7. HAFTA

BLM102

PROGRAMLAMA DİLLERİ II

Yrd. Doç. Dr. Baha ŞEN

baha.sen@karabuk.edu.tr

KBUZEM
Karabük Üniversitesi
Uzaktan Eğitim Araştırma ve Uygulama Merkezi

1.7. Rastgele Erişimli Dosyalar

Random dosyalarda istenilen kayda doğrudan ulaşılır. Ramdom dosyalarda veriler , sabit uzunluktaki bloklar(kayıtlar) şeklinde saklanırlar. Aynı zamanda veriler karakter dizileri şeklinde de saklanırlar. Rastgele erişimli dosylarda, değişkenlerin içeriği bellekte saklanış biçimiyle kütüğe(dosya) yazıldığı için okuma / yazma işlemleri text dosyalara göre daha hızlıdır. Bu nednele veriler dosyada daha az yer kaplar.

Random dosyalarda bilgiler sabit uzunlukta saklandığı için değişiklik anında kayıt bozulması yoktur. Ayrıca gerçel sayılarda hassasiyet kaybı da yaşanmaz.

Dosyaların içindeki herhangi bir veriye ulaşmak için kayıt alanının dosyanın başlangıcına olan uzaklığını hesaplamak gerekir. Bu hesaplamanın yapılabilmesi için, dosyanın kayıt alanlarının uzunluğunu sabit tutmak gerekir. Dosyanın kayıt alanları sabit ise, bu sabit değer ile herhangi bir kaydın numarasının çarpımı, bu kaydın dosya içindeki yerini belirler.

1.7.1. Kayıt Girişi

Dosyalara kayıt girme işlemi, **`fwrite`** fonksiyonu ile gerçekleştirilmektedir. Dosya açıldıktan sonra fseek ile kayıt konumu ayarlandıktan sonra fwrite komutu ile dosyaya yazma işi yapılır.

```
fwrite ( bellek_isaretcisi, veri_boyutu, veri_adedi, dosya_isaretcisi);
```

Bellek İşaretçisi	Yazdırılmak istenen verinin bellekteki adresinin başlangıç değeri
Veri Boyutu	Yazdırılmak istenen veri tipinin bellekteki boyutu
Veri Adedi	Belirtilen tipten dosyaya kaç adet yazılacağı
Dosya İşaretçisi	Verilerin kaydedileceği dosyayı gösteren işaretçi

1.7.2. Kayıt Listeleme

Dosyalardan veri okuma işlemi **`fread`** fonksiyonu ile gerçekleştirilmektedir. Bunun için dosya açıldıktan sonra fseek komutu

ile okunacak verinin başlangıç adresine gidilir ve daha sonra fread fonksiyonu ile okuma yapılır.

```
fwrite ( bellek_isaretcisi, veri_boyutu, veri_adedi, dosya_isaretcisi);
```

Bellek İşaretçisi	Okunacak verinin bellekte tutulacağı adresin başlangıç değeri
Veri Boyutu	Okunacak verinin bellekte kaplayacağı alan
Veri Adedi	Belirtilen tipten kaç tane okunacağı
Dosya İşaretçisi	Verilerin okunacağı dosyayı gösteren işaretçi

1.7.3. Kayıt Arama

Arama işlemi için öncelikle fseek komutu ile dosyanın başına gidilir. Ardından fread fonksiyonu ile veriler okunduktan sonra seçilen anahtar alan ya da alanlara göre karşılaştırma yapılarak arama işlemi gerçekleştirilir.

1.7.4. Kayıt Silme

Kayıt silme işlemi için öncelikle silinecek veri aranılır. Bulunduktan sonra '*' ile aktif olduğu belirtilen kayıt alanı '-' ile tekrar güncellenerek listeleme işleminde görünmeyecek hale getirilmiş olur. Aynı zamanda eski verilerin saklanması da sağlanmış olur.

1.7.4. Kayıt Düzeltme

Kayıt düzeltme işlemi kayıt arama ve silme işlemi gibidir. Farklı olarak güncellenecek alanlar aktifliği gösteren parametre dışında kalan bilgilerdir.

Dosya işlemlerinde kullanılan bazı fonksiyonlar

Fonksiyon		Açıklama
`int fileno(FILE *dosya_degiskeni)`		Dosya değişkeni ile işaret edilen FILE yapısının alanının değerini üretir. Standart dosyalar C derleyicisi tarafından ilk olarak açıldığı zaman dosya numaraları 0 - 4 arasındadır.
		stdin : 0 stdout : 1 stderr : 2 stdaux : 3 stdprn : 4
`void setbuf(FILE *dosya, char*bellek)`		Dosya için tampon belleğin başlangıç adresini bellek değişkeninin değeri olarak C'ye bildirir.
`int setvbuf(FILE *dosya, char*bellek, int mod, uzunluk)`		Tampon belleğin uzunluğunu belirlemek için kullanılır.
`void clearerr(FILE *dosya)`		Dosyalarla ilgili hata kontrolü yapıldıktan sonra, hatalı durum ortadan kalktığında, daha sonraki kontrollerin doğru çalışması için bayrak alt değişkenini tekrar hatasız durumu gösterecek hale getirir.
`int ferror(FILE *dosya)`		Dosyalarla ilgili genel hata kontrolü için kullanılır.
`int fseek(FILE *dosya, long konum, int baslama_sekli)`		Dosyanın, dosya göstericisinin pozisyonunu değiştirmek için kullanılır.
Başlama Şekilleri	`SEEK_SET`	Dosya başlangıcını gösterir.
	`SEEK_CUR`	Dosyanın kayıt pozisyonunu gösterir.
	`SEEK_END`	Dosya sonunu gösterir.
`int fsetpos(FILE *dosya, long *pozisyon)`		Dosya kayıt alanlarına rastgele erişim için kullanılır.
`int fgetpos(FILE *Dosya, long *pozisyon)`		Dosyanın o anki konumunu öğrenmek için kullanılır
`long ftell (FILE *Dosya)`		O anki kayıt pozisyonunun değerini verir.
`int getw(FILE *Dosya)`		Binary tipli bir dosyadan int tipinde değer okumak için kullanılır.
void rewind(FILE *Dosya)		Dosyanın başlangıcına konumlanmak için kullanılır.

Örnek: Dosyaya kayit girisi ve listeleme

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <io.h>
struct ogrenci
{
  char k;
  char num[10];
  char adsoyad[30];
  char tel[16];
} ogrkay;
FILE *dosya;
int i, kaysay, say;
long kayit_yer;
char cev,c;
int ksay(void)
{
  return ((filelength(fileno(dosya))/sizeof(struct ogrenci)));
}
void kayit_ekle(void)
{
  dosya = fopen("ogrenci.dat","r+b");
  if (dosya == NULL)
    dosya = fopen("ogrenci.dat","w+b");
  system("cls");
  fflush(stdin);
  printf("Numara.....:");
  gets(ogrkay.num);
  printf("Ad Soyad ..:");
  gets(ogrkay.adsoyad);
  printf("Tel........:");
  gets(ogrkay.tel);
  printf("Girilen Bilgiler Dogru mu? [E/H]");
  cev=getche();
  if(cev=='E' || cev=='e')
    {
      ogrkay.k='*';
      fseek(dosya,filelength(fileno(dosya)),SEEK_SET);
      fwrite(&ogrkay,sizeof(struct ogrenci),1,dosya);
    }
  fclose(dosya);
}
void baslik(void)
{
  system("cls");
  if(dosya==NULL) printf("Dosya acilamadi");
  printf("%-10s %-30s %-16s \n\n","NUMARA","AD SOYAD", "TELEFON");
  say=0;
}
void listele(void)
{
  dosya = fopen("ogrenci.dat","rb");
  baslik();
  kaysay=ksay();
  fseek(dosya,0,SEEK_SET);
  for(int i=0; i<kaysay; i++)
    {
      fseek(dosya,(i*sizeof(struct ogrenci)),SEEK_SET);
      fread(&ogrkay,sizeof(struct ogrenci),1,dosya);
```

```
      if(ogrkay.k=='*')
        {
          say++;
          printf("%-10s",ogrkay.num);
          printf("%-30s",ogrkay.adsoyad);
          printf("%-16s\n",ogrkay.tel);
          if(say==20)
            {
              printf("Diger sayfaya gecmek icin bir tusa basiniz");
              getch();
              baslik();
            }
        }
    }
  printf("\nToplam Kayit Sayisi= %d",kaysay);
  printf("\n\nListelenecek kayitlar bitti...");
  fclose(dosya);
  getch();
}
int main()
{
  do
    {
      system("cls");
      printf("1-Bilgi girisi \n2-Bilgi Listeleme \n3-Cikis \n\nSecim:");
      if(c=='1') kayit_ekle();
      if(c=='2') listele();
      if(c=='3') exit(0);
      c=getche();
    }
  while(c!='3');
  getch();
  return 0;
}
```