C Programlama Dili'ne Giriş

Ders 16: *Dosya Yönetimi*

```
##################-- (%90)
```

En son güncelleme: Wed, 30 Nov 2011 13:22:02 +0200

Giriş

Birçok programda, bazı verilerin disk üzerinde saklanmasına gerek duyulur. Bütün programlama dillerinde, sabit disk sürücüsü (Hard Disk Drive, HDD) üzerindeki verileri okumak veya diske veri yazmak için hazır fonksiyonlar tanımlanmıştır. C programlama dilinde, disk dosyasına erişim iki yöntemle yapılır. Bunlar üst düzey ve alt düzey olarak adlandırılır. Üst düzey G/Ç yöntemi ANSI C tarafından desteklenmektedir. Bu kısımda Üst düzey G/Ç konu edilecektir[1].

16.1 Dosya Açma ve Kapama

Bir dosyaya okuma/yazma yapmak için onun açılması gerekir. Dosya açmak için `fopen()`, kapatmak için `fclose()` fonksiyonu kullanılır. Bu fonksiyonlar `stdio.h` başlık dosyasında tanımlanmıştır.

Genel olarak, dosya açma kapama adımları şu şekildedir:

```
FILE *dosya; /* dosya göstericisi */

dosya = fopen(const char dosya_adı, const char mod);

...
dosya işlemleri
...

fclose(dosya);
```

Burada `FILE`, `stdio.h` içerisinde bildirilmiş bir yapıdır. *`mod`* ile açılacak olan dosyanın ne amaçla açılacağı belirlenir[2].

Tablo 16.1: *Dosya açma modları*

Açılış Modu	İşlem Türü
`r`	Salt okunur (read only). Dosyanın açılabilmesi için önceden oluştrulmuş olması gerekir. Bu modda açılmş olan bir dosyaya yazma yapılamaz.
`w`	Yalnızca yazma (write only). Dosya diskte kayıtlı olsun veya olamsın dosya yeniden oluşturulur. Bu modda açılmış olan bir dosyadan okuma yapılamaz.
`a`	Ekleme (append). Kayıtlı bir dosyanın sonuna veri eklemek için açılır. Bu modda açılmış olan bir dosyadan okuma yapılamaz.
`r+`	Okuma ve yazma. Bu modda açılmış olan bir dosyanın daha önce varolması gerekir.
`w+`	Okuma ve yazma. Bu modda açılmış olan bir dosya var olsun veya olmasın dosya yeniden oluşturulur.
`a+`	Okuma ve yazma. Kayıtlı bir dosyanın sonuna veri eklemek için açılır.

`deneme.dat` adlı bir dosyanın, yazmak için açılıp açılmadığını test etmek için aşağıdaki kod kullanılır:

```
FILE *yaz;  /* dosya göstericisi */

yaz = fopen("deneme.dat", "w");

if( yaz == NULL ){
      puts("dosya acilmiyor...");
      exit(1);
}

...
/* açılırsa! dosya işlemleri */
...

fclose(yaz);
```

NOT

`deneme.dat` dosyası ile ana program aynı dizin içinde olmalıdır. Aksi halde, dosyanın tam yolu bildirilmelidir.

Örneğin dosyanın yolu: `C:\WINDOWS\DESKTOP\deneme.dat` ise dosya açılırken

```
yaz = fopen("C:\\WINDOWS\\DESKTOP\\deneme.dat", "w");
```

şeklinde açık yol bildirilmelidir. Aynı işlem Linux ortamında da geçerlidir.

```
yaz = fopen("/home/bingul/DATA/deneme.dat","w");
```

gibi.

16.2 Metin ve İkili Dosyalar

İşletim sistemlerinde genelde iki çeşit dosya kullanımına rastlanmaktadır. Bunlar metin (text) ve ikili (binary) dosyalar olmak üzere ikiye ayrılır. Dosyanın hangi türden olduğu açılırken `fopen()` fonksiyonu ile belirtilebilir.

Açılış modunda metin dosyaları için **`t`**, ikili dosyalar için **`b`** eklenir. Örneğin **`"r+t"`** şeklinde bir açılış modu var olan bir dosyanın okuma yazma ve metin olarak açılacağı anlamına gelir. Benzer olarak **`"wb"`** açılış modu dosyanın ikili modda oluşturulacağını gösterir. Fakat bu belirleme yapılmamışsa, varsayılan açılış modu metindir (yani `t`).

16.3 Dosya Fonksiyonları

Bu kısımda, Tablo 16.2'de verilen dosyalama foksiyonlarının bazılarının kullanımı, örnek programlar üzerinde anlatılmıştır.

Tablo 16.2: *Üst düzey dosyalama fonksiyonları*

Fonksiyon	Görevi
`fopen()`	Dosya oluşturur, açar
`fclose()`	Dosyayı kapatır
`putc()`	Dosyaya bir karakter yazar
`getc()`	Dosyadan bir karakter okur
`feof()`	Dosya sonuna gelindiğini sorgular
`fprintf()`	Dosyaya formatlı veri yazar
`fscanf()`	Dosyadan formatlı veri okur
`fputs()`	Dosyaya katar yazar
`fgets()`	Dosyadan katar okur
`fwrite()`	Dosyaya dizi yazar
`fread()`	Dosyadan dizi okur

Program 16.1: *Bir dosyaya veri yazma*

```
01: /* 16prg01.c:
02:    10 öğrenciye ait bilgileri 'ogrenci.txt' dosyasına kaydeder. */
03:
04: #include <stdio.h>
05: #include <stdlib.h>
06:
07: int main()
08: {
09:     FILE *dg;           /* dosya göstericisi */
10:     const int n = 10;   /* öğrenci sayısı    */
11:     char  ad[10];
12:     int   no, Not, i=0;
13:
14:     dg = fopen("ogrenci.txt", "w");
15:
16:     if(  dg == NULL )
17:       puts("ogrenci.txt dosyasi acilmadi. !\n"), exit(1);
18:
19:     puts("10 ogrenciye ait bilgileri girin:");
20:
21:     while( i++<n )
22:     {
23:       printf("%d. ogrencinin numarasi: ",i); scanf("%d",&no);
24:       printf("%d. ogrencinin adi     : ",i); scanf("%s",ad);
25:       printf("%d. ogrencinin notu    : ",i); scanf("%d",&Not);
26:       printf("\n");
27:
28:       fprintf(dg,"%5d %10s %3d\n",no,ad,Not); /* verileri formatlı yaz! */
29:     }
30:
31:     /* dosyayı kapat */
32:     fclose(dg);
33:
34:     puts("Bilgiler kaydedildi.\a");
35:
36:  return 0;
37: }
```

ÇIKTI

```
1. ogrencinin numarasi: 2248
1. ogrencinin adi     : Ahmet
1. ogrencinin notu    : 45

2. ogrencinin numarasi: 1823
2. ogrencinin adi     : Meltem
2. ogrencinin notu    : 88

...

9. ogrencinin numarasi: 6548
9. ogrencinin adi     : Cemil
9. ogrencinin notu    : 38

10. ogrencinin numarasi: 4591
10. ogrencinin adi     : Hasan
10. ogrencinin notu    : 77

Bilgiler kaydedildi.
```

Oluşturulan dosyanın içeriğini görmek için:

MS DOS (Turbo C)	Linux

C:\TC> edit ogrenci.txt	$ edit ogrenci.txt
	yada
	$ pico ogrenci.txt

NOT

`ogrenci.txt` dosyası daha önce oluşturulmuşsa Program 16.1 önceki verileri silip yerine yeni verileri yazacaktır. Bu dosyaya yeni bir veri eklemek için `fopen()` fonksiyonunu 'a' modu ile kullanılmalıdır.

Bir dosyadan veri okumanın bir çok şekli vardır. Veri okuma işlemine basit bir örnek Program 16.2 de sunulumuştur. Bu program `ogrenci.txt` dosyasında kayıtlı öğrencilerden, en düşük notu, en yüksek notu ve sınıf ortalamasını (0'lar hariç) hesaplamaktadır.

Program 16.2: *Bir dosyadan veri okuma*

```
01: /* 16prg02.c:
02:    ogrenci.txt dosyasından no, isim ve not bilgileri okur ve
03:    notlar arasından en büyüğü, en küçüğü ve sınıf ortlamasını (0'lar hariç) hesaplar. */
04:
05: #include <stdio.h>
06: #include <stdlib.h>
07:
08: int main()
09: {
10:    FILE *dg;        /* dosya işaretçisi */
11:    char  Ad[10];
12:    int   Not, No, eb, ek, n, top;
13:    float ort;
14:
15:    if( (dg=fopen("ogrenci.txt","r")) == NULL )
16:      puts("Dosya açılmadı !\n"), exit(1);
17:
18:    /* başlangıç değerleri ata */
19:    ek  =  1000;  /* çok büyük */
20:    eb  = -1000;  /* çok küçük */
21:    top = 0;      /* notların toplamı */
22:    n   = 0;      /* notu 0'dan farlı öğrencilerin toplamı */
23:
24:    while( !feof(dg) )                    /* dosyanın sonuna kadar */
25:    {
26:       fscanf(dg,"%d %s %d",&No,Ad,&Not); /* verileri oku! */
27:
28:       if(Not>eb) eb = Not;               /* en büyük ve en küçük bulunuyor... */
29:       if(Not<ek) ek = Not;
30:
31:       if(Not) n++;                       /* Not 0'dan farklı mı? */
32:
33:       top += Not;                        /* Notlarin toplamı */
34:    }
35:
36:    fclose(dg);                           /* dosyayı kapat */
37:
38:    ort = (float) top/n;                  /* ortalama (0 lar hariç!) */
39:
40:    printf("En yuksek not = %2d\n",eb);   /* sonuçlar ekrana ... */
41:    printf("En  dusuk not = %2d\n",ek);
42:    printf("Ortalama      = %4.1f\n",ort);
43:
44:  return 0;
45: }
```

ÇIKTI

```
En yuksek not = 92
En  dusuk not =  0
Ortalama      = 69.2
```

Bir program içinde birden çok dosya açmak mümkündür. Örneğin bir dosyadan okunan veriler farklı bir formatta başka bir dosyaya yazılması istenebilir.Program 16.3 `kelvin.sck` dosyasında bulunan 100 adet kelvin cinsinden sıcaklık bilgilerini derece karşılıklarını `derece.sck` dosyasına yazar. Bu iki sıcaklık arasındaki çevrim kuralı: K = 273 + C şeklindedir.

Program 16.3: *Bir dosyada saklı verileri farklı bir biçimde başka bir dosyaya yazma*

```
01: /* 16prg03.c
02:    'kelvin.sck' dosyasında bulunan 100 adet kelvin cinsinden sıcaklık
03:    bilgilerini derece karşılıklarını 'derece.sck' dosyasına yazar.
04:    Bu iki sıcaklık arasındaki çevrim kuralı: K = 273 + C şeklindedir. */
05:
06: #include <stdio.h>
07: #include <stdlib.h>
08:
09: int main(void)
10: {
11:    /* dosya göstericileri  */
12:    FILE *oku, *yaz;
13:
14:    /* Dosya adları */
15:    char *kaynak_dosya = "kelvin.sck";
16:    char *hedef_dosya = "derece.sck";
17:
18:    float K,D;
19:    int   i=0, n=100;
20:
21:
22:    /* Dosylara erişim mümkün mü ? */
23:    if( (oku=fopen(kaynak_dosya, "r")) == NULL )
24:    {
```

```
25:      printf("%s dosyası acilmadi.\n", kaynak_dosya);
26:      exit(1);
27:    }
28:    if( (yaz=fopen(hedef_dosya, "w")) == NULL )
29:    {
30:      printf("%s acilmadi.\n", hedef_dosya);
31:      exit(2);
32:    }
33:
34:    for(i=0; i<n; i++)
35:    {
36:       fscanf(oku,"%f",&K);          /* 'kelvin.sck'dan verileri oku     */
37:       D = K - 273.0;                /* dönüşüm denklemi */
38:       fprintf(yaz,"%8.2f\n",D);     /* verileri 'derece.sck'ya yaz      */
39:    }
40:
41:    /* Dosyaları kapat */
42:    fclose(oku);
43:    fclose(yaz);
44:
45:    printf("%s > %s\n", kaynak_dosya, hedef_dosya);
46:    puts("cevirme islemi tamamlandi.");
47:
48:  return 0;
49: }
```

ÇIKTI

```
kelvin.sck > derece.sck
cevirme islemi tamamlandi.
```

16.4 Standart Dosyalar

C Programlama Dili'nde bilgisayarın sahip olduğu ekran, klavye ve portlar birer dosya olarak tanımlanmıştır. Bu dosyalara standart dosyalar denir. Program çalışmaya başladığında beş adet standart dosya otomatik olarak açılır. C, `stdio.h` başlık dosyasında tanımlanan bütün bu standart dosyalara birer sembolik isim vermiştir[3]. Bu isimler Tablo 16.3'de listelenmiştir.

Tablo 16.3: *Standart Dosyalar*

Dosya adı	Görevi
stdout	Standart çıkış ortamı (ekran)
stderr	Standart hata çıkış ortamı (ekran)
stdin	Standart giriş ortamı (klavye)
stdprn	Standart LPT (paralel port)
stdaux	Standart COM (seri port)

Bu dosyaların sembolik isimleri birer dosya göstericisidir. Bu sebeple `FILE` yapısal değişkeni ile kullanılabilen dosya fonksiyonları bu dosyalar için de kullanılabilir. Örneğin, ekrana (standart çıkışa) bir yazı bastırmak için:

```
fprintf(stdout,"Merhaba C\n");
```

Bilgilerin yazıcıya gönderilmesi için yine `fprintf` fonksiyonu kullanılır. Örneğin:

```
fprintf(stdprn,"Merhaba C\n");
```

satırı yazıcıya `Merhaba C` iletinini gönderir.

Ayrıca, `LPT1` veya `PRN` ismini dosya ismi olarak kullanıp yazıcıya basım yapmak da mümkündür [4]. Örneğin:

```
FILE *dg;
...
dg = fopen("LPT1","wt");
fprintf(dg,"Merhaba C\n");
...
flcose(dg);
```

NOT

Tablo 16.3 de verilen standart dosyalardan `stdprn` ve `stdaux` Turbo C'de tanımlı iken Standart C'de tanımlı değildir. (bkz: `stdio.h`)

Program 16.4 Turbo C derleyicisinde derlendiğinde hem ekrana hemde yazıcıya birer mesaj yazar.

Program 16.4: *Standart dosyaların kullanımı*

```
01: /* 16prg04.c
02:    Standart dosyaları kullanarak hem ekrana hemde
03:    yazıcıya birer mesaj yazar. (sadece Turbo C) */
04:
05: #include <stdio.h>
06:
07: int main()
08: {
09:     fprintf(stdout,"Bu mesaj *ekrana*   yazilacak ...\n");
10:     fprintf(stdprn,"Bu mesaj *yaziciya* yazilacak ...\n");
11:
12:  return 0;
13: }
```

ÇIKTI

```
Bu mesaj *ekrana*   yazilacak ...
```

NOT

Eğer yazıcı bağlı yada açık değilse, işletim sistemi kullanıcıyı uyaracak ve programın çıktısı şöyle olacaktır:

```
Bu mesaj *ekrana*   yazilacak ...

Yazma hatası yazılan aygıt PRN
Durdur, yeNiden dene, Yoksay, ipTal?d
```

Powered by PHP