#1,HAT$,46
7030 FIELD #1,1 AS HATK$,12 AS UN$,21 AS AD$, 12 AS TEL$
7035 SAYI=0
7040 FOR I%=1 TO LOF(1)/46
7045 SAYI=SAYI+1
```

```
7050 GET #1, I%
7070 KIM$(I%,1)=HATK$:KIM$(I%,2)=UN$:KIM$(I%,3)=AD$:
KIM$(I%,4)=TEL$
7080 NEXT I%:CLOSE #1
7100 RETURN
7200 'SIRALAMA
7210 F=1
7220 WHILE F
7230 F=0
7240 FOR I=1 TO SAYI-1
7250 IF KIM$(I,3)>KIM$(I+1,3) THEN FOR K=2 TO 4:
SWAP KIM$(I,K),KIM$(I+1,K):NEXT K:F=1
7260 NEXT I
7270 WEND
7280 RETURN
8000 REM***MESAJ YAZMA***
8010 LOCATE 24,8:PRINT SPACE$(70);
8020 LOCATE 24,10:COLOR 0,7:PRINT"<F1>";:COLOR 7,0:
LOCATE 24,14:PRINT"  ANA MENU'YE DÖNUŞ";
8030 LOCATE 24,37:COLOR 0,7:PRINT"<F2>";:COLOR 7,0:
PRINT"  KAYIT ";
8040 LOCATE 24,51:COLOR 0,7:PRINT"<F3>";:COLOR 7,0:
PRINT"  ÇIKIŞ ";
8050 RETURN
8500 'MESAJ1
8510 LOCATE 24,8:PRINT SPACE$(70);
8520 LOCATE 24,12:COLOR 19,8:PRINT "KAYIT YAPILIYOR LÜTFEN
BEKLEYİNİZ !...";:COLOR 7,0
8530 RETURN
8600 REM***L. MESAJ***
8610 LOCATE 24,8:PRINT SPACE$(70);
8620 LOCATE 24,10:COLOR 0,7:PRINT "<F1>";:COLOR 7,0:
PRINT " ÇIKIŞ  ";
8630 COLOR 0,7:PRINT "<F2>";:COLOR 7,0:PRINT " OKEY   ";
8640 COLOR 0,7:PRINT "<F3>";:COLOR 7,0:PRINT " İPTAL   ";
8650 COLOR 0,7:PRINT "<F4>";:COLOR 7,0:PRINT " KAYIT   ";
8655 COLOR 0,7:PRINT "<F5>";:COLOR 7,0:PRINT " YAZICI";
8660 RETURN
```

```
9000 'OKU
9010 '
9020 IF SC$="" THEN 9100
9030 XC=VAL(MID$(SC$,1,2))
9040 YC=VAL(MID$(SC$,3,2))
9050 XL=VAL(MID$(SC$,5,2))
9060 FOR V=1 TO XL
9070 SCH$=SCH$+CHR$(SCREEN(XC,YC+V-1))
9080 NEXT V
9090 COLOR 0,7:LOCATE XC,YC:PRINT SCH$:COLOR 7,0
9100 ZX=VAL(MID$(IN$,1,2))
9110 ZY=VAL(MID$(IN$,3,2))
9120 ZL=VAL(MID$(IN$,5,2))
9130 CP=1:ZC=0:FF=0
9140 COLOR 9,0
9150 LOCATE ZX,ZY:PRINT SPC(ZL)
9160 LOCATE ZX,ZY:PRINT ZZ$
9170 ZP=ZY+CP-1
9180 COLOR 0,7:LOCATE ZX,ZP:PRINT CHR$(SCREEN (ZX,ZP)):
COLOR 9,0
9190 Z1$=INKEY$:IF Z1$="" THEN 9190
9200 IF MID$(Z1$,2,1)="K" THEN 9420
9210 IF MID$(Z1$,2,1)="M" THEN 9450
9220 IF MID$(Z1$,2,1)="H" THEN ZC=-1:GOTO 9500
9230 IF MID$(Z1$,2,1)="P" THEN ZC=1:GOTO 9500
9240 IF MID$(Z1$,2,1)="R" THEN 9520
9250 IF MID$(Z1$,2,1)="S" THEN 9790
9260 IF MID$(Z1$,2,1)="O" THEN 9817
9270 IF MID$(Z1$,2,1)="G" THEN 9190
9280 IF MID$(Z1$,2,1)="I" THEN ZC=-2:GOTO 9500
9290 IF MID$(Z1$,2,1)="Q" THEN ZC=2:GOTO 9500
9300 ZA=ASC(Z1$)
9310 IF ZA=13 THEN ZC=1:GOTO 9500
9320 IF ZA=8  THEN 9830
9330 IF ZA>223 AND ZA<324 THEN FF=ZA-223:GOTO 9500
9335 IF ZA>96 AND ZA<123 THEN Z1$=CHR$(ZA-32)
9340 IF LEN(ZZ$)=0 OR CP>LEN(ZZ$) THEN 9380
```

```
9350 CP=CP+1:ZZ$=MID$(ZZ$,1,CP-2) +Z1$+MID$(ZZ$,CP,LEN(ZZ$)-CP+
9360 IF CP>ZL THEN CP=ZL
9370  GOTO 9150
9380 IF LEN(ZZ$)=ZL THEN 9190
9390 CP=CP+1:ZZ$=ZZ$+Z1$
9400 IF CP>ZL THEN CP=ZL
9410 GOTO 9150
9420 IF CP=1 THEN 9190
9430 LOCATE ZX,ZP:PRINT CHR$(SCREEN (ZX,ZP))
9440 CP=CP-1:GOTO 9170
9450 IF CP=ZL THEN 9190
9460 IF CP>LEN(ZZ$) THEN 9190
9470 CP=CP+1
9480 LOCATE ZX,ZP:PRINT CHR$(SCREEN (ZX,ZP))
9490 GOTO 9170
9500 COLOR 7,0:LOCATE ZX,ZY:PRINT SPC(ZL)
9510 LOCATE ZX,ZY:PRINT ZZ$:GOTO 9890
9520 IF LEN(ZZ$)=0 THEN ZK$="":ZT$="":GOTO 9550
9530 IF CP>LEN(ZZ$) THEN ZK$=ZZ$:ZT$="" :GOTO 9550
9540 ZK$=MID$(ZZ$,1,CP-1):ZT$=MID$(ZZ$,CP,LEN(ZZ$)-LEN(ZK$))
9550 FOR K=1 TO ZL-LEN(ZZ$)
9560 LOCATE ZX,ZY:PRINT SPC(ZL)
9570 LOCATE ZX,ZY:PRINT ZZ$
9580 ZP=ZY+CP-1
9590 LOCATE ZX,ZP:COLOR 0,9:PRINT CHR$(SCREEN (ZX,ZP)):
COLOR 9,0
9600 Z1$=INKEY$:IF Z1$="" THEN 9600
9610 IF MID$(Z1$,2,1)="K" THEN ZZ$=ZK$+ZT$:GOTO 9420
9620 IF MID$(Z1$,2,1)="M" THEN ZZ$=ZK$+ZT$:GOTO 9450
9630 IF MID$(Z1$,2,1)="H" THEN ZC=-1:ZZ$=ZK$+ZT$:GOTO 9500
9640 IF MID$(Z1$,2,1)="P" THEN ZC=1:ZZ$=ZK$+ZT$:GOTO 9500
9650 IF MID$(Z1$,2,1)="R" THEN ZZ$=ZK$+ZT$:GOTO 9150
9660 IF MID$(Z1$,2,1)="S" THEN ZZ$=ZK$+ZT$:GOTO 9790
9670 IF MID$(Z1$,2,1)="O" THEN ZZ$=ZK$+ZT$:GOTO 9817
9680 IF MID$(Z1$,2,1)="G" THEN ZZ$=ZK$+ZT$:GOTO 9600
9690 IF MID$(Z1$,2,1)="I" THEN ZZ$=ZK$+ZT$:GOTO 9600
9700 IF MID$(Z1$,2,1)="Q" THEN ZZ$=ZK$+ZT$:GOTO 9600
```

```
9710 ZA=ASC(Z1$)
9720 IF ZA=13 THEN ZC=1:ZZ$=ZK$+ZT$:GOTO 9500
9730 IF ZA=8  THEN ZZ$=ZK$+ZT$:GOTO 9830
9740 IF ZA>223 AND ZA<234 THEN FF=ZA-223:ZZ$=ZK$+ZT$:
GOTO 9500
9745 IF ZA>96 AND ZA<123 THEN Z1$=CHR$(ZA-32)
9750 ZK$=ZK$+Z1$:CP=CP+1:ZZ$=ZK$+ZT$
9760 NEXT K
9770 ZZ$=ZK$+ZT$:CP=ZL
9780 GOTO 9150
9790 IF LEN(ZZ$)=0 OR CP>LEN(ZZ$) THEN 9190
9800 IF CP=ZL OR LEN(ZZ$)=CP THEN ZZ$=LEFT$(ZZ$,CP-1):
GOTO 9150
9810 IF CP>ZL THEN CP=ZL:GOTO 9150
9815 ZZ$=MID$(ZZ$,1,CP-1)+MID$(ZZ$,CP+1,LEN(ZZ$)-CP):
GOTO 9150
9817 IF CP>ZL OR LEN(ZZ$)=ZL THEN CP=ZL:GOTO 9150
9820 CP=LEN(ZZ$)+1:GOTO 9150
9830 IF CP=1 THEN  ZZ$=MID$(ZZ$,2,LEN(ZZ$)):GOTO 9150
9840 IF CP=2 THEN ZZ$=MID$(ZZ$,2,LEN(ZZ$)-1):GOTO 9880
9850 IF CP>LEN(ZZ$) THEN ZZ$=LEFT$(ZZ$,CP-2):GOTO 9880
9860 IF LEN(ZZ$)=CP THEN ZZ$=LEFT$(ZZ$,LEN(ZZ$)-2)+
RIGHT$(ZZ$,1):GOTO 9880
9870 ZZ$=LEFT$(ZZ$,CP-2)+MID$(ZZ$,CP,LEN(ZZ$)-CP+1)
9880 CP=CP-1:GOTO 9150
9890 COLOR 7,0
9900 LOCATE XC,YC:PRINT SPC(XL)
9910 LOCATE XC,YC:PRINT SCH$
9920 SCH$="":ZP$=ZZ$:ZZ$=""
9930 RETURN
9999 SAVE"GULNUR.BAS",A
10500 FOR I3=4 TO 21
10600 LOCATE I3,1:PRINT SPACE$(80);:NEXT I3:LOCATE 24,8:
PRINT SPACE$(70);
10700 LOCATE 24,25:COLOR 19,8
10710 PRINT"P R O G R A M    S O N U";:COLOR 7,0
10800 LOCATE 10,10:END
```

Örnek III.12.2.1.

Aşağıda verilmiş olan örnek program Yıldız Üniversitesi Matematik Mühendisliği Bölüm Başkanlığına ait kitapların kayıtlarını , ilgili dosyalarda saklayan bir programdır.

```
1 REM ************ YILDIZ ÜNİVERSİTESİ *********************
2 REM ************ MATEMATİK MÜHENDİSLİĞİ ******************
3 REM ******* ELEKTRONİK HESAP I-2 DÖNEM ÖDEVİ *************
4 REM ***************** ÖDEVİ HAZIRLAYANLAR *****************
5 REM ***************** 8861036 Coşar BAYKAL   **************
6 REM ***************** 8861006 Tahir ŞİŞMAN   **************
7 REM ***** Yöneten * Arş.Gör. İBRAHİM EMİROĞLU ************
8 REM ***** Konu     * KİTAPLIK PROGRAMI *********************
20 KEY OFF:COLOR 7,0:CLS:GOSUB 1630:GOSUB 1460
30 REM ******************* M E N U ****************************
40 TUS$=INKEY$
50 LOCATE 4,7:COLOR RN(1),RN(2):PRINT "..DOSYA HAZIRLAMA.."
60 LOCATE 5,7:COLOR RN(3),RN(4):PRINT "...BİLGİ GİRİŞİ...."
70 LOCATE 6,7:COLOR RN(5),RN(6):PRINT "....GÖRÜNTÜLEME...."
80 LOCATE 7,7:COLOR RN(7),RN(8):PRINT ".....LİSTELEME....."
90 LOCATE 8,7:COLOR RN(9),RN(10):PRINT".....DÜZELTME......"
100 LOCATE 9,7:COLOR RN(11),RN(12):PRINT"......EMANET......."
110 LOCATE 10,7:COLOR RN(13),RN(14):PRINT"......SİLME........"
120 LOCATE 11,7:COLOR RN(15),RN(16):PRINT"......ÇIKIŞ........"
130 COLOR 7,0
140 IF  TUS$=SF$+"P"  THEN  SY=SY+2: GOSUB 180: RN(SY-2)=7:
RN(SY-1)=0:RN(SY)=0:RN(SY+1)=7
150 IF TUS$=SF$+"H" THEN SY=SY-2:GOSUB 200:RN(SY)=0:RN(SY+1
)=7:RN(SY+2)=7:RN(SY+3)=0
160 IF TUS$=CHR$(13) THEN ON SY GOSUB 220,1,510,1,800,1,890
,1,1230,1,2030,1,1330,1,1440
170 GOTO 40
180 IF SY>15 THEN SY=15
190 RETURN
200 IF SY<1 THEN SY=1
210 RETURN
```

```
220 REM ********** D O S Y A   H A Z I R L A M A **********
230 LOCATE 4,36:PRINT "EMIN MISINIZ?[E/H]":CV$=INPUT$(1)
240 IF CV$="E" OR CV$="e" THEN 250 ELSE LOCATE 4,36:PRINT S
IL$:RETURN
250 EK=3:FOR TN1=1 TO 17
260 ISIM$=EK$(TN1):GOSUB 430
270 BOS$="B"
280 FOR TN2=1 TO 49:NO1$=STR$(TN2)
290 IF TN2<10 THEN NO2$="0"+MID$(NO1$,2,1)
300 IF TN2>=10 THEN NO2$=MID$(NO1$,2,2)
310 LSET NO$=NO2$:LSET TUR$=EK$(TN1):LSET IM$=BOS$:PUT#1,TN
2:NEXT TN2
320 IF ISIM$="EMA" THEN 350
330 LOCATE TN1+EK,36:PRINT "KITAP";EK$(TN1);" DOSYASI HAZIR
!":BEEP
340 IF TN1=8 THEN EK=-5:GOSUB 380:GOSUB 400
350 NEXT TN1
360 GOSUB 380:GOSUB 400
370 RETURN
380 REM *************** B E K L E M E **********************
390 FOR BK=0 TO 4000:NEXT BK:RETURN
400 REM ************ P E N C E R E   S İ L M E ************
410 FOR I=11 TO 4 STEP -1:LOCATE I,36:PRINT SIL$:NEXT I
420 FOR AL=0 TO 4:LOCATE 15+AL,2:PRINT SPC(3):LOCATE 15+AL,
6:PRINT SPC(2):LOCATE 15+AL,9:PRINT SPC(34):LOCATE 15+AL,44
:PRINT SPC(26):LOCATE 15+AL,71:PRINT SPC(8):NEXT AL: RETURN
430 REM ********** D O S Y A   A Ç M A **********************
440 ON ERROR GOTO 490
450 ATA$="A:KITAP"+ISIM$+".DAT"
460 CLOSE#1:OPEN "R",#1,ATA$,74
470 FIELD#1,1 AS IM$,3 AS TUR$,2 AS NO$,34 AS AD$,26 AS YAZ
$, 8 AS TR$
480 RETURN
490 IF ERR=71 AND ERL=460 THEN LOCATE 8,36:PRINT "DİSKET YO
K.DİSKETİ TAKIP BİR TUŞA BASIN!":GOSUB 500:ART=0:GOSUB 1960
:RESUME 430
500 IF INKEY$="" THEN 500 ELSE RETURN
```

```
510 REM *********B İ L G İ  G İ R İ Ş İ **********************
520 GOSUB 1720
530 GOSUB 1850
540 IF IM$="B" THEN LOCATE 8,36:PRINT "BİLGİLERİ  GİRİN!":
GOTO 680
550 IF IM$="D" THEN LOCATE 8,36:PRINT "DAHA ÖNCE YAPILMIŞ
KAYIT!":GOSUB 1950:GOTO 530
560 IF IM$="S" THEN LOCATE 8,36:PRINT "DAHA ÖNCE SİLİNMİŞ
KAYIT!":GOSUB 1950:GOTO 530
570 REM ************** O K U M A ****************************
580 BL=1
590 GIR$=INPUT$(1)
600 IF GIR$=CHR$(13) THEN RETURN
610 IF GIR$=CHR$(8) THEN BL=BL-1:GOSUB 660:LOCATE 15+ART,DG
2+BL :PRINT " ":GOTO 590
620 IF BL>DG1 THEN BL=DG1
630 CV=ASC(GIR$):IF CV<32 OR CV>122 THEN 590
640 LOCATE 15+ART,DG2+BL:PRINT GIR$:BL=BL+1
650 GOTO 590
660 IF BL<1 THEN BL=1
670 RETURN
680 REM ************* G İ R İ Ş ****************************
690 GOSUB 700:GIT=1:GOTO 1980
700 X1=4:LSET IM$="D":LSET TUR$=ISIM$:LSET NO$=NM$
710 LOCATE  13,22:COLOR 0,7:PRINT BIL1$:COLOR 7,0
720 DG1=34:DG2=8:GOSUB 570:GOSUB 790:LSET AD$=AD1$
730 LOCATE  13,52:COLOR 0,7:PRINT BIL2$:COLOR 7,0
:LOCATE 13,22:PRINT BIL1$
740 DG1=26:DG2=43:GOSUB 570:GOSUB 790:LSET YAZ$=AD1$
750 LOCATE  13,72:COLOR 0,7:PRINT BIL3$:COLOR 7,0:LOCATE 13
,52:PRINT BIL2$
760 IF IM=1 THEN X1=8
770 DG1=X1:DG2=70:GOSUB 570:GOSUB 790:LSET TR$=AD1$
780 LOCATE  13,72:PRINT BIL3$:PUT#1,NM:RETURN
790 AD1$="":FOR KL=1 TO DG1:AD1$=AD1$+CHR$(SCREEN(15+ART,DG
2+KL)):NEXT KL:RETURN
```

```
800 REM ********** G Ö R Ü N T Ü L E M E ******************
810 GOSUB 1720
820 GOSUB 1850
830 IF IM$="B" THEN LOCATE 8,36:COLOR 23,0:PRINT "KAYIT BOŞ!"
:COLOR 7,0:GOSUB 1950:ART=ART-1:GIT=2:GOTO 1980
840 IF IM$="S" THEN LOCATE 8,36:COLOR 23,0:PRINT "BU KAYIT
SİLİNMİŞ!":COLOR 3,0:GOSUB 860:COLOR 7,0:SAY1=0:GOSUB 1950:
GIT=2:GOTO 1980
850 IF IM$="D" THEN GOSUB 860:GIT=2:GOTO 1980
860 LOCATE 15+ART,9:PRINT AD$:LOCATE 15+ART,44:PRINT YAZ$:
LOCATE 15+ART,71:PRINT TR$:RETURN
870 IF ART=5 THEN ART=0:GOSUB 420
880 RETURN
890 REM ********** L İ S T E L E M E ***********************
900 GOSUB 910:GOTO 950
910 LOCATE 22,5:COLOR 15,0:PRINT "Page Up:":COLOR 7,0:
LOCATE 22,13:PRINT "İleri"
920 LOCATE 22,26:COLOR 15,0:PRINT "Page Down:":COLOR 7,0:
LOCATE 22,36:PRINT "Geri"
930 LOCATE 22,49:COLOR 15,0:PRINT "End:":COLOR 7,0: LOCATE
22,53:PRINT "Çıkış"
940 LOCATE 22,66:COLOR 15,0:PRINT "Sayfa No:":COLOR 7,0:
LOCATE 22,76:PRINT "0":RETURN
950 GOSUB 1720
960 LOCATE 8,36:PRINT "EKRANA   KAĞIDA":LOCATE 8,36:COLOR 23,0
:PRINT "E":LOCATE 8,45:PRINT "K":COLOR 7,0:CV$=INPUT$(1)
970 IF CV$="E" OR CV$="e" THEN LOCATE 8,36:PGE=0:PRINT SIL$
:LOCATE 22,76:PRINT "1":GOTO 1000
980 IF CV$="K" OR CV$="k" THEN LOCATE 6,36:GOTO 1100
990 GOTO 960
1000 FOR TKR=1 TO 5
1010 GET#1,TKR+PGE
1020 IF IM$="S" THEN COLOR 3,0
1030 LOCATE 14+TKR,2:PRINT ISIM$:LOCATE 14+TKR,6:PRINT NO$:
LOCATE 14+TKR,9:PRINT  AD$:LOCATE 14+TKR,44:PRINT YAZ$:LOCA
TE 14+TKR,71:PRINT TR$:COLOR 7,0
1040 NEXT TKR
```

```
1050 GIR$=INKEY$:IF GIR$="" THEN 1050
1060 IF GIR$=SF$+"I" THEN PGE=PGE+5:IF PGE>45 THEN PGE=50
:GOTO 1050:ELSE GOSUB 420:LOCATE 22,75:PRINT PGE/5+1
1070 IF GIR$=SF$+"Q" THEN PGE=PGE-5:IF PGE<0 THEN PGE=0
:GOTO 1050:ELSE GOSUB 420:LOCATE 22,75:PRINT PGE/5+1
1080 IF GIR$=SF$+"O" THEN GOSUB 410:LOCATE 22,5:PRINT SPAC
E$(73):RETURN
1090 GOTO 1000
1100 LOCATE 8,36:PRINT "YAZICIYI HAZIRLAYIP BIR TUŞA BASINI
Z.":CV$=INPUT$(1)
1110 LPRINT CHR$(27);"E"
1120 LPRINT:LPRINT:LPRINT TAB(33);"KİTAP LİSTESİ[";ISIM$;"]"
1130 LPRINT TAB(5);STRING$(75,"*")
1140 LPRINT TAB(5);"TUR NO                KİTAP ADI
      KİTAP YAZARI          TARİH"
1150 LPRINT TAB(5);"*** ** ************************************
** ************************** *******"
1160 GOSUB 430
1170 FOR I=1 TO 50
1180 GET#1,I
1190 IF IM$="B" OR IM$="S" THEN 1210
1200 LPRINT TAB(5);TUR$;" ";NO$;" ";AD$;" ";YAZ$;"  ";LEFT$
(TR$,4)
1210 NEXT I
1220 GOSUB 410:RETURN
1230 REM ********* D Ü Z E L T M E ************************
1240 GOSUB 1720
1250 GOSUB 1850
1260 IF IM$="B" THEN LOCATE 8,36:COLOR 23,0:PRINT "KAYIT BO
Ş!":COLOR 7,0:GOSUB 1950:ART=ART-1:GIT=5:GOTO 1980
1270 IF IM$="S" THEN LOCATE 8,36:COLOR 23,0:PRINT "BU KAYIT
SİLİNMİŞ!":COLOR 3,0:GOSUB 860:COLOR 7,0:SAY1=0:GOSUB 1950:
SAY1=2:GIT=5:GOTO 1980
1280 IF IM$="D" THEN GOSUB 860:GOTO 1290
1290 LOCATE 8,36:PRINT "DÜZELTİLECEK KAYIT BU MU?[E/H]":CV$
=INPUT$(1)
1300 IF CV$="E" OR CV$="e" THEN 1310 ELSE GOSUB 870:SAY1=0:
GOSUB 1960:SAY1=2:GIT=5:GOTO 1980
```

```
1310 GOSUB 700:PUT#1,NM
1320 GIT=5:GOTO 1980
1330 REM ********** K A Y I T   S İ L M E ****************
1340 GOSUB 1720
1350 GOSUB 1850
1360 GOSUB 1370:GIT=4:GOTO 1980
1370 IF IM$="B" THEN LOCATE 8,36:COLOR 23,0:PRINT " KAYIT
BOŞ!":COLOR 7,0:GOSUB 1950:ART=ART-1:RETURN
1380 IF IM$="D" THEN GOSUB 860:GOTO 1400
1390 IF IM$="S" THEN LOCATE 8,36:COLOR 23,0:PRINT "BU KAYIT
SİLİNMİŞ!":COLOR 3,0:GOSUB 860:COLOR 7,0:SAY1=0:GOSUB 1950:
SAY1=2:RETURN
1400 LOCATE 8,36:PRINT "SİLİNECEK KAYIT BU MU?[E/H]":CV$=IN
PUT$(1)
1410 IF CV$="E" OR CV$="e" THEN 1420 ELSE GOSUB 870:SAY1=0:
GOSUB 1960:SAY1=2:RETURN
1420 LSET IM$="S":PUT#1,NM
1430 LOCATE 8,36:PRINT SIL$:RETURN
1440 REM ******** Ç I K I Ş ********************************
1450 CLS:CLOSE#1:KEY ON:END
1460 REM ********** Ç E R Ç E V E ************************
1470 LOCATE 1,1:PRINT "┌";STRING$(29,"─");"┬";STRING$(47,"─
");"┐"
1480 LOCATE 2,1:PRINT "│";TAB(31);"│";TAB(79);"│"
1490 LOCATE 3,1:PRINT "├";STRING$(29,"─");"┼";STRING$(47,"─
");"┤"
1500 FOR T=4 TO 11:LOCATE T,1:PRINT "│";TAB(31);"│";TAB(79)
;"│":NEXT T
1510 LOCATE 12,1:PRINT "├──────┴──┴";STRING$(22,"─");"┴";STRIN
G$(11,"─");"┬";STRING$(26,"─");"┬────────┤"
1520 LOCATE 13,1:PRINT "│    │  │";TAB(43);"│";TAB(70);"│
  │"
1530 LOCATE 14,1:PRINT "├──────┼";STRING$(34,"─");TAB(43);
"┼";STRING$(26,"─");TAB(70);"┼────────┤"
1540 FOR T=15 TO 19:LOCATE T,1:PRINT "│    │  │";TAB(43);"│"
;TAB(70);"│        │":NEXT T
1550 LOCATE 20,1:PRINT "├──────┴──┴";STRING$(34,"─");TAB(43);
"┴";STRING$(26,"─");TAB(70);"┴────────┤"
```

```
1560 LOCATE 21,1:PRINT "|";TAB(79);"|"
1570 LOCATE 22,1:PRINT "|";TAB(79);"|"
1580 LOCATE 23,1:PRINT "└";STRING$(77,"─");TAB(79);"┘"
1590 LOCATE 2,12:PRINT "KİTAPLIK":LOCATE 2,50:PRINT "GİRİŞ
/ÇIKIŞ"
1600 LOCATE 13,2:PRINT "TÜR":LOCATE 13,6:PRINT "NO":LOCATE
13,22:PRINT BIL1$:LOCATE 13,52:PRINT BIL2$:LOCATE 13,72:PR
INT BIL3$
1610 LOCATE 21,2: PRINT "KİTAP TÜRLERİ:" : FOR BL=1 TO 16:
LOCATE 21,12+BL*4:PRINT EK$(BL):NEXT BL
1620 RETURN
1630 REM ********** T A N I M L A M A ******************
1640 DIM EK$(17):FOR T=1 TO 17:READ B$:EK$(T)=B$:NEXT T
1650 SIL$=SPACE$(40):SAY1=2:IM=0
1660 DIM RN(16):SY=1:SF$=CHR$(0):RN(1)=0:RN(2)=7:FOR T=3 TO
15 STEP 2:RN(T)=7:RN(T+1)=0:NEXT T
1670 READ BIL1$,BIL2$,BIL3$,BIL4$,BIL5$
1680 RETURN
1690 DATA MKN,MKV,ELS,FZK,ELK,TKR,POT,ANZ,ANL,CBR,NMK,TRG,
LJK,DFD,MMM,LSM,EMA
1700 DATA KITAP ISMI,YAZAR ISMI,TARIH,ALANIN ADI,SINIF/NO
1710 REM ************* K İ T A P  T Ü R Ü ***************
1720 ART=0:LOCATE 4,36:PRINT "KİTAP TÜRÜ:"
1730 BL=0:ISIM$=""
1740 FOR KL=1 TO 3
1750 GOSUB 1930
1760 IF ASK<65 OR ASK>90 THEN 1750
1770 LOCATE 4,46+KL:PRINT CV$:LOCATE 15+ART,KL+1:PRINT CV$:
ISIM$=ISIM$+CV$
1780 NEXT KL
1790 FOR KL=1 TO 16
1800 IF ISIM$=EK$(KL) THEN GOSUB 430:GOTO 1830
1810 NEXT KL
1820 LOCATE 4,47:PRINT SPC(3):LOCATE 15+ART,2:PRINT SPC(3):
GOTO 1730
1830 GET#1,1:IF IM$="B" OR IM$="S" OR IM$="D" THEN RETURN:
ELSE LOCATE 8,36:PRINT "BU DOSYA HAZIRLANMAMIŞ!":GOSUB 380:
GOSUB 400:LOCATE 22,5:PRINT SPACE$(73):RETURN 170
```

```
1840 REM ********** K O D  N O **********
1850 NM$="":LOCATE 6,40:PRINT "KOD NO:"
1860 LOCATE 15+ART,2:PRINT ISIM$
1870 GOSUB 1930:IF ASK<48 OR ASK>52 THEN 1870
1880 KL=1:GOSUB 1940
1890 GOSUB 1930:IF ASK<48 OR ASK>57 THEN 1890
1900 KL=2:GOSUB 1940
1910 NM=VAL(NM$):IF NM=0 THEN IF IM=1 THEN RETURN 2050 ELSE
GOSUB 400:CLOSE#1:RETURN 170
1920 GET#1,NM:RETURN
1930 CV$=INPUT$(1):ASK=ASC(CV$):RETURN
1940 NM$=NM$+CV$:LOCATE 6,46+KL:PRINT CV$:LOCATE 15+ART,KL+5
:PRINT CV$:RETURN
1950 GOSUB 380
1960 LOCATE 8,36:PRINT SIL$:LOCATE 6,47:PRINT SPC(2):LOCATE
15+ART,6:PRINT SPC(SAY1):RETURN
1970 REM ********** K A R A R **********
1980 LOCATE 8,36:PRINT "DEVAM    ÇIKIŞ
:LOCATE 8,36:COLOR 23,0:PRINT "D":LOCATE 8,45:PRINT   "C":
COLOR 7,0:CV$=INPUT$(1)
1990 IF CV$="D" OR CV$="d" THEN ART=ART+1:GOSUB 870:GOSUB 1960
:ON GIT GOTO 530,820,950,1350,1250,2110,2190
2000 IF CV$="C" OR CV$="c" THEN  GOSUB 400:RETURN
2010 GOTO 1980
2020 REM ********** E M A N E T **********
2030 IM=1:PR$=EK$(17):BIL5$=BIL5$+SPACE$(4):SWAP BIL1$,BIL4$
:SWAP BIL2$,BIL5$
2040 GOSUB 1600
2050 GOSUB 410 :LOCATE 4,36: PRINT "[1] GIRIŞ":LOCATE 5,36
:PRINT "[2] LISTE":LOCATE 6,36:PRINT "[3] SILME":LOCATE 7,36
:PRINT "[4] ÇIKIŞ"
2060  CV$=INPUT$(1):CV=VAL(CV$):IF CV<1 OR CV>4 THEN 2060
2070  IF CV=4 THEN SWAP BIL1$,BIL4$:SWAP  BIL2$,BIL5$:GOSUB
410:GOSUB 1600:IM=0:RETURN
2080 GOSUB 410:GOSUB 1720
2090 SWAP PR$,ISIM$:GOSUB 430:SWAP PR$,ISIM$
2100 ON CV GOTO 2110,2170,2190
```

```
2110 GOSUB 1850
2120 IF IM$="B" OR IM$="S" THEN LOCATE 8,36:PRINT "BİLGİLERİ
GİRİN!":GOTO 2140
2130 IF IM$="D" THEN LOCATE 8,36:PRINT "DAHA ÖNCE YAPILMIŞ
KAYIT!":GOSUB 1950:GOTO 2110
2140 GOSUB 700
2150 GIT=6:GOSUB 1980
2160 GOTO 2050
2170 GOSUB 910:GOSUB 960
2180 GOTO 2050
2190 LOCATE 23,1:GOSUB 1850:GOSUB 1370:GIT=7:GOSUB 1980
2200 GOTO 2050
```

IV. BÖLÜM

COBOL PROGRAMLAMA DİLİ

IV.1. COBOL PROGRAMLAMA DİLİNE GİRİŞ

COBOL programlama dili interpreter özelliğe sahip değildir. Bunun için bir yazılım editörü yoktur. Cobol programı yazabilmek için WS.COM , MS.COM , EDLIN.COM , EDIT.COM , TURBO.COM gibi editörlerden faydalanmak gerekir. Bu tip editörlerde bir Cobol programı Cobol programlama dilinin kurallarına uygun bir şekilde yazılarak Cobol derleyicilerinden birisinde program derlenir. Derleme sonucunda herhangi bir hata mevcut ise tekrar editöre girilerek programdaki hatalar düzeltilir ve program tekrar derlenir. Bu işlem cobol derleyicisi hata vermeyinceye kadar devam ettirilir. Eğer derleyici herhangi bir hata vermedi ise program kullanılan cobol derleyicisinin şekline uygun olarak çalıştırılır.

IV.2. COBOL'DA DERLEYİCİ TİPLERİ (VERSİYONLARI)

IV.2.1. Compiler sonucu file uzantısı .OBJ dosya oluşturan COBOL derleyicisi.

Bir COBOL programını bu tip derleyicide derleyebilmek için aşağıdaki aşamaların yapılması gerekir.

a) Cobol programı herhangi bir editörde yazılır sisteme geri dönülür.(prompt'a çıkılır.)

b)

A› COBOL ‹program adı›;

yazılarak enter'e basılır ve program derlenir derleme sonucunda programın yazılmasında herhangi bir hata meydana gelmemiş ise şu işlem yapılır.

c)

A› LINK ‹program adı›;

yazılarak enter'e basılır.

Bu işlemin sonucunda file adı program adı olan ve file uzantısı .EXE olan file oluşturulur.

d) Uzantısı .EXE olan programın adı yazılıp enter'e basılır.

A› ‹program adı›

enter'e basıldıktan sonra program COBRUN.EXE yardımı ile çalıştırılır. Eğer programın çalıştırıldığı sürücüde ki diskette COBRUN.EXE file yok ise çalıştırıcıya bu file'ın hangi sürücüde olduğunu sorar. Herhangi bir sürücüde COBRUN.EXE file 'ı yok ise Cobol programını çalıştımaya imkan yoktur.

IV.2.2. Compiler Sonucu File Uzantısı .INT Olan Dosya Oluşturan COBOL Derleyicisi.

Bir COBOL programını bu tip derleyicide derleyebilmek için aşağıdaki aşamaların yapılması gerekir.

a) Cobol programı herhangi bir editörde yazılır sisteme geri dönülür.(prompt'a çıkılır.)

b)

A› COBOL ‹program adı›;

yazılarak enter'e basılır ve program derlenir derleme sonucunda programın yazılmasında herhangi bir hata meydana gelmemiş ise şu işlem yapılır.

c)

A› RUNCOB ‹program adı›

yazılıp enter'e basıldığı takdirde program çalışır.

IV.3. COBOL PROGRAMLAMA DİLİNDE OPERATÖRLER

+ : TOPLAMA İŞLEMİ.
- : ÇIKARMA İŞLEMİ.
* : ÇARPMA İŞLEMİ.
/ : BÖLME İŞLEMİ.
** : ÜS ALMA İŞLEMİ.

IV.4. COBOL PROGRAMLAMA DİLİNDE KARŞILAŞTIRMA OPERATÖRLERİ

= : EŞİTTİR. (EQUAL)
‹ : KÜÇÜK. (LESS)
› : BÜYÜK. (GREATER)
NOT = : EŞİT DEĞİL. (NOT EQUAL)
NOT ‹ : BÜYÜK EŞİT. (NOT LESS)
NOT › : KÜÇÜK EŞİT (NOT GREATER)

MANTIKSAL OPERATÖRLER

NOT : DEĞİL ANLAMINDADIR.
OR : VEYA ANLAMINDADIR.
AND : VE ANLAMINDADIR.

Örnek.IV.4.a.

IF CEV = "E" OR = "e" GO SON.

Eğer CEV değişkeninin değeri E veya e ise SON adlı paragrafa git.

Örnek IV.4.b.

IF AD = AD1 AND SOYAD = SOYAD1 GO YAZ.

Eğer AD=AD1 ve SOYAD=SOYAD1 ise (her iki şart da gerçekleşiyor ise) YAZ adlı pargrafa git.

Örnek IV.4.c.

IF A NOT < 1 GO GUL.

Eğer A>=1 ise GUL adlı paragrafa git.

IV.5. COBOL PROGRAMLAMA DILININ PROGRAM YAPISI

COBOL programlama dili 4 bölümden oluşur.Bunlar sırası ile :

IV.5.1. IDENTIFICATION DIVISION (Tanıtım Bölümü)

Bu bölümde programın adı , programı yapanın adı , yazıldığı tarih , derlendiği tarih belirtilebilir. Zorunlu bir COBOL kelimesidir. Bu bölümde yapılacak tanımlamalar :

PROGRAM-ID. GS. (programın derleyicideki adı).
AUTHOR. ESRA EMIROGLU. (Programı yapanın adı).
DATE-WRITTEN. 1-6-1989. (Programın yazıldığı tarih).
DATE-COMPILED. 29-6-1989. (Programın derlendiği tarih).

Bunlardan PROGRAM-ID. COBOL için zorunludur. Diğerleri ise programcının isteğine bağlıdır.

IV.5.2. ENVIRONMENT DIVISION (Çevre Bölümü)

Bu bölüm COBOL programının derleneceği ve çalıştırılacağı sistem donanımını belirtmeye yarar. Özellikle dosya ile ilgili açıklamalar bu bölümde yapılır. 2 SECTION 'dan oluşur.

IV.5.2.1. CONFIGURATION SECTION

COBOL programının derleneceği ve çalıştırılacağı sistem donanımı bu SECTION 'da belirtilir.

IV.5.2.2. INPUT-OUTPUT SECTION

Dosya ile ilgili açıklamaların yapıldığı SECTION 'dur. Eğer osya açılacak ise zorunlu bir COBOL kelimesidir.

IV.5.3. DATA DIVISION

Bilgi düzenini bilgi elemanlarının özelliklerini ve

birbirleri ile ilişkilerini tanımladığımız bir bölümdür. Bu bölümde program akışı içerisinde kullanacağımız tüm değişkenler tanımlanmalıdır. 3 SECTION'dan oluşur.

IV.5.3.1. FILE SECTION

Dosya kullanılıyor ise;dosyanın disketteki adının verildiği ve dosya değişkenlerinin tanımlandığı kısımdır.

IV.5.3.2. WORKING-STORAGE SECTION

Program akışı içerisinde kullanılacak değişkenlerin PIC'leri ile verildiği bölümdür.

IV.5.3.3. SCREEN SECTION

Ekran düzenleme ile ilgili tanımlamaların yapıldığı kısımdır.

IV.5.4. PROCEDURE DIVISION

Program mantığının kurulduğu bölümdür. Bir paragraf ismi ile başlar ve STOP RUN deyimi ile sona erer.

IV.6. COBOL PROGRAMLAMA DİLİNDE DEĞİŞKEN (VERİ) TİPLERİ

Cobol programlama dilinde değişkenler PICTURE'ları (PIC) ile tanımlanırlar.

IV.6.1. PICTURE DEYİMİ

Bir kayıt içerisinde verilen bir bilgi parçasının uzunluğu, tipi ve diğer özellikleri , DATA DIVISION bölümünde PIC deyimi ile belirtilir.

Alfabetik bilgiler PIC formatında A ve X sembolleri ile belirtilirler. A sembolü ile belirtilen alfabetik değişkenler rakamları içermezler. X sembolü ile belirtilen alfabetik değişkenler rakamlar dahil tüm karakterleri içerirler.

Sayısal bilgiler PIC formatında 9 veya Z sembolleri ile belirtilirler. Z sembolü ile belirtilen sayısal değişkenler ile aritmetik işlemler yapılamaz. Z sembolü ile tanımlanan sayısal değişkenler , değişkenin solunda bulunan gereksiz sıfırlar ekranda gözükmez. Sayısal bilgilerin ondalık basamaklarını ayırmak için V veya . işareti kullanılabilir. Ondalık kısmı V işareti ile ayrılan sayısal değişkenler için bellekte yer ayrılmaz.

İşaretli sayısal bilgilerin tanımlanabilmesi için (Özellikle negatif sayıları belirtmek için) 9 sembolünün

başına S işareti konulur. Fakat derleyici ekrana yazdırılmak istenen sayıyının önüne işaretini (+ veya -) yazmaz.

PICTURE Deyiminin Genel Yazılım Formatı :

‹Düzey no› ‹Değişken› PIC ‹Değişken Tipi›‹Değişken uzunluğu›
VALUE ‹Değişkenin Başlangıç Değeri›

‹Düzey No› ile tanımlanacak değişkenin düzey numarası tanımlanır. Düzey numaraları (Seviye sayıları) 01-49 sayıları arasında bulunan sayılar ile 77 ve 88 sayıları olabilir.

‹Değişken Tipi› ile , tanımlanmak istenen değişkenin tipi belirtilir.

01 DÜZEY NUMARASI

İlk düzey no (seviye sayısı) 01 düzeyidir. Bu seviye sayısında alt gruplara ayrılan değişkenler tanımlanabilir. Alt grupların ise seviye sayıları 02-49 sayıları arsında olabilir.

01 seviyesinde tanımlanan değişkenlerin alt seviye sayıları 02-49 sayılarıarasında bulunan seviye sayıları ve 88 seviye sayısı olabilir.

Örnek IV.6.1.a.

Bir kimsenin doğum tarihini DOG-TAR adlı bir değişken ile tanımlamak isteyelim. Bu iki ayrı şekilde yapılabilir.

1. Şekil Tanımlama.

```
DATA DIVISION.
WORKING-STORAGE SECTION.
01 DOG-TAR   PIC X(10) VALUE "17-11-1989".
```

Bu tanımlamaya göre DOG-TAR adlı değişkenin değeri 17-11-1989 'dır.

2. Şekil Tanımlama.

DOG-TAR adlı değişken istenilse idi GUN , AY ve YIL olmak üzere üç alt alana ayrılabilir idi.

```
DATA DIVISION.
WORKING-STORAGE SECTION.
01 DOG-TAR.
   02 GUN     PIC X(2) VALUE "17".
   02 AY      PIC X(2) VALUE "11".
```

```
   02 YIL     PIC X(4) VALUE "1989".
```

Bu tanımlamaya göre DOG-TAR adlı değişken 01 seviyesinde tanımlanmış olup GUN , AY ve YIL olarak üç ayrı alt gruba ayrılmıştır.

77 DÜZEY NUMARASI

DATA DIVISION bölümünün WORKING-STORAGE SECTION kısmında , grup bilgisi içermeyen bağımsız değişkenlerin tanımlandığı seviye sayısıdır.

Genel Yazılım Formatı :

77 ‹Değişken Adı› PIC ‹Değişken Tipi› VALUE ‹İlk Değer›.

77 seviyesinde tanımlanmış olan değişkenler bağımsızdır. 77 seviyesinde tanımlanan bir değişkenin alt seviyesi ancak 88 olabilir.

88 SEVİYE SAYISI

01-49 ve 77 düzey numaralarının bir alt seviyeli bilgi türüdür. 88 seviyesinde tanımlanan bir değişkene mutlaka bir değer verilmelidir.

IV.6.2. NUMERİK DEĞİŞKENLER

COBOL 'da nümerik değişkenleri tanımlayabilmek için 9 sembolü kullanılır.

Tamsayı değişkenler için ;

```
77  A     PIC 9(2) VALUE 0.
77  B     PIC 9(2)V99 VALUE 0.
```

ile tanımlanan A değişkeni 2 haneli ve nümerik değişkendir. B değişkeni ise reel sayıdan oluşan bir değişkendir. 2 haneli ve 2 ondalık basamağı vardır.

IV.6.3. ALFANÜMERİK DEĞİŞKENLER

Alfanümerik değişkenlerin tanımlanabilmesi için X veya A sembolü kullanılır.

```
77 C     PIC X(10) VALUE SPACES.
```

C değişkeni alfanümerik bir değişken olup 10 karakterden oluşmaktadır.

IV.6.4. VALUE DEYİMİ

DATA DIVISION bölümünde tanımlanan değişkenlere değer atamak için kullanılır.

Genel Yazılım Formatı :

1. Şekil.

```
          ‹Sayısal karakter›                ‹Sayısal karakter›
VALUE IS  ‹Literal›          THROUGH/THRU ‹Literal›
          ‹Alfabetik karakter›              ‹Alfabetik karakter›
```

Eğer THROUGH veya THRU şeklinde yazılım şekli kullanılır ise , değişkenin alabileceği değerler , THROUGH veya THRU kelimelerinden önce verilen değerden başlayıp , THROUGH veya THRU kelimelerinden sonra verilen değere kadar olan dizi elemanları olabilir.

Örnek:

```
77 GS   PIC X.
   88 VALUE "A" THROUGH "Z".
```

Tanımlamasına göre GS adlı alfabetik değişkenin alabileceği değerler , büyük "A" harfinden büyük "Z" harfine kadar olabilir.

Örnek:

```
77 GS   PIC X.
   88 ES  VALUE "a" THRU "z".
```

Tanımlamasına göre GS adlı alfabetik değişkenin alabileceği değerler , küçük "a" harfinden küçük "z" harfine kadar olabilir.

IV.7. ARITMETIK İŞLEM DEYIMLERI

IV.7.1. ADD KOMUTU

COBOL programlama dilinde toplama işlemi yapmak için kullanılır.

Genel Yazılım Formatı :

```
ADD ‹Değişkenler› TO/GIVING ‹Değişken› ROUNDED
                ON SIZE ERROR ‹Komut›.
```

ADD kelimesinden sonra gelen sayı veya değişkenler toplanıp TO veya GIVING kelimesinden sonra belirtilen değişkene atanır. Eğer TO kullanılmış ise , ADD kelimesinden sonra verilen sayı veya değişkenler toplanarak TO kelimesinden sonra gelen değişkenin üzerine ilave edilir.

Örnek IV.7.1.a.

```
ADD A B C 17 TO D.
```

örneğinde A , B , C değişkenlerinin değerleri ile 17 sayısı toplanarak D değişkeninin üzerine ilave edilir. (Matematiksel olarak D=D+A+B+C+17 işlemi yapılır.)

Eğer GIVING kullanılmış ise , ADD kelimesinden sonra belirtilen sayı veya değişkenler toplanarak GIVING kelimesinden sonra belirtilen değişkene atanırlar.

Örnek IV.7.1.b.

ADD A B C 17 GIVING D.

örneğinde A , B , C değişkenlerinin değerleri ile 17 sayısı toplanarak D değişkenine atanır. (Matematiksel olarak D=A+B+C+17 işlemi yapılır.)

Eğer ROUNDED kullanıldı ise toplama işlemi yapıldıktan sonra toplamın değeri TO veya GIVING kelimesinden sonra verilen değişkene yuvarlatılarak aktarılır.

Eğer ON SIZE ERROR ‹Komut› kullanıldı ise , toplama işlemi esnasında bir hata meydana gelir ise programın akışı ‹komut› ile verilen komuta veya komutlara sapar. Bu toplama esnasında meydana gelebilecek bir takım hataları önlemek için kullanılır.

IV.7.2. SUBTRACT KOMUTU

Cobol programlama dilinde çıkarma işlemi yapmak için kullanılan bir komuttur.

Genel Yazılım Formatı :

SUBTRACT ‹Değişkenler› FROM ‹Değişken1› GIVING ‹Değişken2›
ROUNDED ON SIZE ERROR ‹Komut›.

SUBTRACT kelimesinden sonra verilen sayı veya değişkenler toplanarak FROM kelimesinden sonra verilen değişken1'den çıkartılıp GIVING kelimesinden sonra gelen değişken2'ye atar.

Eğer ROUNDED kullanıldı ise çıkartma işlemi yapıldıktan sonra çıkarma işleminin sonucu GIVING kelimesinden sonra verilen değişkene yuvarlatılarak aktarılır.

Eğer ON SIZE ERROR ‹Komut› kullanıldı ise , çıkarma işlemi esnasında bir hata meydana gelir ise programın akışı ‹komut› ile verilen komuta veya komutlara sapar. Bu çıkarma işlemi esnasında meydana geleilecek bir takım hataları önlemek için kullanılır.

Örnek IV.7.2.a.

SUBTRACT A B C FROM D.

örneğinde A , B , C değişkenlerinin değeri toplanarak D

değişkeninden çıkartılır. (Matematiksel olarak D=D-(A+B+C) işlemi yapılır.)

Örnek IV.7.2.b.

SUBTRACT A B C FROM D GIVING E.

örneğinde A , B , C değişkenlerinin değeri toplanarak D değişkeninden çıkartılır ve çıkan sonuç E değişkenine atanır. (Matematiksel olarak D=D-(A+B+C) işlemi yapılır.)

IV.7.3. MULTIPLY KOMUTU

Cobol programlama dilinde çarpma işlemi yapmak için kullanılan bir komuttur.

Genel Yazılım Formatı :

MULTIPLY‹Değişken veya sayı›BY‹Değişken1›GIVING‹Değişken2›
ROUNDED ON SIZE ERROR ‹Komut›.

MULTIPLY kelimesinden sonra verilen değişken veya sayı ile BY kelimesinden sonra verilen değişken1 ile çarpılarak GIVING kelimesinden sonra verilen değişken veya değişkenlere atanır. Eğer GIVING kelimesi kullanılmadı ise çarpma işleminin sonucu BY kelimesinden sonra verilen değişkene atanır.

Örnek IV.7.3.a.

MULTIPLY A BY B.

örneğinde A sayısı ile B sayısı çarpılarak sonuç B değişkenine atanır. (Matematiksel olarak B=B*A işlemi yapılır.)

Örnek IV.7.3.b.

MULTIPLY 17 BY B GIVING C ROUNDED ON SIZE ERROR GO DUR.

Örneğinde 17 ile B sayısı çarpılarak çıkan sonuç C değişkenine yuvarlatılarak aktarılır. Eğer çarpma işlemi esnasında herhangi bir hata meydana gelecek olur ise programın akışı DUR adlı paragrafa kayacaktır.

IV.7.4. DIVIDE KOMUTU

Cobol programlama dilinde bölme işlemi yapmak için kullanılan bir deyimdir.

Genel Yazılım Formatı :

1. Yazılım Şekli.

DIVIDE ‹Değişken1(Bölen)› INTO ‹Değişken2(Bölünen)›
GIVING ‹Değişken3› ROUNDED ON SIZE ERROR ‹Komut›.

DIVIDE kelimesinden sonra verilen Değişken1 veya bölen sayı , INTO kelimesinden sonra verilen Değişken2 veya bölünen sayı ile bölünerek çıkan sonuç GIVING kelimesinden sonra verilen Değişken3'e aktarılır. Eğer GIVING kelimesi kullanılmaz ise çıkan sonuç INTO kelimesinden sonra verilen Değişken2 veya bölünen sayıya atanır.

Eğer ROUNDED kullanıldı ise bölme işleminin sonucu yuvarlatılarak INTO kelimesinden sonra verilen Değişken2'ye veya GIVING kelimesinden sonra verilen Değişken3'e aktarılır.

Eğer çarpma işleminin yapılması sonucunda herhangi bir hata meydana gelecek olur ise programı akışı ON SIZE ERROR kelimesinden sonra verilen komutlara veya komuta sapar.

2. Yazılım Şekli.

DIVIDE ‹Değişken2(Bölünen)› BY ‹Değişken2(Bölen)›
GIVING ‹Değişken3› ROUNDED ON SIZE ERROR ‹Komut›.

DIVIDE kelimesinden sonra verilen Değişken2 ile ifade edilen bölünen sayı BY kelimesinden sonra verilen Değişken2 veya bölen sayı ile bölünerek , çıkan sonuç DIVIDE kelimesinden sonra verilen Değişken2 veya bölünen sayıya atanır.

Eğer GIVING kullanıldı ise çıkan sonuç GIVING kelimesinden sonra verilen Değişken3'e atanır.

ROUNDED kelimesi kullanıldı ise bölme işleminin sonucu yuvarlatıldıktan sonra ilgili değişkene atanır.

Eğer bölme işlemi esnasında herhangi bir hata oluşacak olur ise programın akışı , ON SIZE ERROR ile verilen komut veya komutlar icra görür.

IV.7.5. COMPUTE KOMUTU

Cobol programlama dilinde matematiksel işaretler (+,-,*,/,**) kullanılarak aritmetik işlemler yapılmak istenildiği zaman kullanılan deyimdir. Bu deyim kullanılarak istenildiği kadar işlem pespeşe yapılabilir.

Genel Yazılım Formatı :

COMPUTE ‹Değişken Adı› ROUNDED = ‹Aritmetik İfade›
ON SIZE ERROR ‹Komut›

Yapılan işlemin sonucu yuvarlatılarak değişkene atanmak

isteniyor ise ‹Değişken Adı›'ından sonra ROUNDED kelimesi kullanılmalıdır.

Aritmetik işlemlerin yapılması esnasında meydana gelebilecek hataları kontrol edebilmek için ON SIZE ERROR cümlesi kullanılabilir. Bu cümle kullanıldı ise işlemlerin yapılması esnasında bir hata meydana gelir ise ON SIZE ERROR ile verilen komut icra görür.

Örnek IV.7.5.a.

COMPUTE A = A + B - C + (B * A - A * B).

örneğinde A+B-C+(B*A-A*B) işleminin sonucu A değişkenine atanır.

IV.7.6. MOVE KOMUTU

Cobol programlama dilinde değişkenler arasında atama işlemlerini yapabilmek için kullanılan bir komuttur. Nümerik olarak tanımlanmış olan bir değişken , alfanümerik olarak tanımlanmış olan bir değişkene atanabilir. Fakat alfanümerik olarak tanımlanmış olan bir değişken nümerik olarak tanımlanmış olan bir değişkene atanamaz.

Genel Yazılım Formatı :

MOVE ‹Değişken-1› TO ‹Değişken-2›.

Genel formda belirtilmiş olan ‹Değişken-1› ile tanımlanmış değişken , ‹Değişken-2› ile tanımlanmış olan değişkene atanır.

Örnek IV.7.6.a.

A değişkeni 9(4) , B değişkeni X(10) olarak tanımlanmış olsun.

MOVE A TO B. yazılımında A değişkeninin değeri B değişkenine atanmaktadır. (B=A) işlemi gerçekleşmektedir.

MOVE B TO A. yazılımı geçerli değildir. Çünkü alfanümerik olarak tanımlanmış olan B değişkeni , nümerik olarak tanımlanmış A değişkenine atanamaz.

IV.8. COBOL PROGRAMLAMA DİLİNDE DİZİLER

COBOL 'da diziler OCCURS kelimesi ile tanımlanırlar. Diziler aynı tipte aynı uzunlukta olan içerikleri açısından birbirleri ile bağlantılı olan belirli miktardaki sahalardan oluşur. WORKING-STORAGE SECTION bölümünde şu şekilde tanımlanabilir.

Örnek IV.8.a.

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. ESRA.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       01 DIZI.
           02 A OCCURS 99 TIMES PIC 9.
       01 AY.
           02 HAFTA OCCURS 4 TIMES.
              03 GUN OCCURS 7 TIMES PIC X(9).
      . . . . . .
```

Tanımlamasına göre;
A dizisi 99 elemandan oluşmakta ve bir tek haneli nümerik bir dizidir. GUN adlı matris ise herbir elemanı 9 karakterden oluşan 4 satır 7 sütunlu bir matristir.

IV.9. COBOL PROGRAMLAMA DİLİNDE GİRİŞ-ÇIKIŞ İŞLEMLERİ

Cobol programlama dilinde giriş çıkış işlemleri olarak DISPLAY ve ACCEPT deyimleri kullanılır. Şimdi sırası ile bunları inceleyelim.

IV.9.1. DISPLAY KOMUTU

Bu komut ekrana yazdırılmak istenen bilgilerin ekrana yazılmasını sağlar. DISPLAY komutunun bir kaç türlü kullanım şekli vardır. Bunların en önemlisi satır sütun belirtilerek yapılan şeklidir.

Genel Yazılım Formatı :

```
DISPLAY(satır no , sütun no) ‹ekrana yazılacak değişken ismi›.
```

Örnek IV.9.1.a.

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. ESRA.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       77 A          PIC 9(5) VALUE 03630.
       77 B          PIC X(19) VALUE "YIDIZ ÜNİVERSİTESİ".
       01 BASLIK.
           02 FILLER       PIC X(35) VALUE SPACES.
```

```
        02 YAZ          PIC X(10) VALUE "GALATASARAY".
        02 FILLER       PIC X(35) VALUE SPACES.
    PROCEDURE DIVISION.
    BASLA.
        DISPLAY(1 1) ERASE.
        DISPLAY(10 10) "A====>" A.
        DIPSLAY(11 10) B.
        DISPLAY(12 1) BASLIK.
        STOP RUN.
```

Programının çalışması sonucu ekranı 1. satır 1. sütunundan itibaren sildikten sonra 10. satır 10. sütuna A değişkeninin değerini yazar. 11. satır 10. sütununa B değerini yazılıp 12. satır 1. sütuna BASLIK değişkeninin değerini yazarak program sona erer.

IV.9.2. ACCEPT KOMUTU

Klavye yolu ile ekrandan , bellekte bulunan değişkenlere bilgi aktarma deyimidir. COBOL programlama dilinde ACCEPT deyiminin 5 türlü kullanım şekli vardır. Bunları sırası ile inceleyelim.

1. Kullanılım Şekli :

ACCEPT ‹Değişken Adı›.

Bu yazılış şekline göre , daha önce PIC deyimi ile tanımlanan değişkenin bellek adresine bilgi aktarma işlemi yapılabilmektedir. Bu bilginin ekranın neresinden ve hangi açıklamaya uygun olarak aktarılması isteniyor ise ACCEPT deyiminden önce bir DISPLAY deyimi kullanılmalıdır. Bu değişken ismi SCREEN SECTION 'da tanımlanmış olan bir ekran düzenleme satır-sütun tablosu da olabilir.

Örnek IV.9.2.a.

AD (15) ve SOYAD (20) adlı değişkenler WORKING-STORAGE SECTION bölümünde tanımlanmış değişkenler olmak üzere;

```
        DISPLAY(10 17) "Adını Giriniz.......: ".
        ACCEPT AD.
        DISPLAY(11 17) "Soyadını Giriniz....: ".
        ACCEPT SOYAD
```

2. Kullanılım Şekli :

ACCEPT ‹Değişken Adı› FROM DATE
DAY
TIME
ESCAPE KEY

Şeklinde kullanılabilir.

a) ACCEPT ‹Değişken adı› FROM DATE Kullanıldı ise :
ACCEPT ile verilen değişken adına sistemin tarihini aktarır. (Makina açıldığında girilmiş olan tarih). Genel formatı YYMMDD olup yıl ay ve gün sırasında herbiri 2 sayısal alandır.

Örnek IV.9.2.b.

Makinanın açıldığı tarih olarak 10.25.1989 girilmiş olsun ; WORKING-STORAGE SECTION bölümünde TARIH adlı değişken aşağıdaki gibi tanımlanmış olsun.

```
01 TARIH.
    02 SENE PIC 99.
    02 AY   PIC 99.
    02 GUN  PIC 99.
```

Tanımlamasına göre;

ACCEPT TARIH FROM DATE. kullanıldığı takdirde ACCEPT ile tanımlanan TARIH adlı değişkene 891025 değeri yüklenir. Bunun ilk iki karakteri SENE adlı değişkene, sonraki iki karakteri AY adlı değişkene , son iki değişken ise GUN adlı değişkene atanır.

b) ACCEPT ‹Değişken adı› FROM DAY Kullanıldı ise :
ACCEPT ile verilen değişken adına sistemin açıldığı tarihin yılın kaçıncı gününe denk geldiğini belirtir. Genel formatı YYNNN olup ilk iki sayısal bilgi yıl için diğer üç sayısal bilgi ise makinanın açıldığı tarihin yılın kaçıncı günü olduğunu belirtir.

Örnek IV.9.2.c.

Makinanın açıldığı tarih bir önceki örnekte olduğu gibi olsun.

GUN adlı değişken ise WORKING-STORAGE SECTION bölümünde aşağıdaki gibi tanımlanmış olsun.

```
    01 GUN.
       02 YIL1     PIC 99.
       02 GUN1     PIC 999.
```

Şeklindeki tanımlamaya göre;

ACCEPT GUN FROM DAY. yazılır ise YIL1 adlı değişkene sistemin açıldığı tarih. YIL1 adlı değişkene 89 değeri, GUN1 adlı değişkene 308 değeri yüklenir. (Buradaki 308 yılın 308. günü olduğunu belirtmektedir.

c) ACCEPT ‹Değişken Adı› FROM TIME kullanıldı ise :

ACCEPT ile tanımlanan değişkene saatin değeri yüklenir. Genel formatı HHMMSSFF olup HH (saat için 0-23) , MM (dakika için 0-59) , SS (saniye için 0-59) , FF (saniyenin 100'de 1'i kadar 0-99) değerlerinden birisi yüklenir.

Örnek IV.9.2.d.

SAAT adlı değişken ise WORKING-STORAGE SECTION bölümünde aşağıdaki gibi tanımlanmış olsun.

```
    01 SAAT.
       02 SAAT1    PIC 99.
       02 DAKIKA   PIC 99.
       02 SANIYE   PIC 99.
       02 SALISE   PIC 99.
```

Şeklindeki tanımlamaya göre ;

ACCEPT ‹Değişken adı› FROM TIME yazılır ise ACCEPT ile belirtilen değişken adına o anki saat değeri yüklenir.

d) ACCEPT ‹Değişken adı› FROM ESCAPE KEY kullanımı :

Uzunlukları iki sayısal veya alfabetik olan fonksiyon tuşlarından hareket edilmesini sağlar. Bu tuşlar :

ESC , ENTER 1 F1 ile F10 arasındaki F tuşlarıdır.

Örnek IV.9.2.e.

F tuşlarının kodlarını öğrenebilmek için aşağıdaki programı yapabiliriz :

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. FTUSLARI.
       ENVIRONMENT DIVISION.
       DATA DIVISON.
       WORKING-STORAGE SECTION.
       77  A        PIC 9 VALUE 0.
```

```
       77  FTUS   PIC X(2) VALUE  SPACES.
       PROCEDURE DIVISION.
       BASLA.
           DISPLAY(10  17) "Kontrol tuslarından (F  tusları)
      -                                    birine basınız :".
           ACCEPT ( , ) A WITH AUTO-SKIP.
           ACCEPT FTUS FROM ESCAPE KEY.
           DISPLAY (11 17) "F tuşunun değeri ....: " FTUS
                                                  " Dır".
           DISPLAY (20 17) "Devam İçin E Yazınız.".
           ACCEPT ( , ) CEV.
           IF CEV = "E" OR = "e" GO BASLA.
           STOP RUN.
```

Yazılan bu program çalıştırıldığı takdirde ;
ENTER tuşuna basıldığı takdirde FTUS adlı değişkene 00 değeri yüklenir.
ESC tuşuna basıldığı takdirde FTUS adlı değişkene 01 değeri yüklenir.
F1 tuşuna basıldığı takdirde FTUS adlı değişkene 02 değeri yüklenir.
F2 tuşuna basıldığı takdirde FTUS adlı değişkene 03 değeri yüklenir.
F3 tuşuna basıldığı takdirde FTUS adlı değişkene 04 değeri yüklenir.
F4 tuşuna basıldığı takdirde FTUS adlı değişkene 05 değeri yüklenir.
F5 tuşuna basıldığı takdirde FTUS adlı değişkene 06 değeri yüklenir.
F6 tuşuna basıldığı takdirde FTUS adlı değişkene 07 değeri yüklenir.
F7 tuşuna basıldığı takdirde FTUS adlı değişkene 08 değeri yüklenir.
F8 tuşuna basıldığı takdirde FTUS adlı değişkene 09 değeri yüklenir.
F9 tuşuna basıldığı takdirde FTUS adlı değişkene 10 değeri yüklenir.
F10 tuşuna basıldığı takdirde FTUS adlı değişkene 11 değeri yüklenir.

3. Kullanım Şekli :

ACCEPT (LIN , COL) ‹Değişken Adı›.

ACCEPT ile belirtilen değişkenin LIN ile belirtilen satır COL ile belirtilen Sütunundan itibaren okuma işlemine başlanır.

Örnek IV.9.2.f.

```
MOVE 10 TO LIN.
MOVE 17 TO COL.
ACCEPT (LIN , COL) A.
```

yazılımında A değişkeninin değeri ekranın 10. satır 17. sütunundan okunur.

4. Kullanım Şekli :

ACCEPT (Satır , Sütun) ‹Değişken Adı› WITH ZERO-FILL
SPACE-FILL
LEFT-JUSTIFY
PROMPT
UPDATE
LENGHT-CHECK
AUTO-SKIP
BEEP
NO-ECHO
EMPTY-CHECK

ACCEPT deyimi ile beraber yukarıda WITH ile belirtilen isimleri de kullanabiliriz.

a) ZERO-FILL kullanıldı ise : Ekrandan okutulan değişkene 0 taşır.

b) SPACE-FILL kullanıldı ise : Ekrandan okutulan değişkene boşluk taşır.

c) LEFT-JUSTIFY : Karakter bilgilerinin sola yanaşık olarak aktarılmasını sağlar.

d) RIGHT-JUSTIFY : Karakter bilgilerin belleğe girişleri sağa dayalı olarak yapılmasını sağlar.

e) PROMPT : Değişkenle beraber kullanıldığı zaman o değişkene PIC ile ayrılan yer kadar . işaretinin yazılmasını sağlar.

f) UPDATE : Belleğe önceden girilen bilgilerin üzerinde ekleme veya silme imkanı tanır.

g) LENGTH-CHECK : Girilecek bilginin , kendisine ayrılan yer kadar girilmesini kontrol eder. Ancak PIC 'te ayrılan yer dolunca ENTER 'e izin verir.

h) EMPTY-CHECK : giriş çıkış için en az bir karakter uzunlukta bile olsa bir bilginin belleğe girişini sağlar.

ı) AUTO-SKIP : PIC de ayrılan yer dolunca ENTER'e basılmasına gerek yoktur. (PIC 'de ayrılan yer dolunca ENTER tuşuna basmadan bir sonraki işleme gidilmesini sağlar).

i) BEEP : ACCEPT ile belirlenen işlem tamamlanır tamamlanmaz düdük sesi verir.

j) NO-ECHO : Bilgi girişi sırasında bilgi yerine * (asteriks) işaretinin görüntülenmesini sağlar.

5. ACCEPT ‹Ekran Adı› ON ESCAPE.

Giriş bilgilerinin toplu olarak ekrandan girişi bir isim altında ACCEPT deyimi ile yapılır. ON ESCAPE seçeneği kullanılır ise bilgi girişini durduran ENTER 'in yaptığı işi ESC tuşu da yapar. Burada belirtilen ekran adı SCREEN SECTION bölümünde tanımlanmalıdır.

IV.10. COBOL PROGRAMLAMA DİLİNDE KULLANILAN KONTROL KOMUTLARI

IV.10.1. IF KOMUTU

Şayet belirtilen şart gerçekleşmiş ise programın belirtilen bölümüne sapma yapar.

Bir kaç kullanım şekli vardır.

1. Kullanım Şekli

IF ‹Şart Komutları Topluluğu› GO ‹Sapılacak Paragraf İsmi›.

Eğer IF ile belirtilen şart komutları topluluğu sağlanıyor ise GO ile belirtilen paragrafa git.

2. Kullanılım Şekli

IF ‹Şart Komutları Topluluğu› GO ‹Sapılacak Paragraf İsmi›
ELSE GO ‹Sapılacak Paragraf İsmi›

Eğer IF ile belirtilen Şart Komutlar topluluğu Sağlanıyor ise GO ile belirtilen paragrafa aksi takdirde ELSE ile verilen pargrafa git.

3. Kullanılım Şekli

IF ‹Şart Komutlar Topluluğu› NEXT SENTENCE ELSE ‹Şart Komutlar topluluğu›.

IF ile belirtilen Şart komutlar topluluğu sağlandığı

takdirde IF deyiminden sonraki satıra aksi takdirde ELSE ile belirtilen işlemi yap.

IV.10.2. EXIT KOMUTU

EXIT komutu her zaman bir paragraf ismi ile kullanılır ve bu paragrafta başka komut bulunmaz. Bir şarta bağlı olarak bulunduğumuz paragrafın kalan kısmını icra etmek istemiyor isek paragrafın sonuna eklenen ve içeriği yanlızca EXIT olan paragrafa atlama yapabiliriz.

IV.10.3. GO KOMUTU

Programın herhangi bir yerinden şartlı veya şartsız olarak paragraf ismi ile belirtilen yere atla. İki türlü kullanım şekli mevcuttur.

1. Kullanılım Şekli

GO ‹Sapılacak Paragraf İsmi›.

GO ile belirtilen satıra sapar.

2. Kullanılım Şekli

GO ‹Paragraf İsimleri› DEPENDING ON ‹Nümerik Değişken›.

Nümerik değişken ile belirtilen değişkenin aldığı değere göre paragraflara sapma yapar.

Örnek IV.10.a.

```
      IDENTIFICATION DIVISION.
      PROGRAM-ID. IBR.
      ENVIRONMENT DIVISION.
      DATA DIVISION.
      WORKING-STORAGE SECTION.
      77  SEC              PIC 9 VALUE 0.
      PROCEDURE DIVISION.
      BASLA.
          DISPLAY(1 1) ERASE.
          DISPLAY(10 10) "---- M E N U ---------".
          DISPLAY(11 10) "‹1›.KAYIT YAPMA.......".
          DISPLAY(12 10) "‹2›.KAYIT GÖRME.......".
          DISPLAY(13 10) "‹3›.KAYIT DEGIŞTIRME..".
          DISPLAY(14 10) "‹4›.KAYIT İPTAL.......".
          DISPLAY(15 10) "‹5›.KAYIT LISTELEME...".
          DISPLAY(16 10) "‹6›.PROGRAM SONU......".
      BASLA1.
```

```
           DISPLAY(17 10) "SEÇIMINIZ...<1>.<2>.<3>.<4>.<5>.
      -                          "<6> [.]".
           ACCEPT (17 54) SEC.
           IF SEC <1 OR SEC>6 GO BASLA1.
           GO BIR IKI UC DORT BES ALTI DEPENDING ON SEC.
       BIR.
       IKI.
       UC.
       DORT.
       BES.
       ALTI.
            STOP RUN.
```

Örnek IV.10.b.

Okutulan A , B , C sayılarından en büyüğünü A sayısına , en küçüğünü C sayısına diğerini ise B sayısına atayacak programı yapınız.

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. GULNUR.
       AUTHOR. AYŞE GULNUR AKYOL.
       DATE-WRITTEN.  09-12-1989.
       DATE-COMPILED. 11-17-1989.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       77 A        PIC 9(4) VALUE 0.
       77 B        PIC 9(4) VALUE 0.
       77 C        PIC 9(4) VALUE 0.
       77 EB       PIC 9(4) VALUE 0.
       77 ORT      PIC 9(4) VALUE 0.
       77 EK       PIC 9(4) VALUE 0.
       PROCEDURE DIVISION.
       BASLA.
           DISPLAY (1 1) ERASE.
           DISPLAY (10 20) "A SAYISINI GIRINIZ:".
           DISPLAY (12 20) "B SAYISINI GIRINIZ:".
           DISPLAY (14 20) "C SAYISINI GIRINIZ:".
           ACCEPT  (10 41) A.
```

```
           ACCEPT  (12 41) B.
           ACCEPT  (14 41) C.
           IF A > B GO BASLA1.
           IF B > C GO BASLA2.
           MOVE C TO EB.
           MOVE B TO ORT.
           MOVE A TO EK.
           GO ATAMA.
       BASLA1.
           IF A > C GO BASLA3.
           MOVE C TO EB.
           MOVE A TO ORT.
           MOVE B TO EK.
           GO ATAMA.
       BASLA2.
           MOVE B TO EB.
           IF A > C MOVE A TO ORT  MOVE C TO EK GO ATAMA.
           MOVE A TO ORT.
           MOVE C TO EK.
           GO ATAMA.
       BASLA3.
           MOVE A TO EB.
           IF B > C  MOVE B TO ORT  MOVE C TO EK GO ATAMA.
           MOVE C TO ORT.
           MOVE B TO EK.
       ATAMA.
           MOVE EB TO A.
           MOVE ORT TO B.
           MOVE EK TO C.
           DISPLAY (1 1) ERASE.
           DISPLAY (12 25) "A SAYISI =" A.
           DISPLAY (14 25) "B SAYISI =" B.
           DISPLAY (16 25) "C SAYISI =" C.
           STOP RUN.
```

IV.11. COBOL PROGRAMLAMA DILINDE ALTPROGRAMLAR

Cobol 'da alt programlar PERFORM kontrol komutu ile yapılır. PERFORM ile belirtilen paragraf veya paragraflara

giderek verilen şart sağlanıncaya kadar işlem yapar , geri döner ve PERFORM komutunu takip eden komutu icra eder.
PERFORM komutunun bir kaç türlü kullanılış şekli vardır. Bunların bazıları aşağıda belirtilmiştir.

1. Kullanım Şekli

PERFORM ‹Paragraf İsmi›.

PERFORM ile belirtilen paragrafa giderek o paragrafın sonuna kadar işlem yaptıktan sonra geri döner.

2.Kullanılım Şekli

PERFORM ‹1.Paragraf ismi› THRU ‹2.Paragraf ismi›.

PERFORM ile verilen 1.Paragraf isminden başlayarak 2.Paragraf isminin sonuna kadar işlem yaptıktan sonra geri döner.

3. Kullanılım Şekli

PERFORM ‹Paragraf ismi› VARYING I FROM X BY Y UNTIL I › N.

PERFORM ile verilen paragraf ismini I değişkeninin başlangıç dllerini X alarak Y artımla I › N oluncaya kadar tekrarlar.

4. Kullanılım Şekli

PERFORM ‹1.Paragraf ismi› THRU ‹2.Paragraf ismi› VARYING I FROM X BY Y UNTIL I › N.

PERFORM ile belirtilen 1.Paragraf isminden 2.Paragrafın sonuna kadar olan kısmı I değişkeninin başlangıç değerini X alarak Y artımla I › N oluncaya kadar tekrarlar.

5. Kullanılım Şekli

PERFORM ‹Paragraf ismi› UNTIL ‹Şart cümlesi›.

PERFORM ile belirtilen paragraf ismini UNTIL ile verilen şart cümlesi sağlanıncaya kadar tekrarlar.

IV.12. SAHALARIN KARAKTER SEVİYESİNDE İNCELENMESİ

Buradaki amaç ALFANUMERIK bir sahanın içerdiği karakterleri incelemektir. Bu inceleme 3 şekilde yapılabilir.

- verilen bir karakterin sahadaki sayısını bulmak.
- sahanın içerdiği bir karakteri değiştirmek.
- sahanın içerdiği karakterleri test ederek bazı işlemlerde bulunmak.

IV.12.1. INSPECT DEYİMİ

Verilen karakter tipindeki değişkenin sahadaki sayısını

bulmak , sahanın içerdiği bir karakteri değiştirmek yada sahanın içerdiği karakterleri test ederek bazı işlemlerde bulunmak için kullanılır. Bir kaç türlü yazım şekli vardır.

1. Kullanım Şekli

```
INSPECT [Verilen isim] TALLYING [Verilen isim]
          FOR |ALL [Literal]|BEFORE
              |LEADING      |AFTER
              |CHARACTERS
```

Örnek IV.12.1.a

SAHA = "203040" olsun.

```
INSPECT SAHA TALYING SAYAC FOR ALL "0".
```

Bu yazılımın anlamı SAHA adlı değişkendeki 0 'ların sayısını bularak SAYAC adlı değişkene atamaktır.

2. Kullanılım Şekli :

```
INSPECT [Verilen isim] REPLACING[Verilen isim] FOR
                |ALL [Literal]|BEFORE
                |LEADING      |AFTER
                              |CHARACTERS
                              |FIRST
```

SAHA = "203040" olsun.

```
INSPECT SAHA TALYING SAYAC FOR ALL "0" REPLACING FIRST 0 BY 1.
```

Bu yazılımın anlamı SAHA adlı değişkendeki 0 'ların sayısını bularak SAYAC adlı değişkene atamak ve ilk 0 'ın yerine 1 yazmaktır.

3. Kullanılım Şekli

```
INSPECT [Verilen isim] TALLYING [Verilen isim]
FOR |ALL [Literal]|BEFORE   [Literal] REPLACING
    |LEADING      |AFTER
                  |CHARACTERS
                  |FIRST
    |ALL       [Literal] [literal] |BEFORE
    |LEADING                       |AFTER
    |FIRST
    |CHARACTERS
```

Örnek IV.12.1.b.

INSPECT SAHA TALYING SAYAC FOR ALL "0" REPLACING LEADING ZERO BY SPACE.

SAHA adlı değişkendeki sıfırların sayısını SAYAC adlı değişkene atayarak soldaki belli başlı 0'ların yerine boşluk atar.

SAHA = "0010302" iken bu komut icra edilirse

SAHA = "10302" olur ve SAYAC = 4 olur.

KULLANILAN KELİMELERİN TURKÇE ANLAMI:

INSPECT : incele (sahayi)

REPLACING : değiştirerek (belirtilen sahayı)

TALLYING : sayarak (belirtilen karakteri)

ALL : tüm

LEADING : belli başli (Soldaki yan yana olan)

FIRST : ilk birinci

CHARACTERS : karakter

BEFORE : önce,önceki

AFTER : Sonra sonraki

BY : yerine

IV.12.2. STRING KOMUTU

Birden fazla sahayı bazı şartlarda birleştirmek için kullanılır.

Genel Yazılım Formatı :

‹Verilen isim veya literal1› ‹literal›
STRING DELIMITED BY INTO
‹verilen isim veya literal2› ‹SIZE›
WITH POINTER ‹Değişken › ON OVERFLOW ‹Komutlar›.

STRING deyiminden sonra verilen karakter tipindeki değişkenleri DELIMITED BY ile verilen literale kadar birleştirerek INTO 'dan sonra verilen karakter tipindeki değişkene atar. Eğer DELIMITED BY ile SIZE kullanılmış ise değişkenlerin uzunluğu kadar olan kısmı birleştirir. WITH POINTER 'den sonra verilen değişken nümerik bir değişken olup birleştirme işleminin INTO 'dan sonra verilen değişkenin kaçıncı karakterinden sonra yapılacağını belirtir. Eğer WITH POINTER kullanılmamış ise birleştirme işlemi INTO 'dan sonra verilen değişkenin 1. karakterinden

itibaren yapılır.Eğer WITH POINTER ile verilen nümerik sayı INTO'dan sonra verilen karakter değişkenin uzunluğundan büyük ise ON OVERFLOW ile verilmiş komutlar topluluğu icra edilir.

Burada kullanılan kelimelerin türkçe anlamı aşağıdaki gibidir.

STRING : Dizi,katar
DELIMITED : Limitli,ayrılmış,
BY : Tarafından,
SIZE : Uzunluk,
INTO : İçine,
POINTER : İşaret edici,gösterici,
OVERFLOW : Taşma,

Örnek IV.12.2.a.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. ESRA.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77 GS          PIC X(10) VALUE "1.HAZIRAN ".
77 ES          PIC X(5)  VALUE "1989 ".
77 GG          PIC X(14) VALUE "PERŞEMBE GÜNÜ ".
77 GA          PIC X(23) VALUE "BIR GERÇEK ORTAYA ÇIKTI".
77 SS          PIC X(52) VALUE SPACES.
PROCEDURE DIVISION.
BASLA.
    DISPLAY(1 1) ERASE.
    STRING GS ES GG GA DELIMITED BY SIZE INTO SS.
    DISPLAY(17 10) SS.
    STOP RUN.
```

Bu örnekte ele alınan GS ES GG GA değişkenleri birleştirilerek SS adlı değişkene atanmaktadır.

IV.12.3. UNSTRING KOMUTU

Verilen bir sahanın içerigini başka sahalara ayırmak için kullanılabilir.

Genel Yazılım Formatı :

UNSTRING‹Verilen isim›DELIMITED BY ALL‹Verilen isim,literal›INT
OR ALL
‹Verilen isim›DELIMITER IN‹Verilen isim›COUNT IN‹Verilen isim›
WITH POINTER‹Verilen isim›TALLYING IN‹Verilen isim›
ON OVERFLOW‹komutlar›.

Bu yazılış şeklinde UNSTRING 'den sonra verilen isim parçalara ayrılacak olan karakter bilgi değişkenidir. DELIMITED BY ile verilen karakter bilgi ise parçalanacak sahanın bir karakteri olmalıdır. Çünkü parçalanacak sahanın DELIMITED BY ile verilmiş kısmı INTO 'dan sonra verilecek karakter tipindeki değişkene aktarılacaktır.

Örnek IV.12.3.a.

Parçalanacak saha GS="8.KASIM 1987*TARİHİ*ESRA'NIN*DOĞDUĞU TARİHTİR" olsun. Bunu 4 parçaya ayırmak isteyelim. Bunu gerçekleştirmek için :

UNSTRING GS DELIMITED BY "*" INTO AS ES IS OS.

Yazılımına göre parçalanacak olan GS sahasının ilk * olan karakterine kadar olan kısmı AS adlı değişkene, bundan itibaren diğer * karakterine kadar olan kısmı ES adlı değişkene , bundan itibaren diğer * karakterine kadar olan kısmı IS adlı değişkene , kalan diğer kısım ise OS adlı değişkene aktarılır.

Bu işlemin sonucunda GS adlı saha ;

AS="8.KASIM 1987",ES="TARİHİ" , IS="ESRA'NIN" ,
OS="DOĞDUĞU TARİHTİR"

şeklinde parçalanır.

WITH POINTER ile tanımlanmış tamsayı değişken ise parçalanacak sahanın parçalama işlemine kaçıncı karakterinden itibaren başlanacağını belirtmek içindir. Kullanılmadığı zaman 1 değerini alır.

TALLYING ile belirtilen tamsayı değişken ise ; UNSTRING ile parçalanacak sahaların kaç parçaya ayrıldığını belirtir.

COUNT IN ile belirtilen tamsayı değişken ise INTO ile verilen değişkenin karakter sayısını verir.

Kullanılan kelimelerin türkçe anlamları :

DELIMITED : Ayıraç,
DELIMITER IN ‹Verilen isim› : Ayıracı içerecek saha,

COUNT IN ‹verilen isim› : Kaç karakterin taşınacağını
                           bildirir.
TALLYING IN ‹Verilen isim› : İlk değer+saha sayısı,
POINTER : İlk değer+parçalara ayrılacak
           sahanın o anki pozisyonu.

Örnek IV.12.3.b.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. ESRA.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 77 GS PIC X(35) VALUE "8.KASIM 1987*ESRA'NIN*DOĞUM*GÜNÜDÜR".
 77 AS PIC X(12) VALUE SPACES.
 77 ES PIC X(8)  VALUE SPACES.
 77 US PIC X(5)  VALUE SPACES.
 77 OS PIC X(7)  VALUE SPACES.
 77 S0 PIC 99    VALUE 0.
 77 S1 PIC 99    VALUE 0.
 77 S2 PIC 99    VALUE 0.
 77 S3 PIC 99    VALUE 0.
 77 S4 PIC 99    VALUE 0.
 PROCEDURE DIVISION.
 BASLA.
     DISPLAY(1 1) ERASE.
     UNSTRING GS DELIMITED BY "*" INTO AS COUNT IN S0
                                       ES COUNT IN S1
                                       US COUNT IN S2
                                       OS COUNT IN S3
                                          TALLYING S4.
     DISPLAY(12 17) "PARÇALANACAK SAHA.....: " GS.
     DISPLAY(15 10) "PARÇALANAN 1.SAHA ....: " AS.
     DISPLAY(15 50) "1.SAHANIN UZUNLUĞU....: " S0.
     DISPLAY(17 10) "PARÇALANAN 2.SAHA ....: " ES.
     DISPLAY(17 50) "2.SAHANIN UZUNLUĞU....: " S1.
     DISPLAY(19 10) "PARÇALANAN 3.SAHA ....: " US.
     DISPLAY(19 50) "3.SAHANIN UZUNLUĞU....: " S2.
     DISPLAY(21 10) "PARÇALANAN 4.SAHA ....: " OS.
```

```
      DISPLAY(21 50) "4.SAHANIN UZUNLUGU....: " S3.
      DISPLAY(23 17) "PARÇALANAN SAHA SAYISI....: " S4.
      STOP RUN.
```

Örnek IV.12.3.c.

Ekrandan okutulacak 1901-2100 tarihleri arasındaki herhangi bir tarihin haftanın hangi gününe rastladığını bulacak ve istenilen yılın takvimini yapacak bir COBOL programı yapınız.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. SABIHA.
 AUTHOR. GULNUR AKYOL.
 DATE-WRITTEN. 09-15-1989.
 DATE-COMPILED. 09-20-1989.
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT SEMA ASSIGN TO PRINTER.
 DATA DIVISION.
 FILE SECTION.
 FD SEMA LABEL RECORD IS OMITTED.
 01 YAZ1 PIC X(80).
 WORKING-STORAGE SECTION.
 01 MATRIS.
    02 SA OCCURS 21 TIMES.
       03 S OCCURS 24 TIMES PIC 99.
 01 GUNLER.
    02 F        PIC X(9) VALUE "PAZAR    ".
    02 F        PIC X(9) VALUE "PAZARTESI".
    02 F        PIC X(9) VALUE "SALI     ".
    02 F        PIC X(9) VALUE "ÇARŞAMBA ".
    02 F        PIC X(9) VALUE "PERŞEMBE ".
    02 F        PIC X(9) VALUE "CUMA     ".
    02 F        PIC X(9) VALUE "CUMARTESI".
 01 GUNL REDEFINES GUNLER.
    02 G OCCURS 7 TIMES PIC X(9).
 01 KISAGUN.
    02 F        PIC X(4) VALUE "PAZ.".
```

```
    02 F          PIC X(4) VALUE "PZT.".
    02 F          PIC X(4) VALUE "SAL.".
    02 F          PIC X(4) VALUE "CAR.".
    02 F          PIC X(4) VALUE "PER.".
    02 F          PIC X(4) VALUE "CUM.".
    02 F          PIC X(4) VALUE "CMT.".
01 KIS REDEFINES KISAGUN.
    02 ISP OCCURS 7 TIMES PIC X(4).
01 AYLAR.
    02 F          PIC 9(2) VALUE 31.
    02 F          PIC 9(2) VALUE 28.
    02 F          PIC 9(2) VALUE 31.
    02 F          PIC 9(2) VALUE 30.
    02 F          PIC 9(2) VALUE 31.
    02 F          PIC 9(2) VALUE 30.
    02 F          PIC 9(2) VALUE 31.
    02 F          PIC 9(2) VALUE 31.
    02 F          PIC 9(2) VALUE 30.
    02 F          PIC 9(2) VALUE 31.
    02 F          PIC 9(2) VALUE 30.
    02 F          PIC 9(2) VALUE 31.
01 AYL REDEFINES AYLAR.
    02 A OCCURS 12 TIMES PIC 9(2).
77 AGA            PIC ZZ   VALUE ZEROS.
77 AGA1           PIC ZZZ  VALUE ZEROS.
77 N3             PIC 99   VALUE 0.
77 N4             PIC 99   VALUE 0.
77 KAL4           PIC 9    VALUE 0.
77 KAL7           PIC 9    VALUE 0.
77 TAM4           PIC 9(3) VALUE 0.
77 TAM7           PIC 9(3) VALUE 0.
77 AY             PIC 9(2) VALUE 0.
77 GUN            PIC 9(2) VALUE 0.
77 YIL            PIC 9(4) VALUE 0.
77 SEC            PIC 9    VALUE 0.
77 K4             PIC 99V99 VALUE 0.
77 TYIL           PIC 9(4) VALUE 0.
```

```
77 K7          PIC 99V99 VALUE 0.
77 I           PIC 9(4) VALUE 0.
77 GUL         PIC 99   VALUE 0.
77 GUL1        PIC 99   VALUE 0.
77 T           PIC 9(3) VALUE 0.
77 T1          PIC 9(2)V99 VALUE 0.
77 T2          PIC 9(2) VALUE 0.
77 T17         PIC 9(2) VALUE 0.
77 SEC1        PIC X    VALUE SPACE.
77 L           PIC 99   VALUE 0.
77 I1          PIC 99   VALUE 0.
77 G1          PIC 99   VALUE 0.
77 S1          PIC 99   VALUE 0.
77 S2          PIC 99   VALUE 0.
77 S3          PIC 99   VALUE 0.
77 K           PIC 99   VALUE 0.
77 J           PIC 99   VALUE 0.
77 N           PIC 99   VALUE 0.
77 N1          PIC 99   VALUE 0.
77 R           PIC 99   VALUE 0.
77 L1          PIC 99   VALUE 0.
77 M           PIC 99   VALUE 0.
77 ESRA        PIC X(9) VALUE SPACES.
77 CIZGI       PIC X(80) VALUE ALL "-".
77 BOSLUK      PIC X(80) VALUE ALL " ".
77 BOS         PIC X(75) VALUE ALL " ".
77 GS          PIC X(3) VALUE SPACES.
77 GS1         PIC X(80) VALUE  SPACES.
77 GS2         PIC X     VALUE SPACE.
77 SAYAC       PIC 9(3) VALUE 0.
77 FTUS        PIC X(2).
    88 ESC      VALUE "01".
    88 F1       VALUE "02".
    88 F2       VALUE "03".
    88 F3       VALUE "04".
    88 F4       VALUE "05".
```

```
01 HASHASI.
    02 F          PIC X(20) VALUE "    ------OCAK-----  ".
    02 F          PIC X(20) VALUE "    ------ŞUBAT----  ".
    02 F          PIC X(20) VALUE "    ------MART-----  ".
    02 F          PIC X(20) VALUE "    ------NİSAN----  ".
01 GOLCUK.
    02 F          PIC X(20) VALUE "    -----MAYIS-----  ".
    02 F          PIC X(20) VALUE "    ----HAZİRAN----  ".
    02 F          PIC X(20) VALUE "    -----TEMMUZ----  ".
    02 F          PIC X(20) VALUE "    ----AGUSTOS----  ".
01 SARIGOL.
    02 F          PIC X(20) VALUE "    -----EYLUL-----  ".
    02 F          PIC X(20) VALUE "    ------EKİM-----  ".
    02 F          PIC X(20) VALUE "    ------KASIM----  ".
    02 F          PIC X(20) VALUE "    -----ARALIK----  ".
01 CUKURKUYU.
    02 EMI        PIC X(4) VALUE SPACES.
    02 F          PIC X    VALUE SPACES.
    02 BBB        PIC X(72) VALUE SPACES.
    02 F          PIC X(3) VALUE SPACES.
01 VARTANSA.
    02 F          PIC X(32) VALUE SPACES.
    02 YIL-GS     PIC Z(4)  VALUE ZEROS.
    02 F          PIC X(14) VALUE "  YILI TAKVİMİ".
    02 F          PIC X(30) VALUE SPACES.
SCREEN SECTION.
01 ESR-1.
    02 LINE 8 COLUMN 26 VALUE "┌".
    02 COLUMN 27 PIC X(26) USING CIZGI.
    02 COLUMN 53 VALUE "┐".
    02 LINE 21 COLUMN 26 VALUE "└".
    02 COLUMN 27 PIC X(26) USING CIZGI.
    02 COLUMN 53 VALUE "┘".
01 ESR-2.
    02 LINE 10 COLUMN 5 VALUE "┌".
    02 COLUMN 6 PIC X(70) USING CIZGI.
    02 COLUMN 75 VALUE "┐".
```

```
    02 LINE 11 COLUMN 5 VALUE "|".
    02 COLUMN 75 VALUE "|".
    02 LINE 12 COLUMN 5 VALUE "└".
    02 COLUMN 6 PIC X(70) USING CIZGI.
    02 COLUMN 75 VALUE "┘".
    02 LINE 11 COLUMN 10 VALUE "1.OCAK TARIHINI ÖĞRENMEK
-   "ISTEDIĞINIZ YILI GIRINIZ.....:".
 01 ESR-3.
    02 LINE 11 COLUMN 68 PIC Z(4) USING YIL UNDERLINE AUTO.
 01 ESR-4.
    02 LINE 11 COLUMN 10 PIC X(63) USING BOSLUK.
 01 ESR-5.
    02 LINE 11 COLUMN 10 VALUE "1.OCAK".
    02 LINE 11 COLUMN 20 PIC 9(4) USING YIL
                         UNDERLINE BLINK REVERSE-VIDEO.
    02 COLUMN 30 VALUE "TARIHI".
    02 COLUMN 42 PIC X(9) USING ESRA
                         UNDERLINE BLINK REVERSE-VIDEO.
    02 COLUMN 56 VALUE "DIR.".
 01 ESR-6.
    02 ESR-6A.
       03 LINE 19 COLUMN 5 VALUE "┌".
       03 COLUMN 6 PIC X(70) USING CIZGI.
       03 COLUMN 75 VALUE "┐".
       03 LINE 20 COLUMN 5 VALUE "|".
       03 COLUMN 75 VALUE "|".
       03 LINE 21 COLUMN 5 VALUE "└".
       03 COLUMN 6 PIC X(70) USING CIZGI.
       03 COLUMN 75 VALUE "┘".
    02 ESR-6B.
       03 LINE 20 COLUMN 17 BLINK REVERSE-VIDEO
     VALUE "DEVAM ETMEK ISTIYORMUSUNUZ  [E/H].....[.]".
       03 COLUMN 59 VALUE "." BLINK.
    02 ESR-6B1.
       03 LINE 20 COLUMN 33 VALUE "<ESC> ÇIKIŞ".
       03 COLUMN 34 VALUE "ESC" REVERSE-VIDEO BLINK.
```

```
01 ESR-6C.
    02 ESR-6C1.
       03 LINE 20 COLUMN 17 BLINK REVERSE-VIDEO
     VALUE "HATALI TARIH GIRILDI....! LUTFEN DUZELTINIZ...!".
    02 ESR-6C2.
       03 LINE 20 COLUMN 12 PIC X(63) USING BOSLUK.
01 ESR-7.
    02 LINE 20 COLUMN 59 PIC X USING  SEC1 BLINK AUTO.
01 ESR-8.
    02 LINE 7 COLUMN 19 VALUE "┌".
    02 COLUMN 20 PIC X(40) USING CIZGI.
    02 COLUMN 60 VALUE "┐".
    02 LINE 8 COLUMN 19 VALUE "│".
    02 COLUMN 60 VALUE "│".
    02 LINE 9 COLUMN 19 VALUE "│".
    02 COLUMN 60 VALUE "│".
    02 LINE 10 COLUMN 19 VALUE "│".
    02 COLUMN 60 VALUE "│".
    02 LINE 11 COLUMN 19 VALUE "│".
    02 COLUMN 60 VALUE "│".
    02 LINE 12 COLUMN 19 VALUE "└".
    02 COLUMN 20 PIC X(40) USING CIZGI.
    02 COLUMN 60 VALUE "┘".
01 ESR-9.
    02 LINE 15 COLUMN 5 VALUE "┌".
    02 COLUMN 6 PIC X(70) USING CIZGI.
    02 COLUMN 75 VALUE "┐".
    02 LINE 16 COLUMN 5 VALUE "│".
    02 COLUMN 75 VALUE "│".
    02 LINE 17 COLUMN 5 VALUE "└".
    02 COLUMN 6 PIC X(70) USING CIZGI.
    02 COLUMN 75 VALUE "┘".
01 ESR-10.
    02 LINE 16 COLUMN 9  PIC Z(2) USING GUN BLINK.
    02 COLUMN 12 VALUE "/".
    02 COLUMN 14 PIC Z(2) USING AY BLINK.
    02 COLUMN 17 VALUE "/".
```

```
    02 COLUMN 20 PIC Z(4) USING YIL BLINK.
    02 COLUMN 30 VALUE "TARİHİ".
    02 COLUMN 38 PIC X(9) USING ESRA BLINK REVERSE-VIDEO.
    02 COLUMN 50 VALUE "GÜNÜNE RASTLAMAKTADIR.".
01 ESR-11.
    02 ESR-11A.
     03 LINE 25 COLUMN 1 VALUE "MESAJ....!" UNDERLINE BLINK
    02 ESR-11B.
     03 LINE 25 COLUMN 15 VALUE "‹F1› YIL DEĞIŞ" UNDERLINE.
     03 COLUMN 16 VALUE "F1" REVERSE-VIDEO.
     03 COLUMN 35 VALUE "‹F2› YAZICI" UNDERLINE.
     03 COLUMN 36 VALUE "F2" REVERSE-VIDEO.
     03 COLUMN 50 VALUE "‹F3› ANA MENÜ" UNDERLINE.
     03 COLUMN 51 VALUE "F3" REVERSE-VIDEO.
     03 COLUMN 65 VALUE "SEÇİMİNİZ...[.]" UNDERLINE.
     03 COLUMN 78 VALUE "." BLINK.
    02 ESR-11C.
     03 LINE 25 COLUMN 15 PIC X(65) USING BOSLUK.
    02 ESR-11D.
     03 LINE 25 COLUMN 78 PIC X USING SEC1 AUTO.
    02 ESR-11E.
     03 LINE 25 COLUMN 15 "TAKVİMİNİ DÖKMEK İSTEDİĞİNİZ
-     " YILI GİRİNİZ....! [....] " UNDERLINE BLINK.
     03 COLUMN 64 VALUE "...." UNDERLINE BLINK.
     03 COLUMN 69 VALUE "‹ESC› ÇIKIŞ".
     03 COLUMN 70 VALUE "ESC" REVERSE-VIDEO.
    02 ESR-11F.
     03 LINE 25 COLUMN 64 PIC Z(4) USING YIL
                                    UNDERLINE AUTO.
    02 ESR-11G.
     03 LINE 25 COLUMN 15 "HATALI TARİH GİRİLDİ LÜTFEN
-     "TARİHİ DÜZELTİNİZ.....!" UNDERLINE BLINK.
    02 ESR-11H.
     03 LINE 25 COLUMN 15 VALUE "HESAPLAMA YAPILIYOR....!
-        "LÜTFEN BEKLEYİNİZ.....!" UNDERLINE BLINK.
01 ESR-12.
    02 LINE 1 COLUMN 1 PIC Z(4) USING YIL UNDERLINE BLINK.
```

```
    02 COLUMN 76 PIC Z(4) USING YIL UNDERLINE BLINK.
    02 LINE 9 COLUMN 1 PIC Z(4) USING YIL UNDERLINE BLINK.
    02 COLUMN 76 PIC Z(4) USING YIL UNDERLINE BLINK.
    02 LINE 17 COLUMN 1 PIC Z(4) USING YIL UNDERLINE BLINK.
    02 COLUMN 76 PIC Z(4) USING YIL UNDERLINE BLINK.
01 YAZ.
    02 LINE 9  COLUMN  24 VALUE "GUN.......: ".
    02 LINE 10 COLUMN  24 VALUE "AY........: ".
    02 LINE 11 COLUMN  24 VALUE "YIL.......: ".
01 OKU.
    02 LINE 9  COLUMN 37 PIC Z(2) USING GUN UNDERLINE AUTO.
    02 LINE 10 COLUMN 37 PIC Z(2) USING AY UNDERLINE AUTO.
    02 LINE 11 COLUMN 37 PIC Z(4) USING YIL UNDERLINE AUTO.
01 MENU.
    02 LINE 10 COLUMN 29 VALUE "********M E N U********" BLINK
                          REVERSE-VIDEO.
    02 LINE 12 COLUMN 28 VALUE "<1>..YILIN ILK GUNU.....".
    02 LINE 14 COLUMN 28 VALUE "<2>..HERHANGI BIR GUN...".
    02 LINE 16 COLUMN 28 VALUE "<3>..TAKVIMI LISTELEME..".
    02 LINE 18 COLUMN 28 VALUE "<4>..PROGRAM SONU.......".
    02 LINE 20 COLUMN 28 VALUE "SECIMINIZ 1,2,3,4....<.>"
                       REVERSE-VIDEO.
    02 LINE 20 COLUMN 50 VALUE "." BLINK REVERSE-VIDEO.
01 SECOKU.
    02 LINE 20 COLUMN 50 PIC Z USING SEC
                    REVERSE-VIDEO BLINK AUTO.
01 TAKVIM.
    02 LINE 1 COLUMN 7 REVERSE-VIDEO
       VALUE "|    OCAK       | ".
    02 COLUMN 24  REVERSE-VIDEO
       VALUE "|   SUBAT       |".
    02 COLUMN 42  REVERSE-VIDEO
       VALUE "|   MART        | ".
    02 COLUMN 59 REVERSE-VIDEO
       VALUE "|   NISAN       | ".
    02 LINE 9 COLUMN 7 REVERSE-VIDEO
       VALUE "|   MAYIS       | ".
```

```
    02 COLUMN 24  REVERSE-VIDEO
         VALUE "|  HAZIRAN        |".
    02 COLUMN 42  REVERSE-VIDEO
         VALUE "|  TEMMUZ         | ".
    02 COLUMN 59 REVERSE-VIDEO
         VALUE "|    AGUSTOS     | ".
    02 LINE 17 COLUMN 7 REVERSE-VIDEO
         VALUE "|    EYLUL       | ".
    02 COLUMN 24  REVERSE-VIDEO
         VALUE "|   EKIM         |".
    02 COLUMN 42  REVERSE-VIDEO
         VALUE "|   KASIM        | ".
    02 COLUMN 59 REVERSE-VIDEO
         VALUE "|    ARALIK      | ".
01 GUNTAK.
    02 LINE 2  COLUMN 1 VALUE "PZR." REVERSE-VIDEO.
    02 LINE 3  COLUMN 1 VALUE "PZT." REVERSE-VIDEO.
    02 LINE 4  COLUMN 1 VALUE "SAL." REVERSE-VIDEO.
    02 LINE 5  COLUMN 1 VALUE "ÇAR." REVERSE-VIDEO.
    02 LINE 6  COLUMN 1 VALUE "PRŞ." REVERSE-VIDEO.
    02 LINE 7  COLUMN 1 VALUE "CUM." REVERSE-VIDEO.
    02 LINE 8  COLUMN 1 VALUE "CMT." REVERSE-VIDEO.
    02 LINE 10 COLUMN 1 VALUE "PZR." REVERSE-VIDEO.
    02 LINE 11 COLUMN 1 VALUE "PZT." REVERSE-VIDEO.
    02 LINE 12 COLUMN 1 VALUE "SAL." REVERSE-VIDEO.
    02 LINE 13 COLUMN 1 VALUE "ÇAR." REVERSE-VIDEO.
    02 LINE 14 COLUMN 1 VALUE "PRŞ." REVERSE-VIDEO.
    02 LINE 15 COLUMN 1 VALUE "CUM." REVERSE-VIDEO.
    02 LINE 16 COLUMN 1 VALUE "CMT." REVERSE-VIDEO.
    02 LINE 18 COLUMN 1 VALUE "PZR." REVERSE-VIDEO.
    02 LINE 19 COLUMN 1 VALUE "PZT." REVERSE-VIDEO.
    02 LINE 20 COLUMN 1 VALUE "SAL." REVERSE-VIDEO.
    02 LINE 21 COLUMN 1 VALUE "ÇAR." REVERSE-VIDEO.
    02 LINE 22 COLUMN 1 VALUE "PRŞ." REVERSE-VIDEO.
    02 LINE 23 COLUMN 1 VALUE "CUM." REVERSE-VIDEO.
    02 LINE 24 COLUMN 1 VALUE "CMT." REVERSE-VIDEO.
```

```
PROCEDURE DIVISION.
BASLA.
    DISPLAY(1 1) ERASE.
    DISPLAY ESR-1.
    PERFORM IBR-1 VARYING LIN FROM 9 BY 1 UNTIL LIN > 20.
    DISPLAY MENU.
BASLA1.
    MOVE 0 TO SEC.
    ACCEPT SECOKU.
    GO BIR IKI UC DORT DEPENDING ON SEC.
    GO BASLA1.
BIR.
    PERFORM IBR-2.
BIR1.
    MOVE " " TO SEC1.
    DISPLAY ESR-2 ESR-6A ESR-6B1. PERFORM IBR-4.
    ACCEPT ESR-3.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF ESC DISPLAY ESR-6C2 GO BASLA.
    IF YIL < 1901 OR YIL > 2099
                DISPLAY ESR-6C1 PERFORM IBR-4
                PERFORM IBR-5 DISPLAY ESR-6C2 GO BIR1.
    PERFORM HESAP THRU ESR1.
    MOVE G(GUL1) TO ESRA.
    DISPLAY ESR-4 ESR-5.
    DISPLAY ESR-6B ESR-7.
    ACCEPT ESR-7.
    IF SEC1 = "E" OR = "e"
            PERFORM IBR-4 DISPLAY ESR-6C2 GO BIR1.
    GO BASLA.
IKI.
    PERFORM IBR-2.
IKI1.
    MOVE SPACES TO ESRA.
    PERFORM IBR-4.
    DISPLAY ESR-8 YAZ ESR-9 ESR-10 ESR-6A.
```

```
IKI2.
    DISPLAY OKU ESR-6B1. ACCEPT OKU.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF ESC DISPLAY ESR-6C2 GO BASLA.
    COMPUTE K4 = YIL / 4.
    MOVE K4 TO TAM4.
    COMPUTE KAL4 = YIL - TAM4 * 4.
    IF KAL4 = 0 MOVE 29 TO A(2) ELSE MOVE 28 TO A(2).
    IF GUN = 0 OR > A(AY) OR AY > 12 OR YIL <1901 OR > 2099
                    DISPLAY ESR-6C1 PERFORM IBR-4
                    PERFORM IBR-5 DISPLAY ESR-6C2 GO IKI2.
    PERFORM HESAP THRU ESR1.
    MOVE G(GUL) TO ESRA. MOVE " " TO SEC1.
    DISPLAY ESR-10.
    DISPLAY ESR-6B ESR-7.
    ACCEPT ESR-7.
    IF SEC1 = "E" OR = "e"
              PERFORM IBR-4 DISPLAY ESR-6C2 GO IKI1.
    GO BASLA.
UC.
    PERFORM IBR-2.
    DISPLAY ESR-11A.
    DISPLAY TAKVIM GUNTAK.
ERZ.
    DISPLAY ESR-11C ESR-11E. MOVE ZEROS TO YIL.
    DISPLAY ESR-11F.
    ACCEPT ESR-11F.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF ESC GO BASLA.
    IF YIL < 1901 OR YIL > 2099
           DISPLAY ESR-11C ESR-11G PERFORM IBR-5 GO ERZ.
    DISPLAY ESR-11C ESR-11H ESR-12.
    PERFORM IBR-11 VARYING I FROM 1 BY 1 UNTIL I > 21.
    PERFORM HESAP THRU ESR1.
    MOVE GUL1 TO G1 S2.
    PERFORM UC2 VARYING I FROM 1 BY 1 UNTIL I > 12.
    PERFORM M1 VARYING M FROM 1 BY 1 UNTIL M > 3.
```

```
ERZIN.
    MOVE " " TO SEC1.
    DISPLAY ESR-11C ESR-11B ESR-11D.
    ACCEPT ESR-11D.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF F1 MOVE ZEROS TO YIL DISPLAY ESR-12
                                    PERFORM IBR-7 GO ERZ.
    IF F2 GO ERZYAZ.
    IF F3 GO BASLA.
    GO ERZIN.
ERZYAZ.
    OPEN OUTPUT SEMA.
    MOVE YIL TO YIL-GS.
    WRITE YAZ1 FROM VARTANSA.
    MOVE 0 TO T17.
    PERFORM IBR-9A VARYING M FROM 1 BY 1 UNTIL M > 3.
    CLOSE SEMA.
    GO ERZIN.
DORT.
    STOP RUN.
HESAP.
    COMPUTE K4 = YIL / 4.
    MOVE K4 TO TAM4.
    COMPUTE KAL4 = YIL - TAM4 * 4.
    IF KAL4 = 0   MOVE YIL TO TYIL GO ARTIK.
    COMPUTE TYIL = YIL - KAL4.
ARTIK.
    COMPUTE K7 = TYIL / 7.
    MOVE K7 TO TAM7.
    COMPUTE KAL7 = TYIL - TAM7 * 7.
    IF KAL7 = 0 MOVE 6 TO GUL GO A1.
    IF KAL7 = 1 MOVE 2 TO GUL GO A1.
    IF KAL7 = 2 MOVE 5 TO GUL GO A1.
    IF KAL7 = 3 MOVE 1 TO GUL GO A1.
    IF KAL7 = 4 MOVE 4 TO GUL GO A1.
    IF KAL7 = 5 MOVE 7 TO GUL GO A1.
    MOVE 3 TO GUL.
```

```
A1.
    MOVE 0 TO T.
    IF KAL4 = 0  MOVE 29 TO A(2) GO ESR1.
    MOVE 28 TO A(2).
    COMPUTE GUL = GUL + KAL4 + 1.
    IF GUL > 7 COMPUTE GUL = GUL - 7.
ESR1.
    MOVE GUL TO GUL1.
    PERFORM ESR2 VARYING I FROM 1 BY 1 UNTIL I > AY - 1.
    COMPUTE T = T + GUN.
    COMPUTE T1 = T / 7.
    MOVE T1 TO T2.
    COMPUTE GUL = GUL + T - T2 * 7 - 1.
    IF GUL = 0 MOVE 7 TO GUL.
    IF GUL > 7 COMPUTE GUL = GUL - 7.
ESR2.
    COMPUTE T = T + A(I).
UC2.
    MOVE 0 TO L.
    COMPUTE I1  = (I - 1) / 4.
    COMPUTE N = I - I1 * 4.
    COMPUTE N1 = (N - 1) * 6 + 1.
    COMPUTE N4 = (I - 1) / 4.
    COMPUTE N3 = 1 + N4 * 7.
    COMPUTE G1 = G1 + N3 - 1.
    PERFORM UC3 VARYING J FROM N1 BY 1 UNTIL J > N * 6.
    COMPUTE S1 = A(I) /7.
    COMPUTE S2 = S2 + A(I) - S1 *7.
    IF S2 > 7 COMPUTE S2 = S2 - 7.
    MOVE S2 TO G1.
UC3.
    PERFORM UC4 VARYING K FROM G1 BY 1 UNTIL K > N3 + 6.
    MOVE N3 TO G1.
UC4.
    ADD 1 TO L.
    IF L > A(I)  MOVE 0 TO S(K , J)
                          ELSE  MOVE L TO S(K , J).
```

```
M1.
    MOVE 3 TO K.
    COMPUTE R = 1 + (M - 1) * 7.
    PERFORM M2 VARYING N FROM 1 BY 1 UNTIL N > 4.
M2.
    COMPUTE L = (M - 1) * 9.
    COMPUTE S3 = 1 + (N - 1) * 6.
    IF L = 0  MOVE 1 TO L.
    IF L = 18 MOVE 17 TO L.
    MOVE L TO L1.
    PERFORM M3 VARYING J FROM S3 BY 1 UNTIL J > S3 + 5.
M3.
    COMPUTE K = K + 3.
    PERFORM M4 VARYING I FROM R BY 1 UNTIL I > R + 6.
    MOVE L1 TO L.
M4.
    COMPUTE L = L + 1.
    MOVE L TO LIN. MOVE K TO COL. MOVE S(I , J) TO AGA.
    DISPLAY (LIN , COL) AGA.
IBR-1.
    DISPLAY (LIN , 26) "|"
            (LIN , 53) "|".
IBR-2.
    PERFORM IBR-3 VARYING LIN FROM 3 BY 1 UNTIL LIN >24.
IBR-3.
    DISPLAY(LIN , 1) BOSLUK.
IBR-4.
    MOVE ZEROS TO GUN AY YIL.
IBR-5.
    PERFORM IBR-6 VARYING I FROM 1 BY 1 UNTIL I > 500.
IBR-6.
IBR-7.
    PERFORM IBR-8 VARYING LIN FROM 2 BY 1 UNTIL LIN > 24.
IBR-8.
    IF NOT (LIN = 9 OR = 17) DISPLAY(LIN , 5) BOS.
IBR-9A.
    MOVE SPACES TO BBB EMI.
```

```
    WRITE YAZ1 FROM CUKURKUYU.
    IF M = 1 WRITE YAZ1 FROM HASHASI.
    IF M = 2 WRITE YAZ1 FROM GOLCUK.
    IF M = 3 WRITE YAZ1 FROM SARIGOL.
    COMPUTE  S3 = (M - 1) * 7 + 1.
    PERFORM IBR-9 VARYING I FROM S3 BY 1 UNTIL I > M *7.
IBR-9.
    MOVE SPACES TO BBB GS1.
    PERFORM IBR-10 VARYING J FROM 1 BY 1 UNTIL J > 24.
    INSPECT BBB REPLACING ALL "-" BY " ".
    INSPECT BBB REPLACING ALL "00" BY "  ".
    ADD 1 TO T17.
    IF T17 > 7 MOVE 1 TO T17.
    MOVE ISP(T17) TO EMI.
    WRITE YAZ1 FROM CUKURKUYU.
IBR-10.
    MOVE S(I , J) TO GS.
    STRING GS1 DELIMITED BY SPACE GS
               DELIMITED BY SPACE "-"
               DELIMITED BY SIZE INTO BBB.
    MOVE BBB TO GS1.
IBR-11.
     PERFORM IBR-12 VARYING J FROM 1 BY 1 UNTIL J > 24.
IBR-12.
     MOVE 00 TO S(I , J).
```

IV.13. COBOL PROGRAMLAMA DİLİNDE DOSYALAR

IV.13.1. Cobol Programlama Dilinde Dosya Düzenleme Şekilleri

Cobol programlama dilinde dosya düzenleme şekilleri üç şekilde olur. Bunlar ;

1) SEQENTIAL (Sıralı) Dosyalar ,
2) INDEXED Dosyalar ,
3) RELATIVE Dosyalar .

IV.13.2. Cobol Programlama Dilinde Dosyaya Erişim Şekilleri

Cobol programlama dilinde dosyaya erişim şekilleri üç şekilde olur. Bunlar ;

1) SEQENTIAL (sıralı) Erişim ,
2) RANDOM (rastgele) Erişim ,
3) INDEXED Erişim (Bu şekildeki erişim şekli hem sıralı hemde rastgele olabilir.)

IV.13.3. SEQUENTIAL DOSYALAR

Sequential dosyalar da kayıtlar sıra ile yapılır. Sequential bir dosyaya erişim şekli yine Sequential olmalıdır. Dosya düzenleme şekli Sequential olan bir dosyaya erişim , random (rastgele) veya indexed (Hem sıralı hemde rastgele) olamaz.

IV.13.3.1. Sequential Dosyalarda Dosya Açılım Şekilleri

1. INPUT modu :Sadece dosyadan bilgi okumak için kullanılır ve yanlızca okuyucu kafa çalışır. Dosyanın INPUT modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.
2. OUTPUT modu :Sadece dosyaya bilgi yazmak için kullanılır ve yanlızca yazıcı kafa çalışır. Dosya diskette ilk defa açıldığı zaman bu modda açılmalıdır. OUTPUT modunda açılmış olan bir dosyada daha önceden bulunan bütün kayıtlar silinir. Bundan dolayı dosyanın OUTPUT modunda açılırken çok dikkatli olunması gerekir. Örnek olarak içerisinde 20 kayıt bulunan bir dosya OUTPUT modda açılacak olur ise daha önceden yapılmış olan 20 kayıt silinir. Çünkü bir dosya OUTPUT modda açılıyor ise yazıcı kafa ilk kayıdın üzerine konumlanır böylece daha önce yapılmış bulunan tüm kayıtlar yok olur.
3. I-O (INPUT-OUTPUT) modu :dosyadan bilgi okuma ve

dosyaya bilgi yazmak için kullanılır. Okuyucu ve yazıcı kafa beraber çalışır. Dosyanın bu modda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.

4. EXTEND modu :Sequential dosyalarda kayıt ilavesi yapmak için kullanılır. Sadece yazıcı kafa çalışır. Herhangi bir dosya bu mod ile açıldığında yazıcı kafa daha önceden yapılmış olan kayıtlardan sonraki kayıda konumlanır. Örnek olarak bir 16 kayıt bulunan dosya EXTEND modunda açılmış ise yazıcı kafa 17. kayıt üzerine konumlanır. Bundan sonra bu dosyaya yapılacak ilk kayıt 17. kayıt olacaktır.

Sıralı dosyalarda (SEQENTIAL) bilgiler girilen sıraya göre dosyaya yazılır. Dosyadan bilgi çağrılırken , aranılan kayıt ilk kayıttan itibren aranmaya başlar ve aranılan kayıt bulununcaya kadar dosyadan bilgi okunmaya devam edilir.

IV.13.3.2. Sequential Dosyaların ENVIRONMENT DIVISION Bölümünde Tanımlanması Gereken Kısımları

ENVIRONMENT DIVISION bölümünün INPUT-OUTPUT SECTION kısmında kullanılan dosya ile ilgili tüm özellikler tanımlanır. Bu özellikler dosyanın düzenleme şekilleri ve dosyaya ulaşım şekilleri dir.

Bu tanımlama aşağıdaki gibidir.

```
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
        SELECT ‹Dosya Adı› ASSIGN TO DISK / PRINTER /TAPE
        ORGANIZATION IS SEQUENTIAL
        ACCESS MODE IS SEQUENTIAL
        FILE STATUS IS ‹Hata değişkeni›.
```

SELECT deyimini takiben verilen isim dosya adıdır ve başka bir yerde tanımlanmaz. Kullanılacak her dosya için mutlaka bir SELECT deyimi kullanılacaktır.

DATA DIVISION bölümünde burada tanımlanmış dosyanın disketteki gerçek adı aktarılacaktır.

Fakat dosyayı açma , okuma ve kapama gibi işlemler burada belirtilen dosya adına göre yapılacaktır. (SELECT ile verilen dosyanın adı)

ASSIGN deyimi ile dosyanın hangi amaçla açılabileceğini verebiliriz.
Eğer ASSIGN TO DISK kullanılmış ise dosya ile ilgili işlemler disk veya disket üzerinde yapılır.
ASSIGN TO TAPE kullanılmış ise dosya ile ilgili işlemler tape üzerinde yapılır.
Son olarak ASSIGN TO PRINTER kullanılmış ise dosya ile ilgili işlemler yazıcı için yapılacak demektir. Böylece kayıt değişkenini oluşturan tüm bilgiler kayıt değişkeni adı altında doğrudan doğruya yazıcıya aktarılır. Döküm almak için kullanılan bu tip dosyalar sadece OUTPUT modunda açılabilirler.
ORGANIZATION IS SEQUENTIAL kelimesi ile açılacak dosyanın düzenleme şeklinin SEQUENTIAL olduğu belirtilmektedir. SEQUENTIAL tipte bir dosya için dosya düzenleme şeklinin belirtilmesine gerek yoktur. Eğer dosya düzenleme şekli SEQUENTIAL ise ORGANIZATION IS SEQUENTIAL kelimesinin yazılmasına gerek yoktur.
ACCESS MODE IS SEQUENTIAL kelimesi ile açılacak olan dosyanın ulaşım şeklinin SEQUENTIAL (sıralı) olduğu belirtilmektedir. SEQUENTIAL (sıralı) bir dosyada erişim şeklinin belirtilmesine gerek yoktur.
Program içerisinde dosya ile ilgili meydana gelebilecek tüm hataların kod numaraları , FILE-STATUS IS ‹Hata değişkeni› ile belirtilen hata değişkenine yüklenir. Bu hata değişkeni WORKING-STORAGE SECTION bölümünde tanımlanmış olup nümerik veya alfa nümerik 2 karakterli bir değişkendir. Bu değişkene yüklenebilecek hataların bazıları şunlardır ; 00 , 10 , 20 , 30 , 95 gibi. Bu hataların anlamı ileriki konularda verilecektir.
Sonuç olarak SEQUENTIAL (sıralı) dosya düzenleme şekilleri için ENVIRONMENT DIVISION bölümünde sadece ;

```
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT ‹Dosya Adı› ASSIGN TO DISK / PRINTER /TAPE
    FILE STATUS IS ‹Hata değişkeni›.
```

yazılması yeterlidir.

IV.13.3.3. Sequential Dosyaların DATA DIVISION Bölümünde Tanımlanması Gereken Kısımları

Çeşitli dosyalar ile ilgili ve belleği ilgilendiren bilgiler DATA DIVISION bölümünün FILE SECTION kısmında tanımlanır. ENVIRONMENT DIVISION bölümünde tanımlanmış her dosya için DATA DIVISION bölümünde bir takım tanımlamalar yapılır. Bunlardan bir tanesi FD seviyesidir. Bir SEQUENTIAL tipte dosya içi DATA DIVISION bölümünde aşağıdaki tanımlamalar yapılabilir.

```
DATA DIVISION.
FILE SECTION.
FD ‹Dosya Adi› LABEL RECORD IS STANDARD/OMITTED
    VALUE OF FILE-ID IS ‹Dosyanın disketteki adı›
    DATA RECORD IS ‹Kayıt Değişkeni›.
01 ‹Kayıt Değişkeni›.
   02 ‹Kayıt bilgilerinin tanimi›.
```

FD seviyesi ile belirtilen dosya adı ENVIRONMENT DIVISION bölümünde SELECT ile verilmiş dosya adı ile aynı olmalıdır. ENVIRONMENT DIVISION bölümünde SELECT ile verilmiş her dosya adı için DATA DIVISION bölümünde bir FD seviyesi bulunmalıdır.

Dosya disket veya disk içinde açılacak ise LABEL RECORD IS STANDARD tanımlaması yapılmalıdır.

VAUE OF FILE-ID ile verilen ‹Dosyanın disketteki adı› ile dosyanın disketteki adı belirtilir. Bu WORKING-STORAGE SECTION bölümünde tanımlanmış olan karakter tipindeki bir değişken de olabilir.

DATA RECORD IS ‹Kayıt Değişkeni› ile verilen kayıt adı DATA DIVISION bölümünün FILE SECTION kısmında 01 seviyesinde tanımlanmış olan bir değişken ismidir. Eğer kayıt değişkeni FD seviyesinin hemen altında belirtilir ise DATA RECORD IS ‹Kayıt Değişkeni› kelimesinin yazılmasına gerek yoktur.

01 seviyesinde tanımlanan ‹Kayıt Değişkeni› ile dosyaya yazılacak bilgiler tanımlanmalıdır. Dosyaya kayıt yapma işlemleri burada belirtilen kayıt değişkeni adı altında yapılacaktır.

Eğer dosya sadece döküm amacı ile yazıcı için açılacak ise LABEL RECORD IS OMITTED kullanılmalıdır. Yazıcı dosyaları için VALUE OF FILE-ID IS ‹Dosyanın disketteki adı› kısmının yazılmasına gerek yoktur. Dosya döküm amacı ile açılacak ise DATA DIVISION bölümünde aşağıda belirtilen tanımlamalar yapılabilir.

```
DATA DIVISION.
FILE SECTION.
FD ‹Dosya Adı› LABEL RECORD IS OMITTED
    LINAGE IS ‹Değişken1› LINES
    LINES AT TOP ‹Değişken2›
    LINES AT BOTTOM ‹Değişken3›.
01 ‹Kayıt Adı›.
    02 ‹Kayıt bilgilerinin tanımı›.
```

Burada LABEL RECORD IS OMITTED ile dosyanın yazıcı dosyası olduğu belirtilmekte. Yazıcı dosyaları için yazılması zorunludur.

LINAGE IS ‹Değişken1› LINES cümlesi yazıcı dosyaları için kullanılır. Kullanılan printer sayfaları genel de 66 satırdan meydana gelmektedir. Ancak değişik dökümler için farklı boyutlarda printer kağıdı kullanılabilir. LINAGE IS ‹Değişken1› LINES cümlesi ile kullanılacak printer kağıdının boyutunu verebiliriz. Değişken1 ile ifade edilen değişkenin WORKING-STORAGE SECTION bölümünde nümerik bir değişken olarak tanımlanmış olması gerekir. Burada değişken yerine herhangi bir tamsayı da kullanılabilir.

Örnek IV.13.3.a.

LINAGE IS 60 LINES cümlesi ile printer kağıdının 60 satırdan meydana geldiğini belirtilmektedir.

Eğer LINAGE kelimesi kullanılmadı ise printer kağıdının 66 satırdan meydana geldiği COBOL derleyicisi tarafından bilinir.

LINES AT BATTOM ‹Değişken2› ile belirtilen Değişken2 ile printer kağıdının başlangıcından itibaren kaç satır boşluk bırakması gerektiği belirtilebilir.

Örnek IV.13.3.b.

LINES AT TOP 5 yazılımı ile printer kağıdının

başlangıcından itibaren 5 satır boş bırakılacağı belirtilmetedir. Yazılımı zorunlu olmayan bir kelimedir.
LINES AT BOTTOM ‹Değişken3› ile belirtilen Değişken3 ile printer kağıdının sonundan itibaren kaç satır boşluk bırakılacağı belirtilebilir. Yazılımı zorunlu olmayan bir kelimedir.
Örnek IV.13.3.c.
LINES AT BOTTOM 5 yazılımı ile printer kağıdının sonunda 5 satır boşluk bırakılacağı belirtilmektedir.
01 seviyesinde ‹Kayıt Adı› ile yazıcı dosyasının kayıt değişkeninin adı belirtilmektedir. Yazıcıdan çıkartılacak dökümler kayıt adı değişkeni vasıtası ile yapılır.

IV.13.3.4. Sequential Dosyaların PROCEDURE DIVISION Bölümünde Tanımlanması Gereken Kısımları

Cobol programlama dilinde PROCEDURE DIVISION bölümünde dosyalar ile ilgili yapılması gereken açıklamaların ilki dosyanın açılmasıdır. Bu bölümde dosyanın açılması için OPEN deyimi kullanılır. Eğer DATA DIVISION bölümünde dosyanın disk veya disketteki adı WORKING-STORAGE SECTION bölümünde tanımlanmış bir karakter tipindeki değişken ise PROCEDURE DIVISION bölümünde , OPEN deyimi kullanılmadan önce bu değişkene dosyanın disk veya disketteki gerçek adı aktarılır.

IV.3.13.3.5. OPEN DEYİMİ

PROCEDURE DIVISION bölümünde dosya açmak için kullanılır.
Genel Yazılım Formatı :

OPEN ‹Dosya Açılım Modu› ‹Dosya Adı›.

Yukarıda elirtilen ‹Dosya Açılım Modu› ile dosyanın hangi amaçla açılacağı belirtilmektedir. Sequential dosyalarda dosya açma şekilleri OUTPUT , INPUT , EXTEND , I-O modlarında olabilir.
‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünün FILE-CONTROL kısmında SELECT ile belirtilmiş olan dosya adı belirtilmelidir.

OUTPUT modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN OUTPUT ‹Dosya Adı›.

OUTPUT modu sadece dosyaya bilgi yazmak için kullanılır ve yanlızca yazıcı kafa çalışır. Dosya diskette ilk defa açıldığı zaman bu modda açılmalıdır. OUTPUT modunda açılmış olan bir dosyada daha önceden bulunan bütün kayıtlar silinir. Bundan dolayı dosyanın OUTPUT modunda açılırken çok dikkatli olunması gerekir. Örnek olarak içerisinde 20 kayıt bulunan bir dosya OUTPUT modda açılacak olur ise daha önceden yapılmış olan 20 kayıt silinir. Çünkü bir dosya OUTPUT modda açılıyor ise yazıcı kafa ilk kayıdın üzerine konumlanır böylece daha önce yapılmış bulunan tüm kayıtlar yok olur. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. OUTPUT modda açılmış olan SEQUENTIAL dosyalarda sadece WRITE deyimi çalışır.

INPUT modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN INPUT ‹Dosya Adı›.

INPUT modu sadece dosyadan bilgi okumak amacı için kullanılır ve yanlızca okuyucu kafa çalışır. Dosyanın INPUT modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. INPUT modunda açılmış SEQUENTIAL dosyalarda sadece READ deyimi çalışır.

I-O modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN I-O ‹Dosya Adı›.

I-O (INPUT-OUTPUT) modu dosyadan bilgi okuma ve dosyaya bilgi yazmak için kullanılır. Okuyucu ve yazıcı kafa beraber hareket eder. Ancak bu hareket dosya sonu işaretine kadar sınırlıdır. Yani I-O modunda açılan bir dosyaya yeni bir kayıt ilavesinde bulunmak söz konusu değildir. I-O modu kullanılarak dosyada bulunan bir kayıt değiştirilebilir. Bu değişikliği yapabilmek için ilk önce READ deyimi ile değiştirilecek kayıdın okunması vee değişiklik yapıldıktan sonra REWRITE deyimi ile bu kayıdın tekrar eski yerine yazdırılması gerekir. Dosyanın bu modda

açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. I-O modunda açılmış olan SEQUENTIAL dosyalarda sadece REWRITE deyimi çalışır.

EXTEND modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN EXTEND ‹Dosya Adı›.

EXTEND modu SEQUENTIAL dosyalarda kayıt ilavesi yapmak için kullanılır. Sadece yazıcı kafa çalışır. Herhangi bir dosya bu mod ile açıldığında yazıcı kafa daha önceden yapılmış olan kayıtlardan sonraki kayıda konumlanır. EXTEND modu disk veya disket sürücüsündeki yazıcı kafayı , daha önce OUTPUT modunda açılmış olan dosyanın sonuna konumlandırıp , bu noktadan itibaren yeni kayıt yapmak amacı ile kullanılır. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. Örnek olarak bir 16 kayıt bulunan dosya EXTEND modunda açılmış ise yazıcı kafa 17. kayıt üzerine konumlanır. Bundan sonra bu dosyaya yapılacak ilk kayıt 17. kayıt olacaktır.

IV.13.3.6. WRITE DEYİMİ

WRITE deyimi DATA DIVISION bölümünün FILE SECTION kısmında 01 seviyesinde belirtilmiş olan (DATA RECORD IS ile verilmiş olan Kayıt Değişkeni) Kayıt Değişkeni 'nin içeriklerini ilgili dosyaya yazılmasını sağlar. WRITE deyiminin çalıştırılabilmesi için dosyanın OUTPUT veya EXTEND mod da açılmış olması gerekmektedir.

Genel Yazılım Formatı :

WRITE ‹Kayıt Değişkeni› FROM ‹Değişken›.

WRITE deyiminin en basit şekli WRITE ‹Kayıt Değişkeni› dir. Bu şekilde kullanıldığında o anda ‹Kayıt Değişkeni›'nde bulunan bilgiler dosyaya yazılır.

Eğer FROM kullanılacak ise FROM 'dan sonra beliritilen ‹Değişken› 'e ait bilgiler ‹Kayıt Değişkeni› aracılığı ile dosyaya yazılırlar. Ancak bu ‹Değişken› içerik , uzunluk ve tip olarak ‹Kayıt Değişkeni› ne uygun olması gerekir.

WRITE deyimi yazıcı döküm amacı için kullanılıyor ise (Yazıcı dosyaları için) aşağıdaki tanımlamalar yapılabilir.

WRITE ‹Kayıt Değişkeni› FROM ‹Değişken›
BEFORE / AFTER ADVANCING PAGE ‹Tamsayı değişken› LINES
AT END-OF-PAGE ‹Komut›.

Eğer BEFORE ADVANCING PAGE ‹Tamsayı değişken› kullanılır ise ; Printer'e takılı bulunan kağıda ‹Kayıt Değişkeni›'ne ait bilgiler yazıldıktan sonra , belirtilen ‹Tamsayı değişken› 'nin sayısı kadar boş satır bırakılır.

Eğer BEFORE ADVANCING PAGE kullanılır ise ; Printer'e takılı bulunan kağıda ‹Kayıt Değişkeni›'ne ait bilgiler yazıldıktan sonra , bir sayfa boş bırakılır.

Eğer AFTER ADVANCING PAGE ‹Tamsayı değişken› kullanılır ise ; Printer'e takılı bulunan kağıda ‹Kayıt Değişkeni›'ne ait bilgiler yazılmadan önce , belirtilen ‹Tamsayı değişken› 'nin sayısı kadar boş satır bırakılır.

Eğer AFTER ADVANCING PAGE kullanılır ise ; Printer'e takılı bulunan kağıda ‹Kayıt Değişkeni›'ne ait bilgiler yazılmadan önce , bir sayfa boş bırakılır.

Eğer AT END-OF-PAGE ‹Komut› kullanılır ise ; WRITE deyimi ile o ana dek , 66 satır tutarında bilgi yazılmış ise , diğer bir deyiş ile printer kağıdının sonuna gelinmiş ise icra akışı bu cümle ile verilen ‹Komut› 'a verilecektir. Bu cümle genellikle döküm esnasında kağıt değiştirmek için kullanıcıya bir fırsat vermek için kullanılabilir.

IV.13.3.7. READ DEYİMİ

Diskette mevcut olan bir dosyanın kayıtlarını okumak ve kayıt değişkenine aktarmak için kullanılır.

Genel Yazılım Formatı :

READ ‹Dosya Adı› RECORD INTO ‹Değişken› AT END ‹Komut›.

Belirtilen formda yer alan RECORD kelimesi cümle bütünlüğü içindir. Yazılmasının yada yazılmamasının bir önemi yoktur. READ deyiminin çalıştırılabilmesi için dosyanın INPUT veya I-O modunda açılmış olması gerekir.

READ deyiminin en basit şekli READ ‹Dosya Adı› dır. Bu şekilde kullanılan bir READ deyimi , okuyucu kafanın bulunduğu üzerinde bulunulan kayıdı okuyarak , DATA

DIVISION bölümünün FILE SECTION kesiminde 01 seviyesinde tanımlanmış olan ‹Kayıt Değişkeni›'ne aktarır. Burada belirtilen ‹Dosya Adı›, ENVIRONMENT DIVISION bölümünün FILE-CONTROL kısmında SELECT ile verilmiş dosya adı ile aynı olması gerekmektedir.

Okutulan kayıt okuma işleminden sonra ‹Kayıt Değişkeni› ninden başka bir değişkene aktarılacak ise ;

READ ‹Dosya Adı› RECORD INTO ‹Değişken›

kullanılır. Bu yazılıma göre dosyadan okutulan kayıt INTO ile verilen ‹Değişken›'e aktarılır. Ancak bu değişkenin ‹Kayıt Değişkeni› ile içerik , uzunluk ve tip olarak aynı olması gerekmektedir.

SEQUENTIAL dosyalarda dosya INPUT veya I-O modunda açılıp READ deyimi icra ettirilir ise , dosyanın ilk kayıdına ait bilgiler ‹Kayıt Değişkeni›'ne aktarılır. Çünkü dosya ilk açıldığında okuyucu kafa dosyanın ilk kayıdı üzerine konumlanır ve her READ deyimi icra ettirildiği zaman okuyucu kafa bir sonraki kaydın üzerine konumlanır. Böylece okuyucu kafa dosya sonuna geldiği zaman READ deyimi çalıştırılır ise program hata verir. Böyle bir hata meydana geldiğinde ENVIRONMENT DIVISION bölümünde FILE STATUS ile belirtilmiş olan ‹Hata Değişkeni›'ne 10 değeri yüklenir. Bu READ deyiminin icrasından önce kontrol edilir ise meydana gelebilecek hata önlenebilir.

Dosya sonuna gelindiğinde meydana gelebilecek hata ;

READ ‹Dosya Adı› RECORD INTO ‹Değişken› AT END ‹Komut›,

kullanılarak önlenebilir. Burada dosya sonuna gelindiğinde, son kayıt okunduktan sonra programın akışı AT END ile belirtilen ‹Komut› 'lara kayar.

IV.13.3.8. REWRITE DEYİMİ

REWRITE deyimi , yazıcı kafayı bulunan dosya pozisyonundan bir önceki kaydın başına göndererek , o anda kayıt değişkeni tarafından temsil edilen bilgileri bu pozisyondaki mevcut kaydın üzerine yazar. Bundan dolayı REWRITE deyimi kullanılmadan önce mutlaka READ deyiminin başarılı bir şekilde çalıştırılmış olması gerekir. Çünkü READ deyimi hiç çalıştırılmadan önce okuyucu kafa dosyanın

birinci kayıdı üzerindedir. Bu durumda iken REWRITE deyimi çalıştırılır ise bir önceki kayıt olmadığından hata meydana gelecek ve program Run-time error hatası vererek kırılacaktır.

Genel Yazılım Formatı :

REWRITE ‹Kayıt Değişkeni› FROM ‹Değişken›.

REWRITE deyimi ile ‹Kayıt Değişkeni› ile ifade edilen bilgiler dosyaya yeniden yazılır. REWRITE deyimi ile dosya içerisinde olan bir bilgi üzerinde gerekli değişiklikler yapıldıktan sonra tekrar aynı yerine yazılır. REWRITE deyimi ile dosyaya yeni kayıt yapılmaz. REWRITE deyiminin çalışabilmesi için dosyanın I-O modunda açılmış olması gerekir.

Örnek IV.13.3.d.

Disketteki adı "SE.DAT" , dosya değişkenleri ADI (10) , SOYADI (20) , NUMARA (4) olan sequential dosyaya bilgi ilavesinde bulunabilmek için aşağıdaki program yapılmalıdır.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. DOSYALAMA.
 AUTHOR. MUSLUM ÖZIŞIK.
 DATE-WRITTEN. 09-11-1989.
 DATE-COMPILED. 09-11-1989.
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT DOSYA ASSIGN TO DISK
     ORGANIZATION IS SEQUENTIAL
     ACCESS MODE  IS SEQUENTIAL
     FILE STATUS  IS HATA.
 DATA DIVISION.
 FILE SECTION.
 FD DOSYA LABEL RECORD IS STANDARD VALUE OF FILE-ID "SE.DAT".
 01 KAYIT.
     02 ADI      PIC X(10).
     02 SOYADI   PIC X(10).
     02 NUMARA   PIC 9(4).
```

```
WORKING-STORAGE SECTION.
77 HATA     PIC XX  VALUE SPACES.
77 CEVAP    PIC X VALUE SPACE.
PROCEDURE DIVISION.
BASLA.
    OPEN EXTEND DOSYA.
    IF HATA = 30 OPEN OUTPUT DOSYA CLOSE DOSYA GO BASLA.
BASLA2.
    DISPLAY (1 1) ERASE.
    DISPLAY (10 20) "ADI.............:".
    ACCEPT ( , ) ADI.
    DISPLAY (11 20) "SOYADI.........:".
    ACCEPT ( , ) SOYADI.
    DISPLAY (12 20)"NUMARA.........:".
    ACCEPT ( , ) NUMARA.
    WRITE KAYIT.
    DISPLAY (23 20) "BAŞKA KAYIT YAPACAK MISINIZ..(E/H).:".
    ACCEPT ( , ) CEVAP WITH BEEP.
    IF CEVAP = "E" OR "e" GO BASLA2.
    CLOSE DOSYA.
SON.
    STOP RUN.
```

Örnek IV.13.3.e.

Yukarıda açılmış olan sequential dosyadan bilgi okuyabilmek için aşağıdaki program yapılmalıdır.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. DOSYALAMA.
AUTHOR. MUSLUM ÖZIŞIK.
DATE-WRITTEN. 09-11-1989.
DATE-COMPILED. 09-11-1989.
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT DOSYA ASSIGN TO DISK
    ORGANIZATION IS SEQUENTIAL
    ACCESS MODE  IS SEQUENTIAL
    FILE STATUS  IS HATA.
```

```
DATA DIVISION.
FILE SECTION.
FD DOSYA LABEL RECORD IS STANDARD VALUE OF FILE-ID "SE.DAT".
01  KAYIT.
    02 ADI      PIC X(10).
    02 SOYADI   PIC X(10).
    02 NUMARA   PIC 9(4).
WORKING-STORAGE SECTION.
77 HATA   PIC XX  VALUE SPACES.
77 CEVAP   PIC X VALUE SPACE.
PROCEDURE DIVISION.
BASLA.
    MOVE 9 TO LIN.
    OPEN INPUT DOSYA.
    IF HATA = 30 OPEN OUTPUT DOSYA CLOSE DOSYA GO BASLA.
BASLA2.
    READ DOSYA AT END GO SON.
    DISPLAY (LIN , 20) ADI.
    DISPLAY (LIN , 35) SOYADI.
    DISPLAY (LIN , 50) NUMARA.
    ADD 1 TO LIN.
    GO BASLA2.
SON.
    STOP RUN.
```

Örnek IV.13.3.f.

Yazıcıda takılı olan kağıda "YIDIZ ÜNİVERSİTESİ" yazısını yazacak programı yazıcı dosyalarını kullanarak yapınız.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. PRINTERE-ALMA.
 AUTHOR. MUSLUM ÖZIŞIK.
 DATE-WRITTEN. 09-11-1989.
 DATE-COMPILED. 09-11-1989.
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT DOSYA ASSIGN TO PRINTER.
```

```
DATA DIVISION.
FILE SECTION.
FD DOSYA LABEL RECORD IS OMITTED DATA RECORD IS KAYIT
    LINAGE IS 12 LINES LINES AT TOP 5 LINES AT BOTTOM 5
    LINES WITH FOOTING SAYFA-SONU.
01 KAYIT PIC X(80).
WORKING-STORAGE SECTION.
77 SAYFA-SONU PIC 9 VALUE 1.
01  BASLIK.
    02 F PIC X(30) VALUE SPACES.
    02 F  PIC X(20) VALUE "YILDIZ  UNIVERSITESI".
    02 F   PIC X(30) VALUE SPACES.
PROCEDURE DIVISION.
BASLA.
   OPEN OUTPUT DOSYA.
   WRITE KAYIT FROM BASLIK.
   CLOSE DOSYA.
SON.
   STOP RUN.
```

Örnek IV.13.3.g.

Disketteki adı "ESRA.DAT" , dosya değişkenleri AD (20) , SOYAD (20) , TEL (7) , ADR (20) olan sequential dosyada kayıt yapma , kayıt görme , kayıt silme , kayıt değişiklik, kayıt listeleme bölümlerini yapacak program aşağıda verilmiştir.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. ESRA.
 AUTHOR. GULNUR AKYOL.
 DATE-WRITTEN. 09-15-1989.
 DATE-COMPILED. 09-18-1989.
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT DOSYA ASSIGN TO DISK
     ORGANIZATION IS SEQUENTIAL
     ACCESS MODE IS SEQUENTIAL
     FILE STATUS IS HATA.
```

```
    SELECT YAZICI ASSIGN TO PRINTER.
DATA DIVISION.
FILE SECTION.
FD  DOSYA LABEL RECORD IS STANDARD
        VALUE OF FILE-ID "ESRA.DAT".
01 KAYIT.
    02  AD        PIC X(20).
    02  SOYAD     PIC X(20).
    02  TEL       PIC X(7).
    02  ADR       PIC X(20).
FD  YAZICI LABEL RECORD IS OMITTED.
01 KAYIT1          PIC X(80).
WORKING-STORAGE SECTION.
77  HATA          PIC X(2)    VALUE SPACES.
77  YATAY         PIC X(80)   VALUE ALL "-".
77  SEC           PIC X       VALUE SPACE.
77  BOS           PIC X(80)   VALUE SPACES.
77  AD2           PIC X(20)   VALUE SPACES.
77  SOYAD2        PIC X(20)   VALUE SPACES.
77  I             PIC 9(3)    VALUE 0.
77  J             PIC 9(3)    VALUE 0.
77  SAY           PIC 9(3)    VALUE 0.
77  SAY1          PIC 9(3)    VALUE 0.
77  SAYFA         PIC 9(3)    VALUE 1.
77  K             PIC 9(3)    VALUE 0.
77  TUS           PIC X(2).
      88 F1                     VALUE "02".
      88 F2                     VALUE "03".
      88 F3                     VALUE "04".
      88 F4                     VALUE "05".
      88 F5                     VALUE "06".
      88 F6                     VALUE "07".
      88 F7                     VALUE "08".
      88 F8                     VALUE "09".
      88 F9                     VALUE "10".
01 DIZI.
    02 DIZ-AD  OCCURS 99 TIMES PIC X(20).
```

```
    02 DIZ-SOY OCCURS 99 TIMES PIC X(20).
    02 DIZ-TEL OCCURS 99 TIMES PIC X(7).
    02 DIZ-ADR OCCURS 99 TIMES PIC X(20).
01 BAS1.
    02 F          PIC X(5)  VALUE SPACES.
    02 F          PIC X(20) VALUE "        ADI         ".
    02 F          PIC X     VALUE SPACE.
    02 F          PIC X(20) VALUE "      SOYADI        ".
    02 F          PIC X     VALUE SPACE.
    02 F          PIC X(7)  VALUE  "TELEFON".
    02 F          PIC X     VALUE SPACE.
    02 F          PIC X(20) VALUE "       ADRES        ".
    02 F          PIC X(5)  VALUE SPACES.
01 BAS2.
    02 F            PIC X(10) VALUE SPACES.
    02 AD1          PIC X(20).
    02 F            PIC X     VALUE SPACE.
    02 SOYAD1       PIC X(20).
    02 F            PIC X     VALUE SPACE.
    02 TEL1         PIC X(7).
    02 F            PIC X     VALUE SPACE.
    02 ADR1         PIC X(20).
SCREEN SECTION.
01 IBR1.
    02 BLANK SCREEN.
01 IBR2.
    02 LINE 1 COLUMN 1 VALUE "┌".
    02 COLUMN 2 PIC X(78) USING YATAY.
    02 COLUMN 80 VALUE "┐".
    02 LINE 2 COLUMN 1 VALUE "│".
    02 COLUMN 80 VALUE "│".
    02 LINE 3 COLUMN 1 VALUE "└".
    02 COLUMN 2 PIC X(78) USING YATAY.
    02 COLUMN 80 VALUE "┘".
01 IBR3.
    02 LINE 22 COLUMN 1 VALUE "┌".
    02 COLUMN 2 PIC X(78) USING YATAY.
```

```
    02 COLUMN 80 VALUE "┐".
    02 LINE 23 COLUMN 1 VALUE "│".
    02 COLUMN 80 VALUE "│".
    02 LINE 24 COLUMN 1 VALUE "└".
    02 COLUMN 2 PIC X(78) USING YATAY.
    02 COLUMN 80 VALUE "┘".
01 IBR4.
    02 LINE 2 COLUMN 10 VALUE "YILDIZ ÜNİVERSİTESİ MATEMAT
-     "İK MÜHENDİSLİĞİ BİLGİ İŞLEM MERKEZİ " REVERSE-VIDEO.
01 IBR5.
    02 LINE 7 COLUMN 22 VALUE "**** A N A    M E N Ü ****"
          REVERSE-VIDEO BLINK.
    02 LINE 9 COLUMN 22 VALUE "‹F1›" REVERSE-VIDEO.
    02 COLUMN 26 VALUE "....KAYIT YAPMA .............".
    02 LINE 11 COLUMN 22 VALUE "‹F2›" REVERSE-VIDEO.
    02 COLUMN 26 VALUE "....KAYIT GÖRME .............".
    02 LINE 13 COLUMN 22 VALUE "‹F3›" REVERSE-VIDEO.
    02 COLUMN 26 VALUE "....KAYIT DEĞİŞİKLİK .......".
    02 LINE 15 COLUMN 22 VALUE "‹F4›" REVERSE-VIDEO.
    02 COLUMN 26 VALUE "....KAYIT İPTAL .............".
    02 LINE 17 COLUMN 22 VALUE "‹F5›" REVERSE-VIDEO.
    02 COLUMN 26 VALUE "....KAYIT LİSTELEME ........".
    02 LINE 19 COLUMN 22 VALUE "‹F6›" REVERSE-VIDEO.
    02 COLUMN 26 VALUE "....PROGRAM SONU ............".
01 IBR6.
    02 LINE 23 COLUMN 2 VALUE "MESAJ...!" REVERSE-VIDEO.
    02 COLUMN 17 BLINK REVERSE-VIDEO VALUE
      "SEÇİMİNİZ ...‹F1›.‹F2›.‹F3›.‹F4›.‹F5›.‹F6›    [.]".
01 IBR7.
    02 LINE 23 COLUMN 63 PIC X USING SEC AUTO.
01 IBR8.
    02 LINE 23 COLUMN 11 PIC X(67) USING BOS.
01 IBR9.
    02 LINE 8 COLUMN 17 VALUE "┌".
    02 COLUMN 18 PIC X(48) USING YATAY.
    02 COLUMN 66 VALUE "┐".
```

```
01 IBR10.
    02 LINE 19 COLUMN 17 VALUE "└".
    02 COLUMN 18 PIC X(48) USING YATAY.
    02 COLUMN 66 VALUE "┘".
01 IBR11.
    02 LINE 11 COLUMN 19 VALUE "ADI.............: ".
    02 LINE 13 COLUMN 19 VALUE "SOYADI..........: ".
    02 LINE 15 COLUMN 19 VALUE "TELEFONU........: ".
    02 LINE 17 COLUMN 19 VALUE "ADRESİ..........: ".
01 IBR12.
    02 LINE 11 COLUMN 36 PIC X(20) USING AD UNDERLINE.
    02 LINE 13 COLUMN 36 PIC X(20) USING SOYAD UNDERLINE.
    02 LINE 15 COLUMN 36 PIC X(7) USING TEL UNDERLINE.
    02 LINE 17 COLUMN 36 PIC X(20) USING ADR UNDERLINE.
01 IBR14.
    02 LINE 23 COLUMN 17 VALUE "ANA MENU".
    02 COLUMN 17 VALUE "A" BLINK REVERSE-VIDEO.
    02 COLUMN 32 VALUE "DEVAM".
    02 COLUMN 32 VALUE "D" BLINK REVERSE-VIDEO.
    02 COLUMN 42 VALUE "YAZICI".
    02 COLUMN 42 VALUE "Y" BLINK REVERSE-VIDEO.
    02 COLUMN 54 VALUE "SEÇİMİNİZ [.]".
    02 COLUMN 65 VALUE "." BLINK REVERSE-VIDEO.
01 IBR15.
    02 LINE 23 COLUMN 65 PIC X USING SEC BLINK
                    REVERSE-VIDEO AUTO.
01 IBR16.
    02 LINE 11 COLUMN 36 PIC X(20) USING AD2 UNDERLINE.
    02 LINE 13 COLUMN 36 PIC X(20) USING SOYAD2 UNDERLINE.
01 IBR17.
    02 LINE 23 COLUMN 17 VALUE "DOSYAYA HENUZ KAYIT YAPILM
-                          "AMIŞ..! ANA MENUYE DÖNULUYOR...!"
                                    REVERSE-VIDEO BLINK.
01 IBR18.
    02 LINE 23 COLUMN 17 VALUE "ARANILAN KAYIT BULUNAMADI
-                         ".......! ANA MENUYE DÖNULUYOR...!"
                                    REVERSE-VIDEO BLINK.
```

```
01 IBR19.
    02 LINE 4 COLUMN 1 VALUE "┌".
    02 COLUMN 2 PIC X(22) USING YATAY.
    02 COLUMN 24 VALUE "┬".
    02 COLUMN 25 PIC X(22) USING YATAY.
    02 COLUMN 47 VALUE "┬".
    02 COLUMN 48 PIC X(9) USING YATAY.
    02 COLUMN 57 VALUE "┬".
    02 COLUMN 58 PIC X(22) USING YATAY.
    02 COLUMN 80 VALUE "┐".
    02 LINE 5 COLUMN 1 VALUE "│".
    02 COLUMN 24 VALUE "│".
    02 COLUMN 47 VALUE "│".
    02 COLUMN 57 VALUE "│".
    02 COLUMN 80 VALUE "│".
    02 LINE 6 COLUMN 1 VALUE "├".
    02 COLUMN 2 PIC X(22) USING YATAY.
    02 COLUMN 24 VALUE "┼".
    02 COLUMN 25 PIC X(22) USING YATAY.
    02 COLUMN 47 VALUE "┼".
    02 COLUMN 48 PIC X(9) USING YATAY.
    02 COLUMN 57 VALUE "┼".
    02 COLUMN 58 PIC X(22) USING YATAY.
    02 COLUMN 80 VALUE "┤".
    02 LINE 5 COLUMN 2  VALUE "       ADI             "
                        REVERSE-VIDEO.
    02 LINE 5 COLUMN 25 VALUE "      SOYADI           "
                        REVERSE-VIDEO.
    02 LINE 5 COLUMN 48 VALUE " TELEFON "
                        REVERSE-VIDEO.
    02 LINE 5 COLUMN 58 VALUE "       ADRES           "
                        REVERSE-VIDEO.
01 IBR20.
    02 LINE 21 COLUMN 1 VALUE "└".
    02 COLUMN 2 PIC X(22) USING YATAY.
    02 COLUMN 24 VALUE "┴".
    02 COLUMN 25 PIC X(22) USING YATAY.
```

```
    02 COLUMN 47 VALUE "┴".
    02 COLUMN 48 PIC X(9) USING YATAY.
    02 COLUMN 57 VALUE "┴".
    02 COLUMN 58 PIC X(22) USING YATAY.
    02 COLUMN 80 VALUE "┘".
01 IBR21.
    02 LINE 23 COLUMN 14 VALUE "ANA MENU".
    02 COLUMN 14 VALUE "A" BLINK REVERSE-VIDEO.
    02 COLUMN 24 VALUE "1 SAYFA İLERİ".
    02 COLUMN 32 VALUE "I" BLINK REVERSE-VIDEO.
    02 COLUMN 39 VALUE "1 SAYFA GERİ".
    02 COLUMN 47 VALUE "G" BLINK REVERSE-VIDEO.
    02 COLUMN 55 VALUE "YAZICI".
    02 COLUMN 55 "Y" REVERSE-VIDEO BLINK.
    02 COLUMN 65 VALUE "SEÇİMİNİZ [.]".
    02 COLUMN 76 VALUE "." BLINK REVERSE-VIDEO.
01 IBR22.
    02 LINE 23 COLUMN 76 PIC X USING SEC
          AUTO BLINK REVERSE-VIDEO.
01 IBR23.
    02 LINE 6 COLUMN 17 VALUE "┌".
    02 COLUMN 18 PIC X(47) USING YATAY.
    02 COLUMN 64 VALUE "┐".
01 IBR24.
    02 LINE 18 COLUMN 17 VALUE "└".
    02 COLUMN 18 PIC X(47) USING YATAY.
    02 COLUMN 64 VALUE "┘".
01 IBR25.
    02 LINE 7 COLUMN 18 VALUE "┌".
    02 COLUMN 19 PIC X(45) USING YATAY.
    02 COLUMN 63 VALUE "┐".
01 IBR26.
    02 LINE 17 COLUMN 18 VALUE "└".
    02 COLUMN 19 PIC X(45) USING YATAY.
    02 COLUMN 63 VALUE "┘".
01 IBR27.
    02 LINE 8 COLUMN 19 VALUE "┌".
```

```
           02 COLUMN 20 PIC X(43) USING YATAY.
           02 COLUMN 62 VALUE "┐".
       01 IBR28.
           02 LINE 16 COLUMN 19 VALUE "└".
           02 COLUMN 20 PIC X(43) USING YATAY.
           02 COLUMN 62 VALUE "┘".
       01 IBR29.
           02 LINE 9 COLUMN 20 VALUE "┌".
           02 COLUMN 21 PIC X(41) USING YATAY.
           02 COLUMN 61 VALUE "┐".
       01 IBR30.
           02 LINE 15 COLUMN 20 VALUE "└".
           02 COLUMN 21 PIC X(41) USING YATAY.
           02 COLUMN 61 VALUE "┘".
       01 IBR33.
           02 LINE 10 COLUMN 21 VALUE "┌".
           02 COLUMN 22 PIC X(39) USING YATAY.
           02 COLUMN 60 VALUE "┐".
       01 IBR34.
           02 LINE 14 COLUMN 21 VALUE "└".
           02 COLUMN 22 PIC X(39) USING YATAY.
           02 COLUMN 60 VALUE "┘".
       01 IBR35.
           02 LINE 11 COLUMN 22 VALUE "┌".
           02 COLUMN 23 PIC X(37) USING YATAY.
           02 COLUMN 59 VALUE "┐".
       01 IBR36.
           02 LINE 13 COLUMN 22 VALUE "└".
           02 COLUMN 23 PIC X(37) USING YATAY.
           02 COLUMN 59 VALUE "┘".
       01 IBR37.
           02 LINE 12 COLUMN 29 VALUE "İ Y İ  G Ü N L E R ....!"
                      REVERSE-VIDEO BLINK.
       01 IBR38.
           02 LINE 23 COLUMN 17 VALUE "L Ü T F E N   M A K İ N
      -       "A Y I  K A P A T I N I Z .....!  [.]"
                    REVERSE-VIDEO BLINK.
```

```
01 IBR39.
    02 LINE 23 COLUMN 77 PIC X USING SEC AUTO.
01 IBR-EK1.
    02 LINE 23 COLUMN 17 REVERSE-VIDEO BLINK
     VALUE "DEGISTIRILECEK KAYIT BU MU [E/H]      [.]".
01 IBR-EK2.
    02 LINE 23 COLUMN 56 PIC X USING SEC AUTO.
PROCEDURE DIVISION.
BASLA.
    DISPLAY IBR1 IBR4 IBR2 IBR3.
BAS.
    MOVE " " TO SEC.
    DISPLAY IBR8 IBR5 IBR6.
BAS-1.
    ACCEPT IBR7.
    ACCEPT TUS FROM ESCAPE KEY.
    IF F1 GO KAY-GIR.
    IF F2 GO KAY-GOR.
    IF F3 GO KAY-DEG.
    IF F4 GO KAY-SIL.
    IF F5 GO KAY-LIST.
    IF F6 GO PRG-SON.
    MOVE " " TO SEC.
    GO BAS-1.
KAY-GIR.
    OPEN EXTEND DOSYA IF HATA ="30" OPEN OUTPUT DOSYA.
       PERFORM ESR1 DISPLAY IBR8.
KAY-GIR-1.
    MOVE SPACES TO AD SOYAD TEL ADR.
    DISPLAY IBR9.
    PERFORM ESR3 VARYING LIN FROM 9 BY 1 UNTIL LIN > 19.
    DISPLAY IBR10 IBR11 IBR14 IBR12.
    ACCEPT IBR12.
    WRITE KAYIT.
KAY-GIR-2.
    MOVE " " TO SEC.
    ACCEPT IBR15.
```

```
     IF SEC = "A" OR ="a" DISPLAY IBR8 CLOSE DOSYA
                          PERFORM ESR1 GO BAS.
     IF SEC = "D" OR ="d" GO KAY-GIR-1.
     IF SEC = "Y" OR ="y" GO KAY-YAZ.
     GO KAY-GIR-2.
KAY-YAZ.
     OPEN OUTPUT YAZICI.
     WRITE KAYIT1 FROM BAS1.
     MOVE AD TO AD1. MOVE SOYAD TO SOYAD1.
     MOVE TEL TO TEL1. MOVE ADR TO ADR1.
     WRITE KAYIT1 FROM BAS2.
     CLOSE YAZICI.
     GO KAY-GIR-1.
KAY-GOR.
     PERFORM ESR1 DISPLAY IBR8.
KAY-GOR-1.
     DISPLAY IBR9.
     PERFORM ESR3 VARYING LIN FROM 9 BY 1 UNTIL LIN > 19.
     DISPLAY IBR10 IBR11 IBR14.
KAY-GOR-AC.
     MOVE SPACES TO AD SOYAD TEL ADR AD2 SOYAD2.
     DISPLAY IBR12.
     OPEN INPUT DOSYA IF HATA = "30" GO KAY-GOR-MES.
     ACCEPT IBR16.
KAY-GOR-2.
     READ DOSYA NEXT AT END GO KAY-GOR-4.
     IF AD = AD2 AND SOYAD = SOYAD2 GO KAY-GOR-3.
     GO KAY-GOR-2.
KAY-GOR-3.
     DISPLAY IBR12.
     CLOSE DOSYA.
     MOVE " " TO SEC.
     ACCEPT IBR15.
     IF SEC = "A" OR ="a" DISPLAY IBR8 PERFORM ESR1 GO BAS.
     IF SEC = "D" OR ="d" GO KAY-GOR-AC.
     IF SEC = "Y" OR ="y" GO KAY-GOR-YAZ.
     GO KAY-GOR-3.
```

```
KAY-GOR-4.
    CLOSE DOSYA.
    DISPLAY IBR8 IBR18.
    PERFORM ESR4 VARYING I FROM 1 BY 1 UNTIL I > 700.
    PERFORM ESR1 GO BAS.
KAY-GOR-MES.
    CLOSE DOSYA.
    DISPLAY IBR8 IBR17.
    PERFORM ESR4 VARYING I FROM 1 BY 1 UNTIL I > 700.
    PERFORM ESR1 GO BAS.
KAY-GOR-YAZ.
    OPEN OUTPUT YAZICI.
    WRITE KAYIT1 FROM BAS1.
    MOVE AD TO AD1. MOVE SOYAD TO SOYAD1.
    MOVE TEL TO TEL1. MOVE ADR TO ADR1.
    WRITE KAYIT1 FROM BAS2.
    CLOSE YAZICI.
    GO KAY-GOR-AC.
KAY-DEG.
    PERFORM ESR1 DISPLAY IBR8.
KAY-DEG-1.
    DISPLAY IBR9.
    PERFORM ESR3 VARYING LIN FROM 9 BY 1 UNTIL LIN > 19.
    DISPLAY IBR10 IBR11 IBR14.
KAY-DEG-AC.
    MOVE SPACES TO AD SOYAD TEL ADR AD2 SOYAD2.
    DISPLAY IBR12.
    OPEN I-O DOSYA IF HATA = "30" GO KAY-GOR-MES.
    ACCEPT IBR16.
KAY-DEG-2.
    READ DOSYA NEXT AT END GO KAY-DEG-4.
    IF AD = AD2 AND SOYAD = SOYAD2 GO KAY-DEG-3.
    GO KAY-DEG-2.
KAY-DEG-3.
    DISPLAY IBR12 IBR8 IBR-EK1.
    MOVE " " TO SEC.
    ACCEPT IBR-EK2. DISPLAY IBR8 IBR14.
```

```
    IF SEC = "E" OR = "e" GO KAY-DEG-3-1.
    GO KAY-DEG-3-2.
KAY-DEG-3-1.
    MOVE SPACES TO AD SOYAD TEL ADR.
    DISPLAY IBR12.
    ACCEPT IBR12.
    REWRITE KAYIT.
KAY-DEG-3-2.
    CLOSE DOSYA.
    ACCEPT IBR15.
    IF SEC = "A" OR ="a" DISPLAY IBR8 PERFORM ESR1 GO BAS.
    IF SEC = "D" OR ="d" GO KAY-DEG-AC.
    IF SEC = "Y" OR ="y" GO KAY-DEG-YAZ.
    GO KAY-DEG-3.
KAY-DEG-4.
    CLOSE DOSYA.
    DISPLAY IBR8 IBR18.
    PERFORM ESR4 VARYING I FROM 1 BY 1 UNTIL I > 700.
    PERFORM ESR1 GO BAS.
KAY-DEG-MES.
    CLOSE DOSYA.
    DISPLAY IBR8 IBR17.
    PERFORM ESR4 VARYING I FROM 1 BY 1 UNTIL I > 700.
    PERFORM ESR1 GO BAS.
KAY-DEG-YAZ.
    OPEN OUTPUT YAZICI.
    WRITE KAYIT1 FROM BAS1.
    MOVE AD TO AD1. MOVE SOYAD TO SOYAD1.
    MOVE TEL TO TEL1. MOVE ADR TO ADR1.
    WRITE KAYIT1 FROM BAS2.
    CLOSE YAZICI.
    GO KAY-GOR-AC.
KAY-SIL.
    PERFORM ESR1 DISPLAY IBR8.
KAY-SIL-1.
    DISPLAY IBR9.
    PERFORM ESR3 VARYING LIN FROM 9 BY 1 UNTIL LIN > 19.
    DISPLAY IBR10 IBR11 IBR14.
```

```
KAY-SIL-AC.
    MOVE SPACES TO AD SOYAD TEL ADR AD2 SOYAD2.
    DISPLAY IBR12.
    OPEN I-O DOSYA IF HATA = "30" GO KAY-GOR-MES.
    ACCEPT IBR16.
KAY-SIL-2.
    READ DOSYA NEXT AT END GO KAY-SIL-4.
    IF AD = AD2 AND SOYAD = SOYAD2 GO KAY-SIL-3.
    GO KAY-SIL-2.
KAY-SIL-3.
    DISPLAY IBR12 IBR8 IBR-EK1.
    MOVE " " TO SEC.
    ACCEPT IBR-EK2. DISPLAY IBR8 IBR14.
    IF SEC = "E" OR = "e" GO KAY-SIL-3-1.
    GO KAY-SIL-3-2.
KAY-SIL-3-1.
    MOVE SPACES TO KAYIT.
    REWRITE KAYIT.
KAY-SIL-3-2.
    CLOSE DOSYA.
    ACCEPT IBR15.
    IF SEC = "A" OR ="a" DISPLAY IBR8 PERFORM ESR1 GO BAS.
    IF SEC = "D" OR ="d" GO KAY-SIL-AC.
    IF SEC = "Y" OR ="y" GO KAY-SIL-YAZ.
    GO KAY-SIL-3.
KAY-SIL-4.
    CLOSE DOSYA.
    DISPLAY IBR8 IBR18.
    PERFORM ESR4 VARYING I FROM 1 BY 1 UNTIL I > 700.
    PERFORM ESR1 GO BAS.
KAY-SIL-MES.
    CLOSE DOSYA.
    DISPLAY IBR8 IBR17.
    PERFORM ESR4 VARYING I FROM 1 BY 1 UNTIL I > 700.
    PERFORM ESR1 GO BAS.
KAY-SIL-YAZ.
    OPEN OUTPUT YAZICI.
```

```
    WRITE KAYIT1 FROM BAS1.
    MOVE AD TO AD1. MOVE SOYAD TO SOYAD1.
    MOVE TEL TO TEL1. MOVE ADR TO ADR1.
    WRITE KAYIT1 FROM BAS2.
    CLOSE YAZICI.
    GO KAY-GOR-AC.
KAY-LIST.
    PERFORM ESR1.
    MOVE " " TO SEC.
    DISPLAY IBR8 IBR19.
    PERFORM ESR5 VARYING LIN FROM 7 BY 1 UNTIL LIN > 20.
    DISPLAY IBR20 IBR21.
    OPEN INPUT DOSYA IF HATA = "30" GO KAY-GOR-MES.
    MOVE 7 TO LIN. MOVE 0 TO I.
KAY-LIST-1.
    ADD 1 TO I.
    READ DOSYA NEXT AT END GO KAY-LIST-2.
    PERFORM ESR6.
    GO KAY-LIST-1.
KAY-LIST-2.
    COMPUTE SAY = (SAYFA - 1) * 14.
    COMPUTE SAY1 = SAY + 14.
    ADD 1 TO SAY.
    MOVE 7 TO LIN.
    PERFORM ESR8 VARYING J FROM SAY BY 1 UNTIL J > SAY1.
KAY-LIST-3.
    MOVE " " TO SEC.
    ACCEPT IBR22.
    IF SEC = "A" OR = "a" CLOSE DOSYA PERFORM ESR1 GO BAS.
    IF SEC = "I" OR = "i" GO KAY-ILERI.
    IF SEC = "G" OR = "g" GO KAY-GERI.
    IF SEC = "Y" OR = "y" GO KAY-LIST-YAZ.
    GO KAY-LIST-3.
KAY-ILERI.
    IF I > SAYFA * 14 GO KAY-ILERI-1.
    GO KAY-LIST-2.
```

```
KAY-ILERI-1.
    MOVE SPACES TO AD SOYAD TEL ADR.
    PERFORM ESR7 VARYING LIN FROM 7 BY 1 UNTIL LIN > 20.
    ADD 1 TO SAYFA.
    GO KAY-LIST-2.
KAY-GERI.
    IF SAYFA < 2 GO KAY-LIST-2.
    COMPUTE SAYFA = SAYFA - 1.
    GO KAY-LIST-2.
KAY-LIST-YAZ.
    OPEN OUTPUT YAZICI.
    WRITE KAYIT1 FROM BAS1.
    PERFORM ESR9 VARYING J FROM 1 BY 1 UNTIL J > I.
    MOVE AD TO AD1. MOVE SOYAD TO SOYAD1.
    MOVE TEL TO TEL1. MOVE ADR TO ADR1.
    CLOSE YAZICI.
    GO KAY-LIST-3.
PRG-SON.
    PERFORM ESR1.
    DISPLAY IBR8 IBR23.
    MOVE 17 TO COL. MOVE 17 TO J. MOVE 7 TO I.
    PERFORM ESR10 DISPLAY IBR24.
    ADD 1 TO COL. ADD 2 TO K. ADD 1 TO I.
    COMPUTE J = J - 1.
    DISPLAY IBR25.
    PERFORM ESR10  DISPLAY IBR26.
    DISPLAY IBR27.
    ADD 1 TO COL. ADD 2 TO K. ADD 1 TO I.
    COMPUTE J = J - 1.
    PERFORM ESR10 DISPLAY IBR28.
    DISPLAY IBR29.
    ADD 1 TO COL. ADD 2 TO K. ADD 1 TO I.
    COMPUTE J = J - 1.
    PERFORM ESR10 DISPLAY IBR30.
    DISPLAY IBR35.
    ADD 1 TO COL. ADD 2 TO K. ADD 1 TO I.
    COMPUTE J = J - 1.
```

```
    PERFORM ESR10 DISPLAY IBR36.
    DISPLAY IBR33.
    ADD 1 TO COL. ADD 2 TO K. ADD 1 TO I.
    COMPUTE J = J - 1.
    PERFORM ESR10 DISPLAY IBR34 IBR37 IBR38.
    ACCEPT IBR39.
    STOP RUN.
ESR1.
    PERFORM ESR2 VARYING LIN FROM 4 BY 1 UNTIL LIN > 21.
ESR2.
    DISPLAY(LIN , 1) BOS.
ESR3.
    DISPLAY (LIN , 17) "|".
    DISPLAY (LIN , 66) "|".
ESR4.
ESR5.
    DISPLAY (LIN , 1)  "|". DISPLAY (LIN , 24) "|".
    DISPLAY (LIN , 47) "|". DISPLAY (LIN , 57) "|".
    DISPLAY (LIN , 80) "|".
ESR6.
    MOVE AD TO DIZ-AD(I). MOVE SOYAD TO DIZ-SOY(I).
    MOVE TEL TO DIZ-TEL(I). MOVE ADR TO DIZ-ADR(I).
ESR7.
    DISPLAY (LIN , 3) AD. DISPLAY (LIN , 26) SOYAD.
    DISPLAY (LIN , 49) TEL. DISPLAY (LIN , 59) ADR.
ESR8.
    DISPLAY (LIN , 3) DIZ-AD(J).
    DISPLAY (LIN , 26) DIZ-SOY(J).
    DISPLAY (LIN , 49) DIZ-TEL(J).
    DISPLAY (LIN , 59) DIZ-ADR(J).
    ADD 1 TO LIN.
ESR9.
    MOVE DIZ-AD(J) TO AD1. MOVE DIZ-SOY(J) TO SOYAD1.
    MOVE DIZ-TEL(J) TO TEL1. MOVE DIZ-ADR(J) TO ADR1.
    WRITE KAYIT1 FROM BAS2.
ESR10.
    PERFORM ESR11 VARYING LIN FROM I BY 1 UNTIL LIN > J.
```

```
ESR11.
    DISPLAY (LIN , COL) "|".
    COMPUTE COL = COL + 47 - K.
    DISPLAY (LIN , COL) "|".
    COMPUTE COL = COL - 47 + K.
```

IV.13.4. INDEXED DOSYALAR

Indexed dosyalarda kayıtlar , kayıt bilgilerinden biri ulaşım anahtarı (Index key) seçilerek tutulurlar. Burada belirtilen ulaşım anahtarı dosyanın kayıt değişkenlerinden biri olmak zorundadır. Bu ulaşım anahtarı Index değişkeni olarak isimlendirilir. Index değişkeni ENVIRONMENT DIVISION bölümünde RECORD KEY ile verilir.

IV.13.4.1. Indexed Bir Dosyaya Erişim Şekilleri

Indexed dosyalara erişim şekilleri SEQUENTIAL , RANDOM ve DYNAMIC olmak üzere üç türlü olabilir.

SEQUENTIAL erişim söz konusu ise ; dosyanın kayıt alanlarına , index değişkeninin artan değerleri ile ulaşılabilir. Dosyaya kayıt veya dosyadan kayıt okuma esnasında ındex değişkenine artan değerler vermek gerektiğinden bu ulaşım modu tercih edilmez. SEQUENTIAL erişimli Indexed dosyalara kayıt ilavesi mümkün değildir.

Örnek IV.13.4.1.a.

Indexed tipindeki bir dosyanın dosyanın 17. kayıdına ulaşılmak isteniyor ise 17. kayıttan önceki 16 kayıt , Index değişkenine artan değerler verilerek okunmalıdır.

RANDOM erişim sözkonusu ise ; dosyanın kayıt alanlarına erişim rastgele olacaktır. Index değişkenine dosyadan okunmak istenen kayıdı index değişken bilgisi atanarak okuyucu kafa istenilen kayıt üzerine konumlandırılabilir. Getirdiği çeşitli kolaylıklar nedeni ile bu mod SEQUENTIAL erişim seklinden daha çok tercih edilmektedir.

DYNAMIC erişim sözkonusu ise ; dosya içerisinde bulunan kayıtlara ulaşım programcının isteğine bağlıdır. DYNAMIC erişimde erişim şekli SEQUENTIAL veya RANDOM olabilir.

Örnek IV.13.4.1.b.

Indexed tipinde DYNAMIC erişimli bir dosyada 100. kayıttan itibaren 17 kayıdı çekebilmek için ; DYNAMIC erişimin

random özelliği kullanılarak 100. kayıda ulaşılarak 100. kayıttan itibaren SEQUENTIAL özelliği kullanılarak 17 kayıt çekilebilir.

IV.13.4.2. INDEXED TİPİ DOSYALARIN DOSYA AÇILIM MODLARI

1. INPUT modu :Sadece dosyadan bilgi okumak için kullanılır ve yanlızca okuyucu kafa çalışır. Dosyanın INPUT modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.
2. OUTPUT modu :Sadece dosyaya bilgi yazmak için kullanılır ve yanlızca yazıcı kafa çalışır. Dosya diskette ilk defa açıldığı zaman bu modda açılmalıdır. OUTPUT modunda açılmış olan bir dosyada daha önceden bulunan bütün kayıtlar silinir. Bundan dolayı dosyanın OUTPUT modunda açılırken çok dikkatli olunması gerekir. Örnek olarak içerisinde 20 kayıt bulunan bir dosya OUTPUT modda açılacak olur ise daha önceden yapılmış olan 20 kayıt silinir. Çünkü bir dosya OUTPUT modda açılıyor ise yazıcı kafa ilk kayıdın üzerine konumlanır böylece daha önce yapılmış bulunan tüm kayıtlar yok olur.
3. I-O (INPUT-OUTPUT) modu :dosyadan bilgi okuma ve dosyaya bilgi yazmak için kullanılır. Okuyucu ve yazıcı kafa beraber çalışır. Dosyanın bu modda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.

IV.13.4.3. INDEXED Dosyaların ENVIRONMENT DIVISION Bölümünde Tanımlanması Gereken Kısımları

ENVIRONMENT DIVISION bölümünün INPUT-OUTPUT SECTION kısmında INDEXED tip dosyalar ile ilgili tanımlamalar aşağıdaki gibi olabilir.

```
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT ‹Dosya Adı› ASSIGN TO DISK / TAPE
    ORGANIZATION IS INDEXED
    ACCESS MODE IS SEQUENTIAL/RANDOM/DYNAMIC
    RECORD KEY IS ‹Index Değişkeni›
    ALTERNATE RECORD KEY IS ‹Alterne Index Değişkeni›
    WITH DUPLICATES
    FILE STATUS IS ‹Hata değişkeni›.
```

SELECT deyimini takiben verilen isim dosya adıdır ve başka bir yerde tanımlanmaz. Kullanılacak her dosya için mutlaka bir SELECT deyimi kullanılacaktır.
DATA DIVISION bölümünde burada tanımlanmış dosyanın disketteki gerçek adı aktarılacaktır.
Fakat dosyayı açma , okuma ve kapama gibi işlemler burada belirtilen dosya adına göre yapılacaktır. (SELECT ile verilen dosyanın adı)
ASSIGN deyimi ile dosyanın hangi amaçla açılabileceği belirtilmektedir.
Eğer ASSIGN TO DISK kullanılmış ise dosya ile ilgili işlemler disk veya disket üzerinde yapılır.
ASSIGN TO TAPE kullanılmış ise dosya ile ilgili işlemler tape üzerinde yapılır.
ORGANIZATION IS INDEXED kelimesi ile açılacak dosyanın düzenleme şeklinin INDEXED dosya olduğu belirtilmektedir.
INDEXED olarak açılmış bir dosyaya SEQUENTIAL (SIRALI) erişim yapılmak isteniyor ise ;

ACCESS MODE IS SEQUENTIAL

cümlesi yazılır.
INDEXED olarak açılmış bir dosyaya RANDOM (RASTGELE) erişim yapılmak isteniyor ise ;

ACCESS MODE IS RANDOM

cümlesi yazılır.
INDEXED olarak açılmış bir dosyaya DYNAMIC (Hem SIRALI hemde RASTGELE) erişim yapılmak isteniyor ise ;

ACCESS MODE IS DYNAMIC

cümlesi yazılır.
RECORD KEY IS ‹Index Değişkeni› cümlesi ile belirtilen index değişkeni dosyanın kayıt değişkenlerinden her hangi biri olabilir. Dosyada üzerinde yapılacak olan tüm giriş çıkış işlemleri (yazma , okuma , değiştirme , silme , listeleme gibi) index değişkeninden yararlanılarak yapılır. Index değişkeni en fazla 60 karakter uzunluğunda grup veya element düzeyde tanımlanabilen sayısal veya alfabetik bir değişken olabilir. Index değişkeni her türlü giriş çıkış işlemlerinde belirli bir değeri temsil etmek zorundadır.

Dosyada bulunan her kayıt için index değişkeninin değeri farklı olmalıdır. Dosya içerisinde aynı index değişkene sahip kayıtlar bulunamaz.

ALTERNATE RECORD KEY IS ‹Alterne Index Değişkeni› ile belirtilen alterne index değişkeni dosyanın kayıt değişkenlerinden biri veya birkaçı olabilir.

WITH DUPLICATES ile dosyanın kayıtları arasında tanımlanan alterne index değişkeni değerlerinden bir taneden fazla olabileceği belirtilmektedir. WITH DUPLICATES kullanılmadığı takdirde kayıtlarda aynı alterne index değişkenine sahip kayıtlar bulunamaz.

Örnek IV.13.4.3.a.

Dosyadan herhangi bir kayıt okunmak istensin; Bu kayıdı okumak iki şekilde mümkündür.

Bunların birincisi okunacak kaydın RECORD KEY 'i (index değişkeninin alacağı değer) biliniyor ise bu index değişkenine göre istenen kayıt okunabilir.

Dosyadan bir kayıdı okumanın ikinci şekli ise eğer okunacak kaydın index değişkeninin değeri bilinmiyor ise tanımlanan alterne index değişkenlerinden herhangi birisi ile okunmak istenen kayıt okunabilir. Bu şu şekilde yapılır.

Okunacak kaydın alterne index değişkenlerinden birisi ALTERNATE RECORD KEY IS ile belirtilen Alterne Index Değişkenine aktarılır ve dosyadan okunmak istenen kayıt okunur.

Örnek IV.13.4.3.b.

INDEXED tipli bir dosyanın index değişkeni OKUL-NO olsun. birinci Alterne index değişkeni AD-SOY , ikinci alterne index değişkeni TEL-NO olsun.

Bu dosyadan herhangi bir kayıt okunmak istenildiği zaman ; okunacak kaydın OKUL-NO değeri biliniyor ise okuma işlemi RECORD KEY ile yapılabilir. Eğer aranılan kayıda ait OKUL-NO değeri bilinmiyor ise ; birinci alterne key olan AD-SOY adlı alterne index değişkenine göre okuma işlemi yapılabilir. Eğer bu bilinmiyor fakat TEL-NO biliniyor ise okuma işlemi TEL-NO adlı ikinci alterne index değişkenine göre yapılabilir.

Program içerisinde dosya ile ilgili meydana gelebilecek tüm hataların kod numaraları , FILE-STATUS IS ‹Hata değişkeni› ile belirtilen hata değişkenine yüklenir. Bu hata değişkeni WORKING-STORAGE SECTION bölümünde tanımlanmış olup nümerik veya alfa nümerik 2 karakterli bir değişkendir. Bu değişkene yüklenebilecek hataların bazıları şunlardır ; 00 , 10 , 20 , 30 , 95 gibi. Bu hataların anlamı ileriki konularda verilecektir.

IV.13.4.4. INDEXED Dosyaların DATA DIVISION Bölümünde Tanımlanması Gereken Kısımları

DATA DIVISION bölümünde INDEXED tip dosyalar için aşağıdaki tanımlamalar yapılır.

```
DATA DIVISION.
    FILE SECTION.
    FD ‹Dosya Adı› LABEL RECORD IS STANDARD
        VALUE OF FILE-ID IS ‹Dosyanın diskketteki adı›
        DATA RECORD IS ‹Kayıt Değişkeni›.
    01 ‹Kayıt Değişkeni›.
      02 ‹Kayıt bilgilerinin tanımı›.
```

FD seviyesi ile belirtilen dosya adı ENVIRONMENT DIVISION bölümünde SELECT ile verilmiş dosya adı ile aynı olmalıdır. ENVIRONMENT DIVISION bölümünde SELECT ile verilmiş her dosya adı için DATA DIVISION bölümünde bir FD seviyesi bulunmalıdır.

Dosya disket veya disk içinde açılacak ise LABEL RECORD IS STANDARD tanımlaması yapılmalıdır.

VALUE OF FILE-ID ile verilen ‹Dosyanın disketteki adı› ile dosyanın disketteki adı belirtilir. Bu WORKING-STORAGE SECTION bölümünde tanımlanmış olan karakter tipindeki bir değişken de olabilir.

DATA RECORD IS ‹Kayıt Değişkeni› ile verilen kayıt adı DATA DIVISION bölümünün FILE SECTION kısmında 01 seviyesinde tanımlanmış olan bir değişken ismidir. Eğer kayıt değişkeni FD seviyesinin hemen altında belirtilir ise DATA RECORD IS ‹Kayıt Değişkeni› kelimesinin yazılmasına gerek yoktur.

01 seviyesinde tanımlanan ‹Kayıt Değişkeni› ile dosyaya yazılacak bilgiler tanımlanmalıdır. Dosyaya kayıt yapma

işlemleri burada belirtilen kayıt değişkeni adı altında yapılacaktır.

IV.13.4.5. INDEXED Dosyaların PROCEDURE DIVISION Bölümünde Tanımlanması Gereken Kısımları

Bu bölümde INDEXED tipli bir dosyanın açılması için OPEN deyimi kullanılır. Eğer DATA DIVISION bölümünde dosyanın disk veya disketteki adı WORKING-STORAGE SECTION bölümünde tanımlanmış bir karakter tipindeki değişken ise PROCEDURE DIVISION bölümünde , OPEN deyimi kullanılmadan önce bu değişkene dosyanın disk veya disketteki gerçek adı aktarılır.

IV.13.4.5.1. OPEN DEYİMİ

PROCEDURE DIVISION bölümünde dosya açmak için kullanılır.

Genel Yazılım Formatı :

OPEN ‹Dosya Açılım Modu› ‹Dosya Adı›.

Yukarıda elirtilen ‹Dosya Açılım Modu› ile dosyanın hangi amaçla açılacağı belirtilmektedir. INDEXED dosyalarda dosya açma şekilleri OUTPUT , INPUT , I-O modlarında olabilir. ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünün FILE-CONTROL kısmında SELECT ile belirtilmiş olan dosya adı belirtilmelidir.

1. OUTPUT modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN OUTPUT ‹Dosya Adı›.

OUTPUT modu sadece dosyaya bilgi yazmak için kullanılır ve yanlızca yazıcı kafa çalışır. Dosya diskette ilk defa açıldığı zaman bu modda açılmalıdır. OUTPUT modunda açılmış olan bir dosyada daha önceden bulunan bütün kayıtlar silinir. Bundan dolayı dosyanın OUTPUT modunda açılırken çok dikkatli olunması gerekir. Örnek olarak içerisinde 20 kayıt bulunan bir dosya OUTPUT modda açılacak olur ise daha önceden yapılmış olan 20 kayıt silinir. Çünkü bir dosya OUTPUT modda açılıyor ise yazıcı kafa ilk kayıdın üzerine konumlanır böylece daha önce yapılmış bulunan tüm kayıtlar yok olur. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. OUTPUT modda açılmış olan INDEXED dosyalarda sadece WRITE deyimi çalışır.

2. INPUT modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN INPUT ‹Dosya Adı›.

INPUT modu sadece dosyadan bilgi okumak amacı için kullanılır ve yanlızca okuyucu kafa çalışır. Dosyanın INPUT modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. INPUT modunda açılmış INDEXED dosyalarda sadece READ deyimi çalışır. Eğer dosya OUTPUT modda açılmadan INPUT modda açılmaya çalışılır ise FILE STATUS ile verilen hata değişkenine 30 değeri yüklenir.

3. I-O modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN I-O ‹Dosya Adı›.

I-O (INPUT-OUTPUT) modu dosyadan bilgi okuma ve dosyaya bilgi yazmak için kullanılır. Okuyucu ve yazıcı kafa beraber hareket eder. I-O modu ile açılan INDEXED tipli dosyalarda her türlü giriş ve çıkış işlemi (kayıt yapma , kayıt ilave , kayıt silme , kayıt düzeltme , listeleme) yapılabilir. Dosyanın I-O modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir. Eğer OUTPUT modda açılmamış bir dosya I-O modunda açılmaya kalkılır ise Hata değişkenine 30 değeri yüklenir. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. I-O modunda açılmış olan INDEXED dosyalarda sadece WRITE , READ , START , DELETE ve REWRITE deyimleri çalışır. Bundan dolayı INDEXED dosyalarda dosya açılım şekli olarak I-O modu tercih edilir.

IV.13.4.5.2. WRITE DEYİMİ

WRITE deyimi DATA DIVISION bölümünün FILE SECTION kısmında 01 seviyesinde belirtilmiş olan (DATA RECORD IS ile verilmiş olan Kayıt Değişkeni) Kayıt Değişkeni 'nin içeriklerini ilgili dosyaya yazılmasını sağlar. INDEXED dosyalarda WRITE deyiminin çalıştırılabilmesi için dosyanın

OUTPUT veya I-O modunda açılmış olması gerekmektedir.
Genel Yazılım Formatı :
WRITE ‹Kayıt Değişkeni› FROM ‹Değişken› INVALID KEY ‹İfade›
WRITE deyiminin en basit şekli WRITE ‹Kayıt Değişkeni› dir. Bu şekilde kullanıldığında o anda ‹Kayıt Değişkeni›'nde bulunan bilgiler dosyaya yazılır.
Eğer FROM kullanılacak ise FROM 'dan sonra beliritilen ‹Değişken› 'e ait bilgiler ‹Kayıt Değişkeni› aracılığı ile dosyaya yazılırlar. Ancak bu ‹Değişken› içerik , uzunluk ve tip olarak ‹Kayıt Değişkeni› ne uygun olması gerekir.
Eğer dosyada daha önceden var olan bir kayıt dosyaya WRITE deyimi ile tekrar yazılmaya kalkışılır ise bir hata meydana gelir. WRITE deyiminin icrası sırasında böyle bir hata meydana gelir ise ;
Eğer INVALID KEY ‹İfade› kelimesi kullanıl dı ise programın akışı INVALID KEY ile verilen ifadeye yönlenecektir.

IV.13.4.5.3. READ DEYİMİ

Diskette mevcut olan bir dosyanın kayıtlarını okumak ve kayıt değişkenine aktarmak için kullanılır.
Genel Yazılım Formatı :
READ ‹Dosya Adı› NEXT RECORD INTO ‹Değişken› AT END ‹Komut›.
veya
READ‹Dosya Adı›NEXT RECORD INTO‹Değişken›INVALID KEY‹İfade›.
Belirtilen formda yer alan RECORD kelimesi cümle bütünlüğü içindir. Yazılmasının yada yazılmamasının bir önemi yoktur.
READ deyiminin çalıştırılabilmesi için dosyanın INPUT veya I-O modunda açılmış olması gerekir.
READ deyiminin en basit şekli READ ‹Dosya Adı› dır. Bu şekilde kullanılan bir READ deyimi , okuyucu kafanın bulunduğu üzerinde bulunulan kayıdı okuyarak , DATA DIVISION bölümünün FILE SECTION kesiminde 01 seviyesinde tanımlanmış olan ‹Kayıt Değişkeni›'ne aktarır. Burada belirtilen ‹Dosya Adı› , ENVIRONMENT DIVISION bölümünün FILE-CONTROL kısmında SELECT ile verilmiş dosya adı ile aynı olması gerekmektedir.
Okutulan kayıt okuma işleminden sonra ‹Kayıt Değişkeni› ninden başka bir değişkene aktarılacak ise ;

READ ‹Dosya Adı› NEXT RECORD INTO ‹Değişken› kullanılır. Bu yazılıma göre dosyadan okutulan kayıt INTO ile verilen ‹Değişken›'e aktarılır. Ancak bu değişkenin ‹Kayıt Değişkeni› ile içerik , uzunluk ve tip olarak aynı olması gerekmektedir.

INDEXED tipli bir dosyaya erişim şekli SEQUENTIAL (Sıralı) ise READ deyimi ile NEXT kelimesinin birlikte kullanılması gerekir. Eğer NEXT kullanılmamış ise okuma işi rastgele yapılacaktır.

INDEXED dosyalarda AT END ve INTO deyimi aynı SEQUENTIAL dosyalarda kullanıldığı gibidir.

INVALID KEY ‹İfade› ile dosyada olmayan bir kayıt okunmak istenildiği zaman meydana gelebilecek hatalar ile mesaj verilebilir.

IV.13.4.5.4 START DEYİMİ

INDEXED dosyalarda istenilen kayıt üzerine konumlanabilmek için kullanılan bir deyimdir. Hacmi oldukça büyük dosyalarda , genellikle listeleme bölümlerinde bir takım zorluklar ile karşılaşılabilir. Örneğin dosyanın tamamı değilde belirli bir kısmının listelenmesi istenilebilir. Bu işin gerçekleştirilebilmesi için listelenmesi istenilen aralığın ilk kaydına konumlanma sağlanmalıdır. Bu konumlama START deyimi ile yapılabilir.

Genel Yazılım Formatı :

```
START ‹Dosya Adı› KEY IS EQUAL TO          ‹Index Değişkeni›
                          NOT LESS THAN    ‹Index Değişkeni›
                          GREATER THAN     ‹Index Değişkeni›
                          INVALID KEY      ‹İfade›.
```

START deyimi kullanılmadan önce okunmak istenen kayıtların başlangıç değeri index değişkenine aktarılır. START deyimi ile dosyanın istenilen kayıdı üzerine konumlama yapılabilir. KEY IS cümlesinde kullanılan ifadelere göre ortaya çıkacak etkiler şunlardır.

EQUAL TO kullanıldı ise ; Index değişkenine ait değer "KEY" dosyasından aranır. Aranılan kayıt bulunur ise okuyucu kafa bu kayıt üzerine konumlanır. Eğer aranılan kayıt bulunamadı ise hata değişkenine 23 değeri yüklenir veya programın

akışı INVALID KEY ile verilen ifadeye kayar.
GREATER THAN kullanıldı ise ; "KEY" dosyasındaki değeri index değişkeninden büyük olan ilk kayıt üzerine konumlama yapılır. Eğer "KEY" dosyasındaki en son bilginin değeri , index değişkenine eşit veya daha küçük ise istenilen konumlanma gerçekleşmez ve hata değişkenine 23 değeri yüklenir yada programın akışı INVALID KEY ile verilen ifadeye kayar.
NOT LESS THAN kullanıldı ise ; "KEY" dosyası içerisinde index değişkenine eşit olan değer aranır , böyle bir kayıt yok ise değeri index değişkeninden büyük olan kayıt üzerine konumlama yapılır. Index değişkeni ile temsil edilen değer "KEY" dosyasında yok ise hata değişkenine 23 değeri yüklenir veya programın akışı INVALID KEY ile verilen ifadeye kayar.
START deyimi ile istenilen kayıt üzerine konumlama yapıldıktan sonra READ deyimi çalıştırılarak okunmak istenen kayıt veya kayıtlar okunur.

IV.13.4.5.5. REWRITE DEYİMİ

REWRITE deyimi , yazıcı kafayı index değişkenin bulunduğu kayıt üzerine konumlandırır ve kayıt değişkeni tarafından temsil edilen bilgileri bu pozisyondaki mevcut kaydın üzerine yazar.
Genel Yazılım Formatı :
REWRITE ‹Kayıt Değişkeni›FROM ‹Değişken›INVALID KEY ‹İfade›.
REWRITE deyimi ile ‹Kayıt Değişkeni› ile ifade edilen bilgiler dosyaya yeniden yazılır. REWRITE deyimi ile dosya içerisinde olan bir bilgi üzerinde gerekli değişiklikler yapıldıktan sonra tekrar aynı yerine yazılır. REWRITE deyimi ile dosyaya yeni kayıt yapılmaz. REWRITE deyiminin çalışabilmesi için dosyanın OUTPUT veya I-O modunda açılmış olması gerekir.
INDEXED dosyalarda erişim şekli SEQUENTIAL ise REWRITE deyimi kullanılmadan önce READ deyiminin başarılı bir şekilde çalıştırılmış olması gerekir.
RANDOM ve DYNAMIC erişimin söz konusu olduğu durumlarda REWRITE deyimi ile değiştirilecek olan kayıdın READ deyimi

ile önceden okutulmuş olmasına gerek yoktur. REWRITE deyiminin çalışması için değiştirilecek kayda ait index değerinin index değişkenine aktarılmış olması şarttır.
REWRITE deyimi ile bir kaydın index değişkeninin değeri hariç tüm bilgileri değiştirilebilir.
REWRITE deyiminin çalıştırılması esnasında herhangi bir hata meydana gelcek olur ise programın akışı INVALID KEY ile verilen ifadeye kayar.

IV.13.4.5.6. DELETE DEYİMİ

INDEXED dosyalarda istenilen bir kayıt silinmek istendiği zaman kullanılan bir deyimdir. Silinecek kayıdın index değeri index değişkenine aktarılarak silme işlemi gerçekleştirilir.
Genel Yazılım Formatı :

DELETE ‹Dosya Adı› RECORD INVALID KEY ‹İfade›.

INDEXED dosyaya ulaşım şekli SEQUENTIAL ise silinecek kayıt DELETE deyimi kullanılmadan önce mutlaka READ deyimi ile okutulmuş olamlıdır. Dosyaya erişim şeklinin RANDOM veya DYNAMIC olması durumunda silinecek kaydın önceden READ deyimi ile okutulmuş olmasına gerek yoktur.
DELETE deyiminin kullanılması esnasında herhangi bir hata meydana geldiğinde programın akışı INVALID KEY ile verilen ifadeye kayar.
Örnek IV.13.4.5.a.
Devlet personel bordro programı aşağıdaki şekilde yapılabilir.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. URAL.
 AUTHOR. GULNUR AKYOL.
 DATE-WRITTEN. 09-25-1989.
 DATE-COMPILED. 09-26-1989.
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT URAL1 ASSIGN TO DISK
     ORGANIZATION IS INDEXED
     ACCESS MODE IS DYNAMIC
```

```
    RECORD KEY IS SIC-NO
    ALTERNATE RECORD KEY IS AD-SOYAD WITH DUPLICATES
    FILE STATUS IS URAL1-HATA.
    SELECT  URAL2 ASSIGN TO DISK
    ORGANIZATION IS INDEXED
    ACCESS MODE IS DYNAMIC
    RECORD KEY IS MAAS-KAT
    ALTERNATE RECORD KEY IS Y-OD-KAT WITH DUPLICATES
    FILE STATUS IS URAL2-HATA.
    SELECT URAL3 ASSIGN TO DISK
    FILE STATUS IS URAL3-HATA.
    SELECT URAL4 ASSIGN TO PRINTER.
DATA DIVISION.
FILE SECTION.
FD URAL1 LABEL RECORD IS STANDARD
        VALUE OF FILE-ID "BORDRO.DAT".
01 BORD-KAYIT.
    02 IS-YER-KOD     PIC X(2).
    02 BOL-KOD        PIC X(2).
    02 BIL-GIR-TAR.
       03 BGUN         PIC 9(2).
       03 BAY          PIC 9(2).
       03 BYIL         PIC 9(2).
    02 SIC-NO         PIC 9(4).
    02 AD-SOYAD       PIC X(15).
    02 ADR.
       03 SEMT         PIC X(10).
       03 SOKAK        PIC X(14).
       03 NUMARA       PIC 9(2).
    02 TEL            PIC 9(7).
    02 DOG-YER        PIC X(10).
    02 BABA-ADI       PIC X(10).
    02 IL-ILCE.
       03 IL           PIC X(10).
       03 ILCE         PIC X(8).
    02 KOY-MAH        PIC X(10).
    02 HANE-CILT.
```

```
      03 HANE          PIC X(3).
      03 CILT          PIC X(2).
   02 IS-GIR-TAR.
      03 GGUN          PIC 9(2).
      03 GAY           PIC 9(2).
      03 GYIL          PIC 9(2).
   02 IS-CIK-TAR.
      03 CGUN          PIC 9(2).
      03 CAY           PIC 9(2).
      03 CYIL          PIC 9(2).
   02 IS-CIK-NEDEN     PIC X(15).
   02 SENDIKA          PIC X(10).
   02 YAN-OD-PUANI     PIC 9(4)V99.
   02 GOREV            PIC X(8).
   02 MED-HALI         PIC X.
      88 EVLI          VALUE "E" "e".
      88 BEKAR         VALUE "B" "b".
   02 E-DER-KADROSU    PIC X(8).
   02 E-GOSTERGE       PIC 9(4).
   02 G-DER-KADROSU    PIC X(3).
   02 G-GOSTERGE       PIC 9(5).
   02 EK-GOSTERGE      PIC 9(4).
   02 COCUK-SAYI       PIC 9.
   02 SAKAT-DERECE     PIC 9.
   02 TOP-GEL-VERGI    PIC 9(7).
   02 KESILENLER.
      03 ICRA          PIC 9(7).
      03 IKRAZ         PIC 9(7).
      03 KEFALET       PIC 9(7).
      03 KG-GOST       PIC 9(5).
      03 E-BORCU       PIC 9(7).
FD URAL2 LABEL RECORD IS STANDARD
      VALUE OF FILE-ID "KATSAYI.DAT".
01 KAT-KAYIT.
   02 MAAS-KAT         PIC 9(4).
   02 Y-OD-KAT         PIC 9(5).
   02 O-HIZ-YUZ        PIC 9(2).
```

```
    02 MUST-MAAS      PIC 9(7).
    02 YAK-YAR        PIC 9(7).
    02 KON-ED         PIC 9(7).
    02 OZ-IN          PIC 9(7).
FD URAL3 LABEL RECORD IS STANDARD
        VALUE OF FILE-ID "STOK.DAT".
01 MAAS-KAT1           PIC 9(4).
FD URAL4 LABEL RECORD IS OMITTED.
01 SATIR               PIC X(80).
WORKING-STORAGE SECTION.
77 FTUS     PIC X(2).
    88 ESC          VALUE "01".
    88 F1           VALUE "02".
    88 F2           VALUE "03".
    88 F3           VALUE "04".
    88 F4           VALUE "05".
    88 F5           VALUE "06".
77 SEC          PIC X VALUE SPACE.
77 CIZGI        PIC X(80) VALUE ALL "-".
77 BOS          PIC X(80) VALUE ALL " ".
77 URAL1-HATA   PIC X(2) VALUE SPACES.
77 URAL2-HATA   PIC X(2) VALUE SPACES.
77 URAL3-HATA   PIC X(2) VALUE SPACES.
77 AAA          PIC X(10) VALUE SPACES.
77 BBB          PIC X(11) VALUE SPACES.
77 D            PIC 99    VALUE 0.
77 K            PIC 99    VALUE 0.
77 I            PIC 9(3) VALUE 0.
77 ALT-KEY      PIC X(3) VALUE SPACES.
77 YOD          PIC 9(7) VALUE 0.
77 BMAAS        PIC 9(7) VALUE 0.
77 OZHIZ        PIC 9(7) VALUE 0.
77 AILEYAR      PIC 9(7) VALUE 0.
77 COYAR        PIC 9(7) VALUE 0.
77 KESEP        PIC 9(7) VALUE 0.
77 KESEP1       PIC 9(7) VALUE 0.
77 TASKES       PIC 9(7) VALUE 0.
```

```
77 DAMVER       PIC 9(7) VALUE 0.
77 MATRAH       PIC 9(7) VALUE 0.
77 TAS-TES-PAR PIC 9(7) VALUE 0.
77 KESGEVER     PIC 9(7) VALUE 0.
77 VERILEN      PIC 9(7) VALUE 0.
77 KESILEN      PIC 9(7) VALUE 0.
77 NET-MAAS     PIC 9(7) VALUE 0.
01 ESR1.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "SICIL NO.....: ".
    02 SNO1     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " ADI SOYADI.....: ".
    02 ADS      PIC X(15) VALUE SPACES.
    02 F        PIC X(8) VALUE SPACES.
01 ESR2.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "BRUT MAAŞ.....: ".
    02 BUCR     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " KESILEN GELIR VERGISI.: ".
    02 KGVE     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR3.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "YAN ÖDEME.....: ".
    02 YOD1     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " DAMGA VERGISI.....: ".
    02 DMGV     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR4.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "ÖZEL HIZMET.....: ".
    02 OZHZ     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " EMEKLI BORCU.....: ".
    02 EBRC     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR5.
    02 F        PIC X(5) VALUE SPACES.
```

```
    02 F        PIC X(20) VALUE "YAKACAK YARDIMI...: ".
    02 YKYR     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " İCRA.................: ".
    02 ICR1     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR6.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "AİLE YARDIMI......: ".
    02 AYAR     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " İKRAZ................: ".
    02 IKRZ     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR7.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "ÇOCUK YARDIMI.....: ".
    02 COYR     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " KEFALET..............: ".
    02 KFLT     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR8.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "EMEKLİ YARDIMI....: ".
    02 KSP2     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " KONUT EDİNME.........: ".
    02 KNE2     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR9.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "KONUT EDİNME......: ".
    02 KNE1     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(25) VALUE " KESİLEN TASARRUF K....: ".
    02 TKS1     PIC Z(7) VALUE ZEROS.
    02 F        PIC X(16) VALUE SPACES.
01 ESR10.
    02 F        PIC X(5) VALUE SPACES.
    02 F        PIC X(20) VALUE "TASARRUF TEŞVİK PA: ".
    02 TTP1     PIC Z(7) VALUE ZEROS.
```

```
    02 F      PIC X(25) VALUE " ÖZEL İNDİRİM.....: ".
    02 OZN1   PIC Z(7) VALUE ZEROS.
    02 F      PIC X(16) VALUE SPACES.
01 ESR11.
    02 F      PIC X(5) VALUE SPACES.
    02 F      PIC X(20) VALUE "TOPLAM VERİLEN MİK: ".
    02 VRL1   PIC Z(7) VALUE ZEROS.
    02 F      PIC X(25) VALUE " TOPLAM KESİLEN MİK....: ".
    02 KSL1   PIC Z(7) VALUE ZEROS.
    02 F      PIC X(16) VALUE SPACES.
01 ESR12.
    02 F      PIC X(26) VALUE SPACES.
    02 F      PIC X(21) VALUE "*****NET MAAŞ***** : ".
    02 NMAAS  PIC Z(7) VALUE ZEROS.
    02 F      PIC X(26) VALUE SPACES.
SCREEN SECTION.
01 UST-CER.
    02 LINE 1  COLUMN 1  VALUE "┌".
    02 LINE 1  COLUMN 2  PIC X(78) USING CIZGI.
    02 LINE 1  COLUMN 80 VALUE "┐".
    02 LINE 2  COLUMN 1  VALUE "│".
    02 LINE 2  COLUMN 80 VALUE "│".
    02 LINE 3  COLUMN 1  VALUE "└".
    02 LINE 3  COLUMN 2  PIC X(78) USING CIZGI.
    02 LINE 3  COLUMN 80 VALUE "┘".
01 ALT-CER.
    02 LINE 23 COLUMN 1  VALUE "┌".
    02 LINE 23 COLUMN 2  PIC X(77) USING CIZGI.
    02 LINE 23 COLUMN 79 VALUE "┐".
    02 LINE 24 COLUMN 1  VALUE "│".
    02 LINE 24 COLUMN 79 VALUE "│".
    02 LINE 25 COLUMN 1  VALUE "└".
    02 LINE 25 COLUMN 2  PIC X(77) USING CIZGI.
    02 LINE 25 COLUMN 79 VALUE "┘".
01 OZ-CER.
    02 LINE 1  COLUMN 1  VALUE "┌".
    02 LINE 1  COLUMN 2  PIC X(77) USING CIZGI.
```

```
      02 LINE 1  COLUMN 79 VALUE "┐".
      02 LINE 9  COLUMN 1  VALUE "└".
      02 LINE 9  COLUMN 2  PIC X(77) USING CIZGI.
      02 LINE 9  COLUMN 79 VALUE "┘".
  01 OZ-CER1.
      02 LINE 10 COLUMN 1  VALUE "┌".
      02 LINE 10 COLUMN 2  PIC X(77) USING CIZGI.
      02 LINE 10 COLUMN 79 VALUE "┐".
      02 LINE 22 COLUMN 1  VALUE "└".
      02 LINE 22 COLUMN 2  PIC X(77) USING CIZGI.
      02 LINE 22 COLUMN 79 VALUE "┘".
  01 MESAJ.
      02 LINE 24 COLUMN 2  VALUE "MESAJ:" REVERSE-VIDEO.
      02 LINE 24 COLUMN 17  VALUE "SEÇİMİNİZ  ‹F1›,‹F2›,‹F3›
-       ",‹F4›.......[.]".
  01 SMESAJ.
      02 LINE 24 COLUMN 56 PIC X USING SEC  AUTO.
  01 MESAJ1.
      02 LINE 24 COLUMN 12 VALUE "‹F1› ANA MENU    ‹F2› DEV
-       "AM    ‹F3› IPTAL    ‹F4› DUZELTME [KAYIT]".
  01 MESAJ1KAY.
      02 LINE 24 COLUMN 72 VALUE "KAYIT" BLINK.
  01 ANA-MENU.
      02 LINE 9 COLUMN 21 VALUE "* * * * * A N A   M E N U
-             "* * * * *" BLINK.
      02 COLUMN 31 VALUE "A N A   M E N U"  REVERSE-VIDEO .
      02 LINE 11 COLUMN 24 VALUE "‹F1›  ÖZLÜK BİLGİLER....".
      02 COLUMN 25 VALUE "F1" REVERSE-VIDEO BLINK.
      02 LINE 13 COLUMN 24 VALUE "‹F2›  KATSAYI BİLGİLERİ.".
      02 COLUMN 25 VALUE "F2" REVERSE-VIDEO BLINK.
      02 LINE 15 COLUMN 24 VALUE "‹F3›  MAAŞ HESABI.......".
      02 COLUMN 25 VALUE "F3" REVERSE-VIDEO BLINK.
      02 LINE 17 COLUMN 24 VALUE "‹F4›  PROGRAM SONU......".
      02 COLUMN 25 VALUE "F4" REVERSE-VIDEO BLINK.
  01 OZ-BIL.
      02 LINE 2  COLUMN 2  VALUE "SICIL NUMARASI    :".
      02 LINE 3  COLUMN 2  VALUE "ADI SOYADI        :".
```

```
    02 LINE 4  COLUMN 2  VALUE "İKAMET ETTİĞİ SEMT:".
    02 LINE 5  COLUMN 2  VALUE "                SOKAK:".
    02 LINE 6  COLUMN 2  VALUE "             NUMARA:".
    02 LINE 7  COLUMN 2  VALUE "TELEFON            :".
    02 LINE 8  COLUMN 2  VALUE "TARİH              :   /  /19".
    02 LINE 2  COLUMN 41 VALUE "DOĞUM YERI         :".
    02 LINE 3  COLUMN 41 VALUE "BABA ADI           :".
    02 LINE 4  COLUMN 41 VALUE "NUF.KAY.OLD. İL    :".
    02 LINE 5  COLUMN 41 VALUE "              İLÇE :".
    02 LINE 6  COLUMN 41 VALUE "          KÖY-MAH. :".
    02 LINE 7  COLUMN 41 VALUE "             HANE  :".
    02 LINE 8  COLUMN 41 VALUE "             CİLT  :".
01 OZ-BIL1.
    02 LINE 11 COLUMN 2  VALUE "İŞ YERİ KODU       :".
    02 LINE 12 COLUMN 2  VALUE "BÖLUM KODU         :".
    02 LINE 13 COLUMN 2  VALUE "GÖREV UNVANI       :".
    02 LINE 14 COLUMN 2  VALUE "SENDİKASI          :".
    02 LINE 15 COLUMN 2  VALUE "MEDENİ HALİ [E/B]  :".
    02 LINE 16 COLUMN 2  VALUE "ÇOCUK SAYISI       :".
    02 LINE 17 COLUMN 2  VALUE "SAKATLIK DERECESİ  :".
    02 LINE 18 COLUMN 2  VALUE "İŞE GİRİŞ TARİHİ   :   /  /
-                              "19  ".
    02 LINE 19 COLUMN 2  VALUE "İŞTEN ÇIKIŞ TARİHİ:   /  /
-                              "19  ".
    02 LINE 20 COLUMN 2  VALUE "İŞTEN ÇIKIŞ NEDENİ:".
    02 LINE 21 COLUMN 2  VALUE "YAN ÖDEME PUANI    :".
    02 LINE 11 COLUMN 41 VALUE "GÖREV DERECE KADR.:".
    02 LINE 12 COLUMN 41 VALUE "EMEK.DERECE KADR.  :".
    02 LINE 13 COLUMN 41 VALUE "GÖREV GÖSTERGESİ   :".
    02 LINE 14 COLUMN 41 VALUE "EMEK. GÖSTERGESİ   :".
    02 LINE 15 COLUMN 41 VALUE "EK GÖSTERGE        :".
    02 LINE 16 COLUMN 41 VALUE "TOP.GELİR VERGİSİ  :".
    02 LINE 17 COLUMN 41 VALUE "KESİLENLER;İCRA    :".
    02 LINE 18 COLUMN 41 VALUE "                İKRAZ  :".
    02 LINE 19 COLUMN 41 VALUE "              KEFALET:".
    02 LINE 20 COLUMN 41 VALUE "            GÖR.GOS.:".
    02 LINE 21 COLUMN 41 VALUE "           EMEK.BOR.:".
```

```
01 OZ-OKU.
    02 LINE 2  COLUMN 22 PIC Z(4) USING SIC-NO UNDERLINE.
01 OZ-OKU1.
    02 LINE 3  COLUMN 22 PIC X(15) USING AD-SOYAD UNDERLINE.
01 OZ-OKU2.
    02 ADRES.
       03 LINE 4  COLUMN 22 PIC X(10) USING SEMT UNDERLINE.
       03 LINE 5  COLUMN 22 PIC X(14) USING SOKAK UNDERLINE.
       03 LINE 6  COLUMN 22 PIC Z(2)  USING NUMARA UNDERLINE.
    02 LINE 7  COLUMN 22 PIC Z(7) USING TEL UNDERLINE.
    02 TARIH.
       03 LINE 8  COLUMN 22 PIC Z(2) USING BGUN UNDERLINE.
       03 COLUMN 25 PIC Z(2) USING BAY UNDERLINE.
       03 COLUMN 30 PIC Z(2) USING BYIL UNDERLINE.
    02 LINE 2 COLUMN 62 PIC X(10) USING DOG-YER UNDERLINE.
    02 LINE 3 COLUMN 62 PIC X(10) USING BABA-ADI UNDERLINE.
    02 IL-ILCE1.
       03 LINE 4  COLUMN 62 PIC X(10) USING IL UNDERLINE.
       03 LINE 5  COLUMN 62 PIC X(8) USING ILCE UNDERLINE.
    02 LINE 6  COLUMN 62 PIC X(10) USING KOY-MAH UNDERLINE.
    02 HANE-CILT1.
       03 LINE 7  COLUMN 62 PIC X(3) USING HANE UNDERLINE.
       03 LINE 8  COLUMN 62 PIC X(2) USING CILT UNDERLINE.
    02 LINE 11 COLUMN 22 PIC X(2) USING IS-YER-KOD UNDERLINE.
    02 LINE 12 COLUMN 22 PIC X(2) USING BOL-KOD UNDERLINE.
    02 LINE 13 COLUMN 22 PIC X(8) USING GOREV UNDERLINE.
    02 LINE 14 COLUMN 22 PIC X(10) USING SENDIKA UNDERLINE.
    02 LINE 15 COLUMN 22 PIC X USING MED-HALI UNDERLINE.
    02 LINE 16 COLUMN 22 PIC 9 USING COCUK-SAYI UNDERLINE.
    02 LINE 17 COLUMN 22 PIC 9 USING SAKAT-DERECE UNDERLINE.
    02 TARIH1.
       03 LINE 18 COLUMN 22 PIC Z(2) USING GGUN UNDERLINE.
       03 COLUMN 25 PIC Z(2) USING GAY UNDERLINE.
       03 COLUMN 30 PIC Z(2) USING GYIL UNDERLINE.
    02 TARIH2.
       03 LINE 19 COLUMN 22 PIC Z(2) USING CGUN UNDERLINE.
       03 COLUMN 25 PIC Z(2) USING CGUN UNDERLINE.
```

```
           03 COLUMN 30 PIC Z(2) USING CYIL UNDERLINE.
        02 LINE 20 COLUMN 22 PIC X(15) USING IS-CIK-NEDEN
                                          UNDERLINE.
        02 LINE 21 COLUMN 22 PIC Z(4).ZZ USING YAN-OD-PUANI
                                             UNDERLINE.
        02 LINE 11 COLUMN 62 PIC X(3) USING G-DER-KADROSU
                                             UNDERLINE.
        02 LINE 12 COLUMN 62 PIC X(8) USING E-DER-KADROSU
                                             UNDERLINE.
        02 LINE 13 COLUMN 62 PIC Z(5) USING G-GOSTERGE
                                             UNDERLINE.
        02 LINE 14 COLUMN 62 PIC Z(4) USING E-GOSTERGE
                                             UNDERLINE.
        02 LINE 15 COLUMN 62 PIC Z(4) USING EK-GOSTERGE
                                             UNDERLINE.
        02 LINE 17 COLUMN 62 PIC Z(7) USING ICRA UNDERLINE.
        02 LINE 18 COLUMN 62 PIC Z(7) USING IKRAZ UNDERLINE.
        02 LINE 19 COLUMN 62 PIC Z(7) USING KEFALET UNDERLINE.
        02 LINE 20 COLUMN 62 PIC Z(5) USING KG-GOST UNDERLINE.
        02 LINE 21 COLUMN 62 PIC Z(7) USING E-BORCU UNDERLINE.
    01 TOPVER.
        02 LINE 16 COLUMN 62 PIC Z(7) USING TOP-GEL-VERGI
                                             UNDERLINE.
    01 MESAJ1SI.
        02 LINE 24 COLUMN 8 PIC X(70) USING BOS.
    01 MESAJ2.
        02 LINE 24 COLUMN 8 VALUE "SİLİNMEK İSTENEN KAYIT BULU
    -     "NAMAMIŞTIR   !........" BLINK.
    01 MESAJ3.
        02 LINE 24 COLUMN 12 VALUE "GİRİLEN BİLGİLER KAYIT ED
    -       "İLECEKMİ [E/H]....!  [.]" BLINK.
    01 MESAJ4.
        02 LINE 24 COLUMN 25 VALUE "‹F1› ANA MENU   ‹F2› KAYIT".
        02 COLUMN 26 VALUE "F1" REVERSE-VIDEO.
        02 COLUMN 42 VALUE "F2" REVERSE-VIDEO.
    01 MESAJ5.
        02 LINE 24 COLUMN 12 VALUE " ‹F1› ANA MENU  ‹F2› DEVAM
    -       "  ‹F3› YAZICI     SECİMİNİZ [.]".
```

```
           02 COLUMN 14 VALUE "F1" REVERSE-VIDEO.
           02 COLUMN 29 VALUE "F2" REVERSE-VIDEO.
           02 COLUMN 41 VALUE "F3" REVERSE-VIDEO.
           02 COLUMN 68 VALUE "." BLINK.
       01 MESAJ6.
           02 LINE 24 COLUMN 19 VALUE "BORDROSU İSTENEN KIŞININ K
      -        "AYDI GİRİLMEMİŞTİR.....!".
       01 MESAJ7.
           02 LINE 24 COLUMN 19 VALUE "DOSYADA HİÇ BİLGİ YOKTUR.A
      -         "NA MENUYE DÖNÜLÜYOR.....!".
       01 ICICE.
           02 LINE 1 COLUMN 7 VALUE "┌".
           02 COLUMN 8 PIC X(68) USING CIZGI.
           02 COLUMN 76 VALUE "┐".
           02 LINE 20 COLUMN 7 VALUE "└".
           02 COLUMN 8 PIC X(68) USING CIZGI.
           02 COLUMN 76 VALUE "┘".
           02 LINE 2 COLUMN 9 VALUE "┌".
           02 COLUMN 10 PIC X(64) USING CIZGI.
           02 COLUMN 74 VALUE "┐".
           02 LINE 19 COLUMN 9 VALUE "└".
           02 COLUMN 10 PIC X(64) USING CIZGI.
           02 COLUMN 74 VALUE "┘".
           02 LINE 3 COLUMN 11 VALUE "┌".
           02 COLUMN 12 PIC X(60) USING CIZGI.
           02 COLUMN 72 VALUE "┐".
           02 LINE 18 COLUMN 11 VALUE "└".
           02 COLUMN 12 PIC X(60) USING CIZGI.
           02 COLUMN 72 VALUE "┘".
       01 ORBIL.
           02 LINE 5  COLUMN 21 VALUE "MAAŞ KATSAYISI      :".
           02 LINE 7  COLUMN 21 VALUE "YAN ÖDEME KATSAYISI:".
           02 LINE 9  COLUMN 21 VALUE "ÖZEL HİZMET YUZDESİ:".
           02 LINE 11 COLUMN 21 VALUE "MÜSTEŞAR MAAŞI      :".
           02 LINE 13 COLUMN 21 VALUE "YAKACAK YARDIMI     :".
           02 LINE 15 COLUMN 21 VALUE "KONUT EDİNME YARD.  :".
           02 LINE 17 COLUMN 21 VALUE "ÖZEL İNDİRİM        :".
```

```
01 OKUORT.
    02 LINE 5  COLUMN 42 PIC Z(4) USING MAAS-KAT UNDERLINE.
01 OKUORT1.
    02 LINE 7  COLUMN 42 PIC Z(5) USING Y-OD-KAT UNDERLINE.
    02 LINE 9  COLUMN 42 PIC Z(2) USING O-HIZ-YUZ UNDERLINE.
    02 LINE 11 COLUMN 42 PIC Z(7) USING MUST-MAAS UNDERLINE.
    02 LINE 13 COLUMN 42 PIC Z(7) USING YAK-YAR UNDERLINE.
    02 LINE 15 COLUMN 42 PIC Z(7) USING KON-ED UNDERLINE.
    02 LINE 17 COLUMN 42 PIC Z(7) USING OZ-IN UNDERLINE.
01 BORCER.
    02 LINE 3  COLUMN 1  VALUE "┌".
    02 COLUMN 2 PIC X(77) USING CIZGI.
    02 COLUMN 79 VALUE "┐".
    02 LINE 4  COLUMN 1  VALUE "│".
    02 COLUMN 79 VALUE "│".
    02 LINE 5  COLUMN 1 VALUE "└".
    02 COLUMN 2 PIC X(77) USING CIZGI.
    02 COLUMN 79 VALUE "┘".
    02 LINE 7  COLUMN 1  VALUE "┌".
    02 COLUMN 2 PIC X(77) USING CIZGI.
    02 COLUMN 79 VALUE "┐".
    02 LINE 18 COLUMN 2 PIC X(77) USING CIZGI.
    02 LINE 21 COLUMN 1 VALUE "└".
    02 COLUMN 2 PIC X(77) USING CIZGI.
    02 COLUMN 79 VALUE "┘".
01 MAASHB.
    02 LINE 4 COLUMN 2 VALUE "SİCİL NO                :".
    02 COLUMN 41 VALUE "ADI SOYADI                :".
    02 LINE 9  COLUMN 2  VALUE "BRUT MAAŞ                :".
    02 LINE 10 COLUMN 2  VALUE "YAN ÖDEME                :".
    02 LINE 11 COLUMN 2  VALUE "OZEL HİZMET              :".
    02 LINE 12 COLUMN 2  VALUE "YAKACAK YARDIMI          :".
    02 LINE 13 COLUMN 2  VALUE "AİLE YARDIMI             :".
    02 LINE 14 COLUMN 2  VALUE "ÇOCUK YARDIMI            :".
    02 LINE 15 COLUMN 2  VALUE "EMEKLİ YARDIMI(%20):".
    02 LINE 16 COLUMN 2  VALUE "KONUT EDİNME             :".
    02 LINE 17 COLUMN 2  VALUE "TASARRUF TEŞVİK PA.:".
```

```
    02 LINE 19 COLUMN 2  VALUE "TOPLAM VERILEN MIK.:".
    02 LINE 9  COLUMN 41 VALUE "KESILEN GELIR VER. :".
    02 LINE 10 COLUMN 41 VALUE "DAMGA VERGISI      :".
    02 LINE 11 COLUMN 41 VALUE "EMEKLI BORCU       :".
    02 LINE 12 COLUMN 41 VALUE "ICRA               :".
    02 LINE 13 COLUMN 41 VALUE "IKRAZ              :".
    02 LINE 14 COLUMN 41 VALUE "KEFALET            :".
    02 LINE 15 COLUMN 41 VALUE "KONUT EDINME       :".
    02 LINE 16 COLUMN 41 VALUE "KESILEN TASARRUF K.:".
    02 LINE 17 COLUMN 41 VALUE "OZEL INDIRIM       :".
    02 LINE 19 COLUMN 41 VALUE "TOPLAM KESILEN MIK.:".
    02 LINE 20 COLUMN 23 VALUE "*** NET  MAAS ***  ".
01 BITIS.
    02 LINE 9  COLUMN 24 PIC Z(7) USING BMAAS UNDERLINE.
    02 LINE 10 COLUMN 24 PIC Z(7) USING YOD UNDERLINE.
    02 LINE 11 COLUMN 24 PIC Z(7) USING OZHIZ UNDERLINE.
    02 LINE 12 COLUMN 24 PIC Z(7) USING YAK-YAR UNDERLINE.
    02 LINE 13 COLUMN 24 PIC Z(7) USING AILEYAR UNDERLINE.
    02 LINE 14 COLUMN 24 PIC Z(7) USING COYAR UNDERLINE.
    02 LINE 15 COLUMN 24 PIC Z(7) USING KESEP UNDERLINE.
    02 LINE 16 COLUMN 24 PIC Z(7) USING KON-ED UNDERLINE.
    02 LINE 17 COLUMN 24 PIC Z(7) USING TAS-TES-PAR UNDERLINE.
    02 LINE 19 COLUMN 24 PIC Z(7) USING VERILEN UNDERLINE.
    02 LINE 9  COLUMN 62 PIC Z(7) USING KESGEVER UNDERLINE.
    02 LINE 10 COLUMN 62 PIC Z(7) USING DAMVER UNDERLINE.
    02 LINE 11 COLUMN 62 PIC Z(7) USING E-BORCU UNDERLINE.
    02 LINE 12 COLUMN 62 PIC Z(7) USING ICRA UNDERLINE.
    02 LINE 13 COLUMN 62 PIC Z(7) USING IKRAZ UNDERLINE.
    02 LINE 14 COLUMN 62 PIC Z(7) USING KEFALET UNDERLINE.
    02 LINE 15 COLUMN 62 PIC Z(7) USING KON-ED UNDERLINE.
    02 LINE 16 COLUMN 62 PIC Z(7) USING TASKES UNDERLINE.
    02 LINE 17 COLUMN 62 PIC Z(7) USING OZ-IN UNDERLINE.
    02 LINE 19 COLUMN 62 PIC Z(7) USING KESILEN UNDERLINE.
01 HES.
    02 LINE 4 COLUMN 23 PIC Z(4) USING SIC-NO UNDERLINE.
01 HES1.
    02 LINE 4 COLUMN 62 PIC X(15) USING AD-SOYAD UNDERLINE.
```

```
01 YILDIZ.
    02 LINE 20 COLUMN 23 VALUE "***" BLINK.
    02 COLUMN 37 VALUE "***" BLINK.
01 NMAAS-1.
    02 LINE 20 COLUMN 42 PIC Z(7) USING NET-MAAS REVERSE-VIDEO
01 OKUMAAS.
    02 OKUMAAS1.
      03 LINE 4  COLUMN 23 PIC Z(4) USING SIC-NO UNDERLINE.
    02 OKUMAAS2.
      03 LINE 4 COLUMN 62 PIC X(15) USING AD-SOYAD UNDERLINE.
PROCEDURE DIVISION.
BASLA.
    MOVE "BORDRO.DAT" TO AAA.
    CALL "EXIST" USING AAA , URAL1-HATA.
    MOVE "KATSAYI.DAT" TO BBB.
    CALL "EXIST" USING BBB , URAL2-HATA.
    IF URAL1-HATA = "30"  OPEN OUTPUT URAL1 CLOSE URAL1.
    OPEN I-O URAL1.
    IF URAL2-HATA = "30" OPEN OUTPUT URAL2 CLOSE URAL2.
    OPEN I-O URAL2.
MENU.
    MOVE " " TO SEC.
    DISPLAY ANA-MENU ALT-CER MESAJ1SI MESAJ.
BASLA1.
    ACCEPT SMESAJ.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF F1 GO BIR.
    IF F2 GO IKI.
    IF F3 GO UC.
    IF F4 DISPLAY (1 1) ERASE CLOSE URAL1 URAL2 URAL3 STOP RUN
    GO BASLA1.
BIR.
    PERFORM SIL.
    PERFORM SIFIRLA.
    DISPLAY OZ-CER.
    PERFORM OZ-CERD VARYING LIN FROM 2 BY 1 UNTIL LIN > 8.
    DISPLAY MESAJ1SI MESAJ1 MESAJ1KAY.
```

```
           DISPLAY OZ-BIL OZ-CER1.
           PERFORM OZ-CERD VARYING LIN FROM 11 BY 1 UNTIL LIN > 21.
       DEVAM.
           PERFORM SIFIRLA.
           DISPLAY OZ-BIL1.
           DISPLAY OZ-OKU OZ-OKU1 OZ-OKU2.
           ACCEPT OZ-OKU.
           ACCEPT FTUS FROM ESCAPE KEY.
           IF F1 PERFORM SIL GO MENU.
           IF SIC-NO = 0  GO ALTER1.
           START URAL1 KEY IS EQUAL TO SIC-NO INVALID KEY GO BIR-1.
           READ URAL1 NEXT.
       BIR-1.
           DISPLAY OZ-OKU OZ-OKU1 OZ-OKU2 TOPVER.
           IF ALT-KEY = "GUL"  MOVE SPACES TO ALT-KEY GO BIR-11.
           ACCEPT OZ-OKU1.
           ACCEPT FTUS FROM ESCAPE KEY.
           IF F1 GO KAYIT1.
       BIR-11.
           ACCEPT OZ-OKU2.
           ACCEPT FTUS FROM ESCAPE KEY.
           IF F3 GO BIR-2.
           IF F4 GO DUZ1.
           IF F1 GO KAYIT1.
           IF F2 GO KAYIT.
       KAYIT.
           WRITE BORD-KAYIT INVALID KEY REWRITE BORD-KAYIT GO DEVAM.
           GO DEVAM.
       KAYIT1.
           DISPLAY MESAJ1SI MESAJ3.
           ACCEPT (24 59) SEC WITH AUTO-SKIP.
           IF SEC = "E" OR = "e"  WRITE BORD-KAYIT
                                  INVALID KEY REWRITE BORD-KAYIT
           PERFORM SIL. DISPLAY MESAJ1SI. GO MENU.
      ********** B İ R   K A Y I D I   S İ L E R **********
       BIR-2.
           DELETE URAL1 INVALID KEY GO BIR-3.
```

```
     GO BIR-4.
 BIR-3.
     DISPLAY MESAJ1SI MESAJ2.
     PERFORM SAY.
     DISPLAY MESAJ1SI MESAJ1 MESAJ1KAY.
 BIR-4.
     GO DEVAM.
********************KAYIT DÜZELTME*****************************
 DUZ1.
     REWRITE BORD-KAYIT INVALID KEY GO KAYIT.
     GO DEVAM.
******************** ALTERNE KEY'E GÖRE ARAMA ***************
 ALTER1.
     ACCEPT OZ-OKU1.
     ACCEPT FTUS FROM ESCAPE KEY.
     IF F1 PERFORM SIL GO MENU.
     IF AD-SOYAD = SPACES GO DEVAM.
     START URAL1 KEY IS EQUAL TO AD-SOYAD INVALID KEY GO DEVAM.
     READ URAL1 NEXT.
     MOVE "GUL" TO ALT-KEY.
     GO BIR-1.
*****************************************************************
******************ORTAK BİLGİLERİN GİRİŞİ************************
 IKI.
     PERFORM SIL.
     DISPLAY MESAJ1SI ICICE.
     MOVE 1 TO K.
     PERFORM AA1 VARYING I FROM 2 BY 2 UNTIL I > 6.
 IKI1.
     PERFORM SIFIRLA1.
     DISPLAY MESAJ4 ORBIL OKUORT OKUORT1.
     ACCEPT OKUORT.
     ACCEPT FTUS FROM ESCAPE KEY.
     IF F1 PERFORM SIL GO MENU.
     START URAL2 KEY IS EQUAL TO MAAS-KAT INVALID KEY GO IKI2.
     READ URAL2 NEXT.
     DISPLAY OKUORT1.
```

```
IKI2.
    ACCEPT OKUORT1.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF F1 GO BIR7.
    IF F2 GO BIR8.
    IF FTUS = "00" GO BIR9.
BIR7.
    DISPLAY MESAJ1SI MESAJ3.
    ACCEPT (24 59) SEC WITH AUTO-SKIP.
    IF SEC = "E" OR = "e" GO BIR6 ELSE GO BIR5.
BIR6.
    PERFORM YENIDOSYA.
    WRITE KAT-KAYIT INVALID KEY REWRITE KAT-KAYIT.
BIR5.
    PERFORM SIL. DISPLAY MESAJ1SI.
    GO MENU.
BIR8.
    PERFORM YENIDOSYA.
    WRITE KAT-KAYIT INVALID KEY REWRITE KAT-KAYIT.
    GO IKI1.
BIR9.
    DISPLAY MESAJ1SI MESAJ3.
    ACCEPT (24 59) SEC WITH AUTO-SKIP.
    IF SEC = "E" OR = "e" PERFORM YENIDOSYA
            WRITE KAT-KAYIT INVALID KEY REWRITE KAT-KAYIT.
    DISPLAY MESAJ1SI MESAJ4.
    GO IKI1.
UC.
    OPEN INPUT URAL3.
    IF URAL3-HATA = "30" DISPLAY MESAJ1SI MESAJ7
                    PERFORM SAY GO MENU.
    READ URAL3.
    MOVE MAAS-KAT1 TO MAAS-KAT.
    START URAL2 KEY IS EQUAL TO MAAS-KAT.
    READ URAL2 NEXT.
    PERFORM SIL.
    DISPLAY MESAJ1SI BORCER MESAJ5.
```

```
    PERFORM OZ-CERD VARYING LIN FROM 8 BY 1 UNTIL LIN > 20.
UC1.
    DISPLAY BITIS.
    MOVE 0 TO SIC-NO.
    MOVE SPACES TO AD-SOYAD.
    DISPLAY MAASHB YILDIZ HES HES1.
    ACCEPT  HES.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF F1 PERFORM SIL CLOSE URAL3 GO MENU.
    IF SIC-NO = 0   GO UC-ALT.
    START URAL1 KEY IS EQUAL TO SIC-NO INVALID KEY GO ESRA.
    READ URAL1 NEXT.
    GO  HESAPLA.
ESRA.
    DISPLAY MESAJ1SI MESAJ6.
    PERFORM SAY.
    DISPLAY MESAJ5.
    GO UC1.
UC-ALT.
    ACCEPT HES1.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF F1 PERFORM SIL CLOSE URAL3 GO MENU.
    START URAL1 KEY IS EQUAL TO AD-SOYAD INVALID KEY GO ESRA.
    READ URAL1 NEXT.
HESAPLA.
    DISPLAY HES HES1.
    COMPUTE BMAAS = (G-GOSTERGE + EK-GOSTERGE) * MAAS-KAT.
    COMPUTE YOD = Y-OD-KAT * YAN-OD-PUANI.
    COMPUTE OZHIZ = O-HIZ-YUZ * MUST-MAAS.
    IF MED-HALI = "B" GO HESAPLA1.
    COMPUTE AILEYAR = MAAS-KAT * 25.
    IF COCUK-SAYI = 1 COMPUTE COYAR = MAAS-KAT * 25 GO HESAPLA1.
    COMPUTE COYAR = MAAS-KAT * 50.
HESAPLA1.
    COMPUTE TAS-TES-PAR = (BMAAS + YOD + OZHIZ + YAK-YAR) * .045
    COMPUTE KESEP = (E-GOSTERGE + EK-GOSTERGE) * MAAS-KAT * 0.2.
    COMPUTE KESEP1 = (E-GOSTERGE + EK-GOSTERGE) * MAAS-KAT * .12
```

```
    COMPUTE TASKES = (BMAAS + YOD + YAK-YAR + OZHIZ) * 0.075.
    COMPUTE DAMVER = (BMAAS + YOD + YAK-YAR + OZHIZ) * 0.004.
    COMPUTE MATRAH = (BMAAS + YOD + YAK-YAR) * 0.03.
    IF SAKAT-DERECE = 2 COMPUTE OZ-IN = OZ-IN * 2 GO HESAPLA2
    IF SAKAT-DERECE = 3 COMPUTE OZ-IN = OZ-IN * 3 .
HESAPLA2.
    COMPUTE KESGEVER = TOP-GEL-VERGI + BMAAS + OZHIZ - OZ-IN
                - KESEP1 - MATRAH.
    IF KESGEVER > 5000000 COMPUTE KESGEVER = (BMAAS + OZHIZ
                    - OZ-IN - KESEP1 - MATRAH) * 0.3
            ELSE COMPUTE KESGEVER =  (BMAAS + OZHIZ - OZ-IN
                                - KESEP1 - MATRAH) * 0.25.
    IF SAKAT-DERECE = 1 MOVE 0 TO KESGEVER.
    COMPUTE TOP-GEL-VERGI = TOP-GEL-VERGI + KESGEVER.
    REWRITE BORD-KAYIT.
    COMPUTE VERILEN = BMAAS + YOD + OZHIZ + YAK-YAR + COYAR +
                AILEYAR + KESEP + KON-ED + TAS-TES-PAR.
    COMPUTE KESILEN = KESGEVER + DAMVER + TASKES + E-BORCU +
            ICRA + IKRAZ + KEFALET + KON-ED + KESEP + KESEP1.
    COMPUTE NET-MAAS = VERILEN - KESILEN.
    DISPLAY BITIS NMAAS-1.
DONGU.
    ACCEPT (24 68) SEC WITH AUTO-SKIP.
    ACCEPT FTUS FROM ESCAPE KEY.
    IF F1  PERFORM SIL CLOSE URAL3 GO MENU.
    IF F2  PERFORM SIFIRLA3 GO UC1.
    IF F3 GO PRINTER-YAZ.
    MOVE " " TO SEC.
    GO DONGU.
PRINTER-YAZ.
    PERFORM DEG-ATA.
    OPEN OUTPUT URAL4.
    WRITE SATIR FROM ESR1. WRITE SATIR FROM BOS.
    WRITE SATIR FROM ESR2. WRITE SATIR FROM ESR3.
    WRITE SATIR FROM ESR4. WRITE SATIR FROM ESR5.
    WRITE SATIR FROM ESR6. WRITE SATIR FROM ESR7.
    WRITE SATIR FROM ESR8. WRITE SATIR FROM ESR9.
```

```
    WRITE SATIR FROM ESR10. WRITE SATIR FROM BOS.
    WRITE SATIR FROM ESR11. WRITE SATIR FROM BOS.
    WRITE SATIR FROM ESR12.
    CLOSE URAL4.
    GO DONGU.
AA1.
    ADD 1 TO K.
    COMPUTE D = 21 - K.
    PERFORM BB1 VARYING LIN FROM K BY 1 UNTIL LIN > D.
BB1.
    COMPUTE COL = I + 5.
    DISPLAY (LIN , COL) "|".
    COMPUTE COL = 78 - I.
    DISPLAY (LIN , COL) "|".
SAY.
    PERFORM SAYI VARYING I FROM 1 BY 1 UNTIL I > 600.
SAYI.
OZ-CERD.
    DISPLAY (LIN , 1 ) "|"
            (LIN , 79) "|".
SIL.
    PERFORM SIL1 VARYING LIN FROM 1 BY 1 UNTIL LIN > 22.
SIL1.
    DISPLAY (LIN , 1) BOS.
SIFIRLA.
    MOVE ZEROS TO IS-YER-KOD BOL-KOD SIC-NO BIL-GIR-TAR TEL
        IS-GIR-TAR IS-CIK-TAR YAN-OD-PUANI E-DER-KADROSU
        E-GOSTERGE G-DER-KADROSU G-GOSTERGE EK-GOSTERGE
        COCUK-SAYI SAKAT-DERECE TOP-GEL-VERGI ICRA IKRAZ
        KEFALET NUMARA KG-GOST E-BORCU.
    MOVE SPACES TO AD-SOYAD SEMT SOKAK DOG-YER BABA-ADI IL
        ILCE KOY-MAH HANE CILT IS-CIK-NEDEN SENDIKA GOREV.
        MOVE SPACE TO MED-HALI.
SIFIRLA1.
    MOVE ZEROS TO MAAS-KAT Y-OD-KAT O-HIZ-YUZ MUST-MAAS YAK-YAR
         KON-ED OZ-IN.
```

```
YENIDOSYA.
    OPEN OUTPUT URAL3.
    MOVE MAAS-KAT TO MAAS-KAT1.
    WRITE MAAS-KAT1. CLOSE URAL3.
SIFIRLA3.
    MOVE ZEROS TO BMAAS YOD OZHIZ COYAR AILEYAR KESEP
         TASKES MATRAH KESGEVER VERILEN KESILEN NET-MAAS
         DAMVER TAS-TES-PAR.
DEG-ATA.
    MOVE SIC-NO TO SNO1. MOVE AD-SOYAD TO ADS.
    MOVE BMAAS TO BUCR. MOVE YOD TO YOD1.
    MOVE DAMVER TO DMGV. MOVE OZHIZ TO OZHZ.
    MOVE E-BORCU TO EBRC. MOVE YAK-YAR TO YKYR.
    MOVE ICRA TO ICR1. MOVE AILEYAR TO AYAR.
    MOVE IKRAZ TO IKRZ. MOVE COYAR TO COYR.
    MOVE KEFALET TO KFLT. MOVE KESEP TO KSP2.
    MOVE KON-ED TO KNE2. MOVE KON-ED TO KNE1.
    MOVE TASKES TO TKS1. MOVE TAS-TES-PAR TO TTP1.
    MOVE OZ-IN TO OZN1. MOVE KESILEN TO KSL1.
    MOVE VERILEN TO VRL1. MOVE NET-MAAS TO NMAAS.
    MOVE KESGEVER TO KGVE.
```

IV.13.5. RELATIVE DOSYALAR

Cobol programlama dilindeki RELATIVE dosya organizasyonu diğer programlama dillerindeki RANDOM dosya organizasyonuna benzer.

RELATIVE dosya içerisinde yer alan kayıtlar 1-32767 'e kadar olan değişken numaralarına sahiptir. RELATIVE dosyalarda her türlü giriş çıkış işlemi Kayıt No Değişkeni olarak adlandırılacak bu değişken ile yapılır.

IV.13.5.1. RELATIVE Dosyalara Erişim Şekilleri

RELATIVE dosyalara erişim şekilleri SEQUENTIAL , RANDOM ve DYNAMIC olmak üzere üç türlü olabilir.

SEQUENTIAL erişim söz konusu ise ; dosyanın kayıt alanlarına , Kayıt No Değişkeninin artan değerleri ile ulaşılabilir.

RANDOM erişim sözkonusu ise ; dosyanın kayıt alanlarına erişim rastgele olacaktır. Kayıt No Değişkenine dosyadan

okunmak istenen kayıdın Kayıt No Değişken bilgisi atanarak okuyucu kafa istenilen kayıt üzerine konumlandırılabilir. DYNAMIC erişim sözkonusu ise ; dosya içerisinde bulunan kayıtlara ulaşım programcının isteğine bağlıdır. DYNAMIC erişimde erişim şekli SEQUENTIAL veya RANDOM olabilir.

IV.13.5.2. RELATIVE TİPİ DOSYALARIN DOSYA AÇILIM MODLARI

1. INPUT modu :

Sadece dosyadan bilgi okumak için kullanılır ve yanlızca okuyucu kafa çalışır. Dosyanın INPUT modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.

2. OUTPUT modu :

Sadece dosyaya bilgi yazmak için kullanılır ve yanlızca yazıcı kafa çalışır. Dosya diskette ilk defa açıldığı zaman bu modda açılmalıdır. OUTPUT modunda açılmış olan bir dosyada daha önceden bulunan bütün kayıtlar silinir. Bundan dolayı dosyanın OUTPUT modunda açılırken çok dikkatli olunması gerekir. Örnek olarak içerisinde 17 kayıt bulunan bir dosya OUTPUT modda açılacak olur ise daha önceden yapılmış olan 17 kayıt silinir. Çünkü bir dosya OUTPUT modda açılıyor ise yazıcı kafa ilk kayıdın üzerine konumlanır böylece daha önce yapılmış bulunan tüm kayıtlar yok olur.

3. I-O (INPUT-OUTPUT) modu :

Dosyadan bilgi okuma ve dosyaya bilgi yazmak için kullanılır. Okuyucu ve yazıcı kafa beraber çalışır. Dosyanın bu modda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.

IV.13.5.3. RELATIVE Dosyaların ENVIRONMENT DIVISION Bölümünde Tanımlanması Gereken Kısımları

ENVIRONMENT DIVISION bölümünün INPUT-OUTPUT SECTION kısmında RELATIVE dosyalar ile ilgili tanımlamalar aşağıdaki gibi olabilir.

```
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT ‹Dosya Adı› ASSIGN TO DISK / TAPE
    ORGANIZATION IS RELATIVE
```

ACCESS MODE IS SEQUENTIAL/RANDOM/DYNAMIC
RELATIVE KEY IS ‹Kayıt No Değişkeni›
FILE STATUS IS ‹Hata değişkeni›.

SELECT deyimini takiben verilen isim dosya adıdır ve başka bir yerde tanımlanmaz. Kullanılacak her dosya için mutlaka bir SELECT deyimi kullanılacaktır.

DATA DIVISION bölümünde burada tanımlanmış dosyanın disketteki gerçek adı aktarılacaktır.

Fakat dosyayı açma , okuma ve kapama gibi işlemler burada belirtilen dosya adına göre yapılacaktır. (SELECT ile verilen dosyanın adı)

ASSIGN deyimi ile dosyanın hangi amaçla açılabileceğini verebiliriz.

Eğer ASSIGN TO DISK kullanılmış ise dosya ile ilgili işlemler disk veya disket üzerinde yapılır.

ASSIGN TO TAPE kullanılmış ise dosya ile ilgili işlemler tape üzerinde yapılır.

ORGANIZATION IS RELATIVE kelimesi ile açılacak dosyanın düzenleme şeklinin RELATIVE dosya olduğu belirtilmektedir.

RELATIVE olarak açılmış bir dosyaya SEQUENTIAL (SIRALI) erişim yapılmak isteniyor ise ;

ACCESS MODE IS SEQUENTIAL

cümlesi yazılır.

RELATIVE olarak açılmış bir dosyaya RANDOM (RASTGELE) erişim yapılmak isteniyor ise ;

ACCESS MODE IS RANDOM

cümlesi yazılır.

RELATIVE olarak açılmış bir dosyaya DYNAMIC (Hem SIRALI hemde RASTGELE) erişim yapılmak isteniyor ise ;

ACCESS MODE IS DYNAMIC

cümlesi yazılır.

RELATIVE KEY IS ‹Kayıt No Değişkeni› cümlesi ile belirtilen Kayıt No Değişkeni dosyanın kayıt değişkenlerinden biri değildir. Kayıt No Değişkeni WORKING-STORAGE SECTION bölümünde tanımlanmış nümerik tipte bir değişkendir. Dosya üzerinde yapılacak olan tüm giriş çıkış işlemleri (yazma , okuma , değiştirme , silme , listeleme gibi) Kayıt No

Değişkeninden yararlanılarak yapılır.
Program içerisinde dosya ile ilgili meydana gelebilecek tüm hataların kod numaraları , FILE-STATUS IS ‹Hata değişkeni› ile belirtilen hata değişkenine yüklenir. Bu hata değişkeni WORKING-STORAGE SECTION bölümünde tanımlanmış olup nümerik veya alfa nümerik 2 karakterli bir değişkendir. Bu değişkene yüklenebilecek hataların bazıları şunlardır ; 00 , 10 , 20 , 30 , 95 gibi. Bu hataların anlamı ileriki konularda verilecektir.

IV.13.5.4. RELATIVE Dosyaların DATA DIVISION Bölümünde Tanımlanması Gereken Kısımları

DATA DIVISION bölümünde RELATIVE tip dosyalar için aşağıdaki tanımlamalar yapılır.

```
DATA DIVISION.
FILE SECTION.
FD ‹Dosya Adı› LABEL RECORD IS STANDARD
    VALUE OF FILE-ID IS ‹Dosyanın disketteki adı›
    DATA RECORD IS ‹Kayıt Değişkeni›.
01 ‹Kayıt Değişkeni›.
   02 ‹Kayıt bilgilerinin tanımı›.
```

FD seviyesi ile belirtilen dosya adı ENVIRONMENT DIVISION bölümünde SELECT ile verilmiş dosya adı ile aynı olmalıdır. ENVIRONMENT DIVISION bölümünde SELECT ile verilmiş her dosya adı için DATA DIVISION bölümünde bir FD seviyesi bulunmalıdır. RELATIVE dosyalar sadece disk , disket veya tape üzerinde açılırlar. Bundan dolayı LABEL RECORD IS STANDARD tanımlaması yapılmalıdır.
VALUE OF FILE-ID ile verilen ‹Dosyanın disketteki adı› ile dosyanın disketteki adı belirtilir. Bu WORKING-STORAGE SECTION bölümünde tanımlanmış olan karakter tipindeki bir değişken de olabilir.
DATA RECORD IS ‹Kayıt Değişkeni› ile verilen kayıt adı DATA DIVISION bölümünün FILE SECTION kısmında 01 seviyesinde tanımlanmış olan bir değişken ismidir. Eğer kayıt değişkeni FD seviyesinin hemen altında belirtilir ise DATA RECORD IS ‹Kayıt Değişkeni› kelimesinin yazılmasına gerek yoktur.
01 seviyesinde tanımlanan ‹Kayıt Değişkeni› ile dosyaya

yazılacak bilgiler tanımlanmalıdır. Dosyaya kayıt yapma işlemleri burada belirtilen kayıt değişkeni adı altında yapılacaktır.

IV.13.5.5. RELATIVE Dosyaların PROCEDURE DIVISION Bölümünde Tanımlanması Gereken Kısımları

Bu bölümde RELATIVE tipli bir dosyanın açılması için OPEN deyimi kullanılır. Eğer DATA DIVISION bölümünde dosyanın disk veya disketteki adı WORKING-STORAGE SECTION bölümünde tanımlanmış bir karakter tipindeki değişken ise PROCEDURE DIVISION bölümünde , OPEN deyimi kullanılmadan önce bu değişkene dosyanın disk veya disketteki gerçek adı aktarılır.

IV.13.5.5.1. OPEN DEYİMİ

PROCEDURE DIVISION bölümünde dosya açmak için kullanılır.

Genel Yazılım Formatı :

OPEN ‹Dosya Açılım Modu› ‹Dosya Adı›.

Yukarıda belirtilen ‹Dosya Açılım Modu› ile dosyanın hangi amaçla açılacağı belirtilmektedir. RELATIVE dosyalarda dosya açma şekilleri OUTPUT , INPUT , I-O modlarında olabilir. ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünün FILE-CONTROL kısmında SELECT ile belirtilmiş olan dosya adı belirtilmelidir.

1. OUTPUT modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN OUTPUT ‹Dosya Adı›.

OUTPUT modu sadece dosyaya bilgi yazmak için kullanılır ve yanlızca yazıcı kafa çalışır. Dosya diskette ilk defa açıldığı zaman bu modda açılmalıdır. OUTPUT modunda açılmış olan bir dosyada daha önceden bulunan bütün kayıtlar silinir. Bundan dolayı dosyanın OUTPUT modunda açılırken çok dikkatli olunması gerekir. Örnek olarak içerisinde 17 kayıt bulunan bir dosya OUTPUT modda açılacak olur ise daha önceden yapılmış olan 17 kayıt silinir. Çünkü bir dosya OUTPUT modda açılıyor ise yazıcı kafa ilk kayıdın üzerine konumlanır böylece daha önce yapılmış bulunan tüm kayıtlar yok olur. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. OUTPUT modda açılmış olan RELATIVE

dosyalarda sadece WRITE deyimi çalışır.

2. INPUT modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN INPUT ‹Dosya Adı›.

INPUT modu sadece dosyadan bilgi okumak amacı için kullanılır ve yanlızca okuyucu kafa çalışır. Dosyanın INPUT modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir.Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. INPUT modunda açılmış RELATIVE dosyalarda sadece READ deyimi çalışır. Eğer dosya OUTPUT modda açılmadan INPUT modda açılmaya çalışılır ise FILE STATUS ile verilen hata değişkenine 30 değeri yüklenir.

3. I-O modunda dosya açılımı :

Genel Yazılım Formatı :

OPEN I-O ‹Dosya Adı›.

I-O (INPUT-OUTPUT) modu dosyadan bilgi okuma ve dosyaya bilgi yazmak için kullanılır. Okuyucu ve yazıcı kafa beraber hareket eder. I-O modu ile açılan RELATIVE tipli dosyalarda her türlü giriş ve çıkış işlemi (kayıt yapma , kayıt ilave , kayıt silme , kayıt düzeltme , listeleme) yapılabilir. Dosyanın I-O modunda açılabilmesi için daha önceden OUTPUT modunda açılmış olması gerekir. Eğer OUTPUT modda açılmamış bir dosya I-O modunda açılmaya kalkılır ise Hata değişkenine 30 değeri yüklenir. Burada belirtilen ‹Dosya Adı› ile ENVIRONMENT DIVISION bölümünde SELECT ile belirtilmiş olan dosya adının aynı olması gerekir. I-O modunda açılmış olan RELATIVE dosyalarda sadece WRITE , READ , START , DELETE ve REWRITE deyimleri çalışır. Bundan dolayı RELATIVE dosyalarda dosya açılım şekli olarak I-O modu tercih edilir.

IV.13.5.5.2. WRITE DEYIMI

WRITE deyimi DATA DIVISION bölümünün FILE SECTION kısmında 01 seviyesinde belirtilmiş olan (DATA RECORD IS ile verilmiş olan Kayıt Değişkeni) Kayıt Değişkeni 'nin içeriklerini ilgili dosyaya yazılmasını sağlar. RELATIVE

dosyalarda WRITE deyiminin çalıştırılabilmesi için dosyanın OUTPUT veya I-O modunda açılmış olması gerekmektedir. WRITE deyimi kullanılmadan önce , yapılacak olan kaydın kayıt nosu Kayıt No Değişkenine aktarılmalıdır.

Genel Yazılım Formatı :

WRITE ‹Kayıt Değişkeni› FROM ‹Değişken› INVALID KEY ‹İfade›

WRITE deyiminin en basit şekli WRITE ‹Kayıt Değişkeni› dir. Bu şekilde kullanıldığında o anda ‹Kayıt Değişkeni›'nde bulunan bilgiler dosyaya yazılır.

Eğer FROM kullanılacak ise FROM 'dan sonra beliritilen ‹Değişken› 'e ait bilgiler ‹Kayıt Değişkeni› aracılığı ile dosyaya yazılırlar. Ancak bu ‹Değişken› içerik , uzunluk ve tip olarak ‹Kayıt Değişkeni› ne uygun olması gerekir.

Eğer dosyada daha önceden var olan bir kayıt dosyaya WRITE deyimi ile tekrar yazılmaya kalkışılır ise bir hata meydana gelir. WRITE deyiminin icrası sırasında böyle bir hata meydana gelir ise ;

Eğer INVALID KEY ‹İfade› kelimesi kullanıl dı ise programın akışı INVALID KEY ile verilen ifadeye yönlenecektir.

IV.13.5.5.3. READ DEYİMİ

Diskette mevcut olan bir dosyanın kayıtlarını okumak ve kayıt değişkenine aktarmak için kullanılır. READ deyimi çalıştırılmadan önce Kayıt No Değişkenine okunacak olan kaydın kayıt nosu aktarılmalıdır.

Genel Yazılım Formatı :

READ ‹Dosya Adı› NEXT RECORD INTO ‹Değişken› AT END ‹Komut›.

veya

READ‹Dosya Adı›NEXT RECORD INTO‹Değişken›INVALID KEY‹İfade›.

Belirtilen formda yer alan RECORD kelimesi cümle bütünlüğü içindir. Yazılmasının yada yazılmamasının bir önemi yoktur.

READ deyiminin çalıştırılabilmesi için dosyanın INPUT veya I-O modunda açılmış olması gerekir.

READ deyiminin en basit şekli READ ‹Dosya Adı› dır. Bu şekilde kullanılan bir READ deyimi , okuyucu kafanın bulunduğu üzerinde bulunulan kayıdı okuyarak , DATA DIVISION bölümünün FILE SECTION kesiminde 01 seviyesinde tanımlanmış olan ‹Kayıt Değişkeni›'ne aktarır. Burada

belirtilen ‹Dosya Adı› , ENVIRONMENT DIVISION bölümünün FILE-CONTROL kısmında SELECT ile verilmiş dosya adı ile aynı olması gerekmektedir.

Okutulan kayıt okuma işleminden sonra ‹Kayıt Değişkeni› ninden başka bir değişkene aktarılacak ise ;

READ ‹Dosya Adı› NEXT RECORD INTO ‹Değişken›

kullanılır. Bu yazılıma göre dosyadan okutulan kayıt INTO ile verilen ‹Değişken›'e aktarılır. Ancak bu değişkenin ‹Kayıt Değişkeni› ile içerik , uzunluk ve tip olarak aynı olması gerekmektedir.

RELATIVE tipli bir dosyaya erişim şekli SEQUENTIAL (Sıralı) ise READ deyimi ile NEXT kelimesinin birlikte kullanılması gerekir. Eğer NEXT kullanılmamış ise okuma iş rastgele yapılacaktır.

RELATIVE dosyalarda AT END ve INTO deyimi aynı SEQUENTIAL dosyalarda kullanıldığı gibidir.

INVALID KEY ‹İfade› ile dosyada olmayan bir kayıt okunmak istenildiği zaman meydana gelebilecek hatalar ile mesaj verilebilir.

IV.13.5.5.4. START DEYİMİ

RELATIVE dosyalarda istenilen kayıt üzerine konumlanabilmek için kullanılan bir deyimdir. Hacmi oldukça büyük dosyalarda , genellikle listeleme bölümlerinde bir takım zorluklar ile karşılaşılabilir. Örneğin dosyanın tamamı değilde belirli bir kısmının listelenmesi istenilebilir. Bu işin gerçekleştirilebilmesi için listelenmesi istenilen aralığın ilk kaydına konumlanma sağlanmalıdır. Bu konumlama START deyimi ile yapılabilir.

Genel Yazılım Formatı :

```
START ‹Dosya Adı› KEY IS EQUAL TO          ‹Kayıt No Değişkeni›
                         NOT LESS THAN     ‹Kayıt No Değişkeni›
                         GREATER THAN      ‹Kayıt No Değişkeni›
                         INVALID KEY       ‹İfade›.
```

START deyimi kullanılmadan önce okunmak istenen kayıtların başlangıç değeri Kayıt No Değişkenine aktarılır. START deyimi ile dosyanın istenilen kayıdı üzerine konumlama yapılabilir. KEY IS cümlesinde kullanılan ifadelere göre

ortaya çıkacak etkiler şunlardır.
EQUAL TO kullanıldı ise ; Kayıt No Değişkenine ait değer "KEY" dosyasından aranır. Aranılan kayıt bulunur ise okuyucu kafa bu kayıt üzerine konumlanır. Eğer aranılan kayıt bulunamadı ise hata değişkenine 23 değeri yüklenir veya programın akışı INVALID KEY ile verilen ifadeye kayar.
GREATER THAN kullanıldı ise ; "KEY" dosyasındaki değeri Kayıt No Değişkeninden büyük olan ilk kayıt üzerine konumlama yapılır. Eğer "KEY" dosyasındaki en son bilginin değeri , Kayıt No Değişkenine eşit veya daha küçük ise istenilen konumlanma gerçekleşmez ve hata değişkenine 23 değeri yüklenir yada programın akışı INVALID KEY ile verilen ifadeye kayar.
NOT LESS THAN kullanıldı ise ; "KEY" dosyası içerisinde Kayıt No Değişkenine eşit olan değer aranır , böyle bir kayıt yok ise değeri Kayıt No Değişkeninden büyük olan kayıt üzerine konumlama yapılır. Kayıt No Değişkeni ile temsil edilen değer "KEY" dosyasında yok ise hata değişkenine 23 değeri yüklenir veya programın akışı INVALID KEY ile verilen ifadeye kayar.
START deyimi ile istenilen kayıt üzerine konumlama yapıldıktan sonra READ deyimi çalıştırılarak okunmak istenen kayıt veya kayıtlar okunur.

IV.13.5.5.5. REWRITE DEYİMİ

REWRITE deyimi , yazıcı kafayı Kayıt No Değişkenin bulunduğu kayıt üzerine konumlandırır ve kayıt değişkeni tarafından temsil edilen bilgileri bu pozisyondaki mevcut kaydın üzerine yazar.
Genel Yazılım Formatı :
REWRITE ‹Kayıt Değişkeni›FROM ‹Değişken›INVALID KEY ‹İfade›.
REWRITE deyimi ile ‹Kayıt Değişkeni› ile ifade edilen bilgiler dosyaya yeniden yazılır. REWRITE deyimi ile dosya içerisinde olan bir bilgi üzerinde gerekli değişiklikler yapıldıktan sonra tekrar aynı yerine yazılır. REWRITE deyimi ile dosyaya yeni kayıt yapılmaz. REWRITE deyiminin çalışabilmesi için dosyanın OUTPUT veya I-O modunda açılmış olması gerekir.

RELATIVE dosyalarda erişim şekli SEQUENTIAL ise REWRITE deyimi kullanılmadan önce READ deyiminin başarılı bir şekilde çalıştırılmış olması gerekir.
RANDOM ve DYNAMIC erişimin söz konusu olduğu durumlarda REWRITE deyimi ile değiştirilecek olan kayıdın READ deyimi ile önceden okutulmuş olmasına gerek yoktur. REWRITE deyiminin çalışması için değiştirilecek kayda ait kayıt no değerinin Kayıt No Değişkenine aktarılmış olması şarttır.
REWRITE deyiminin çalıştırılması esnasında herhangi bir hata meydana gelecek olur ise programın akışı INVALID KEY ile verilen ifadeye kayar.

IV.13.5.5.6. DELETE DEYİMİ

RELATIVE dosyalarda istenilen bir kayıt silinmek istendiği zaman kullanılan bir deyimdir. Silinecek kayıdın kayıt no değeri Kayıt No Değişkenine aktarılarak silinme işlemi gerçekleştirilir.
Genel Yazılım Formatı :

DELETE ‹Dosya Adı› RECORD INVALID KEY ‹İfade›.

RELATIVE dosyaya ulaşım şekli SEQUENTIAL ise silinecek kayıt DELETE deyimi kullanılmadan önce mutlaka READ deyimi ile okutulmuş olmalıdır. Dosyaya erişim şeklinin RANDOM veya DYNAMIC olması durumunda silinecek kaydın önceden READ deyimi ile okutulmuş olmasına gerek yoktur.
DELETE deyiminin kullanılması esnasında herhangi bir hata meydana geldiğinde programın akışı INVALID KEY ile verilen ifadeye kayar.

SONUÇ

Bu çalışmamda temel amacım işletim sistemlerini en ince noktasına kadar açıklamak ve bu işletim sistemleri içerisinde DOS işletim sistemini kullanan yüksek seviyeli dillerden BASIC ve COBOL incelemektir.

Tezin I. bölümünde DOS işletim sistemi incelenmiş ve komutlar örneklerle açıklanmıştır.

II. bölümde çok kullanıcılı sistem XENIX incelenmiş kullanılışı ve komutları örnekler ile açıklanmıştır.

III. bölümde DOS işletim sistemine dayalı yüksek seviyeli dillerden -Mühendislik Uygulamalarına Yönelik- BASIC programlama dili kullanılışı , komutları ve dosya yapıları en ince ayrıntısına kadar açıklanmıştır.

IV. bölümde DOS işletim sistemine dayalı yüksek seviyeli dillerden -Ticari Uygulamaya Yönelik- COBOL programlama dili komutları ve dosya yapıları en ince noktasına kadar açıklanmıştır.

Çok yoğun bir çalışmanın ürünü olan bu tez yukarıda belirtilen bölümler şeklinde hazırlanmıştır.

Bu çalışma sonucunda bilgisayar kullanımı , işletim sistemleri , BASIC ve COBOL dilleri ve etkin kullanımları çalışmamda incelenmistir.

ÖZGEÇMİŞİM ;

1965 - ERZİNCAN doğumluyum. İlköğrenimimi Erzincan İnönü İlkokulu'nda orta okul birinci ve ikinci sınıfı Erzincan Cumhuriyet Ortaokulu'nda tamamladıktan sonra ortaokul üçüncü sınıf ve lise öğrenimimi İstanbul Suadiye Lisesinde tamamladım. 1982 yılında Yıldız Üniversitesi Mühendislik Fakültesi Matematik Mühendisliği Bölümüne girdim ve buradan 1986 yılında mezun oldum. 1987 yılında Yıldız Üniversitesi Fen Bilimleri Enstitüsü Araştırma Görevliliği kadrosuna atanarak Matematik Mühendisliği Bölümünde Araştırma Görevliliği yaptım. Aynı yıl Yıldız Üniversitesi Fen Bilimleri Enstitüsü Matematik Mühendisliği ana bilim dalında mastır yapmaya hak kazandım. 1989 yılında Matematik Mühendisliği Sistem Analizi ana bilim dalında Araştırma Görevliliğine atandım ve halen bu görevime devam etmekteyim.

İbrahim EMİROĞLU