**C’de Dosya İşlemleri(File Operations)**

C’de dosya işlemleri oldukça önemli bir yere sahiptir. Dosya işlemleri yapan fonksiyonlar stdio.h başlık dosyası içerisinde bulunan fonksiyonlar yardımıyla yapılır. Bu işlemler dosyalarla pek çok iş yapabilir. Örneğin dosyayı açma, dosyayı okuma, dosyaya yazma ve dosyayı kapatma. Öncelik bilmemiz gereken ilk şey bir dosya eğer açılmışsa mutlaka kapatılma gerekmektedir. Aksi durumda tanımsız davranış oluşabilir. Bu oldukça tehlikeli bir durumdur. Bu duruma kaynak sızıntısı(resource leak) adı verilir. Bunu dinamik olarak oluşturulmuş bir bellek bloğunun free edilmesine benzetebiliriz.

Biz dosyayı açmak için standart kütüphanenin fopen fonksiyonu kullanılır. Bu fonksiyonun genel biçimi şu şekildedir:

fopen(<dosya ismi>,<dosyanın açılma modu>);

**Anahtar Notlar :** Bir dosyanın açılabilmesi için illa o dosyanın bilgisayarda var olması gerekmez. Eğer bir dosya yok ise bu durumda bu dosya yaratılır ve istenildiği gibi kullanılması mümkündür.

Aşağıdaki örnekte fopen fonksiyonu kullanılarak bir kod yazılmıştır. Daha önceden de bahsettiğimiz gibi bu dosya yok ise fopen fonksiyonuyla yaratılması durumundan bahsedilebilir:

#include <stdio.h>

int main(void)

{

FILE\* file = fopen("asal.txt", "r");

return 0;

}

Burdaki örnekte ise ikinci parametre olarak görüldüğü gibi “r” gönderilmiştir. r burda read anlamına gelmektedir. Yani dosya sadece okuma amaçlı açılıyor anlamına gelmektedir. İşte bazı dosya açma biçimlerinin tablosu:

|  |  |  |
| --- | --- | --- |
| Fonksiyona gönderilecek argüman | İfadenin ismi | İfadenin yaptığı iş |
| "r" | Read | Bu açma modu, bir dosyayı yazma amaçlı açar. Dosya bulunamazsa hata verir. |
| "w" | Write | Bu açma modu, bir dosyayı yazma amaçlı açar. Dosya mevcutsa içeriğini siler ve yeni bir dosya oluşturur. Dosya yoksa yeni bir dosya oluşturulur. |
| "a" | Append | Bu açma modu, bir dosyayı yazma amaçlı açar, ancak dosyanın sonuna ekler. Dosya mevcutsa içeriğini korur ve yeni verileri dosyanın sonuna ekler. Dosya yoksa yeni bir dosya oluşturulur. |
| "rb" | Read Binary | Bu, bir dosyayı ikilik(binary) okuma amaçlı açar. İkili verileri okumak için kullanılır. |
| "wb" | Write Binary | Bu, bir dosyayı ikili yazma amaçlı açar. İkili verileri yazmak için kullanılır. |
| "ab" | Append Binary | Bu, bir dosyayı ikili yazma amaçlı açar, ancak dosyanın sonuna ekler. İkili verileri dosyanın sonuna eklemek için kullanılır. |
| "+r" | Read and Update | Bu mod, bir dosyanın hem okunmasına hem de güncellenmesine izin verir. Dosya mevcutsa okuma ve yazma işlemleri yapabilirsiniz. Dosya mevcut değilse hata verir. |
| "+w" | Write and Update | Bu mod, bir dosyanın hem yazılmasına hem de güncellenmesine izin verir. Dosya mevcutsa içeriğini siler ve yeni bir dosya oluşturur. Dosya mevcut değilse yeni bir dosya oluşturur. |
| "+a" | Append and Update | Bu mod, bir dosyanın hem eklenebilmesine hem de güncellenmesine izin verir. Dosya mevcutsa içeriğini korur ve yeni verileri dosyanın sonuna ekler. Dosya mevcut değilse yeni bir dosya oluşturur. |

Daha önceden de söylediğimiz gibi bir dosya açıldığı mutlaka kapatılması gerekmektedir. Bu söz konusu dosyanın kapatılması fclose fonksiyonuyla yapılabilir. Örneğin:

#include <stdio.h>

int main(void)

{

FILE\* file = fopen("asal.txt","r");

//Dosyayla gerekli işlemler yapıldı

fclose(file);

return 0;

}

Dosyaya yazma standart kütüphanenin fprintf fonksiyonu kullanılarak yapılabilir. Örneğin aşağıdaki örnekte sayı asal ise dosyaya yazılacaktır:

#include <stdio.h>

#include <stdbool.h>

#include <stdlib.h>

bool is\_prime(const int val)

{

if(val <= 1)

return false;

if(val % 2 == 0)

return val == 2;

if(val % 3 == 0)

return val == 3;

if(val % 5 == 0)

return val == 5;

if(val % 7 == 0)

return val == 7;

if(val % 11 == 0)

return val == 11;

for(int i = 13;i \* i < val;i += 2)

{

if(val % i == 0)

return false;

}

return true;

}

int main(void)

{

FILE\* file = fopen("asal.txt","w");

if(!file)

{

printf("Dosya acilamadi\n");

exit(EXIT\_FAILURE);

}

for(int i = 0;i < 1000;++i) {

if (is\_prime(i))

fprintf(file, "%d" ,i);

}

fclose(file);

exit(EXIT\_SUCCESS);

}

Bu örnekten de az çok anlaşılabileceği gibi printf fonksiyonunun genel biçimi şu şekilde olacaktır:

fprintf(<fonksiyonun kendisi>,<tırnak içerisinde format ifadesi>, <değişken(ler)>);

Dosyadan okuma yapma da aşağıdaki örnekteki gibi yapılabilir:

1-)Önce kaynak dosyayı aşağıdaki kod ile oluşturalım.

#include <stdio.h>

typedef struct {

char name[50];

char surname[50];

int age;

}Person;

int main() {

FILE \*file = fopen("data.txt", "w");

if (file == NULL) {

printf("Dosya acilamadi!\n");

return 1;

}

Person parr[3] = {{"Berke","Sertel",25},{"Batuhan","Menkuer",26},{"Mustafa","Yilmaz",67}};

for(int i = 0;i < 3;++i)

{

fprintf(file,"%s ",parr[i].name);

fprintf(file,"%s ",parr[i].surname);

fprintf(file,"%d ",parr[i].age);

}

fclose(file);

return 0;

}

2-)Yaratılmış olan dosyadan okuma yapan kodu yazalım.

#include <stdio.h>

typedef struct {

char name[50];

char surname[50];

int age;

}Person;

int main() {

FILE \*file = fopen("data.txt", "r");

if (file == NULL) {

printf("Dosya acilamadi!\n");

return 1;

}

Person person;

while (fscanf(file, "%s %s %d", person.name, person.surname, &person.age) != EOF) {

printf("Ad: %s, Soyad: %s, Yas: %d\n", person.name, person.surname, person.age);

}

fclose(file);

return 0;

}