

KOD ÖRNEKLERİ:

Dosyadaki aranan kelimeyi bularak ondan kaç tane olduğunu veren program:

```
6-dosyada-kelime-arama.cpp x 7-dosyada-kelime-bul-degistir.cpp x fork.c x
1 #include <stdio.h>
2 #include <string.h>
3 #include <stdlib.h>
4 int main(int argc, char* argv[])
5 {
6
7     FILE *fp1;
8     int i, toplam, sıra=0, konum, bitti=0;
9     char c;
10    char *aranan;
11    aranan=(char*) malloc (sizeof(argv[1]));
12    strcpy(aranan,argv[1]);
13    fp1= fopen (argv[2], "r");
14    while(1)
15    {
16        toplam=0;
17        for(i=0;i<strlen(aranan);i++)
18        {
19            c = fgetc(fp1);
20            sıra++;
21            if(c==EOF)
22            {
23                bitti=1;
24            }
25            else
26            {
27                if(c==aranan[i])
28                {
29                    toplam++;
30                }
31                else
32                {
33                    break;
34                }
35            }
36        }
37        if(toplam==strlen(aranan))
38        {
39            konum=sıra-strlen(aranan)+1;
40            printf("Bulundu %d\n",konum);
41        }
42        if(bitti==1)
43        {
44            break;
45        }
46    }
47    fclose(fp1);
48    return 0;
49 }
```

Program parametreler üzerinden çalışıyor, ana programımıza 2 tane parametre gönderiyoruz. 1. parametre dosyada aranacak kelimeyi 2. si ise hangi dosyada arayacağımızı söylüyor.

FILE *fp1; file içinde bir değişken tanımladık(pointer tipinde)

int i, toplam, sıra=0, konum, bitti=0; "sıra" dosyadaki cursor pozisyonunu tutmak için, "konum" dosyada hareket etmek için kullanılan bildi, dosya sonuna gelince "bitti"

char c;

char *aranan; char tipinde pointer değişken tanımladık. (c de dinamik bir dizi oluşturabilmek için lazım çünkü ilkeldir C dili

aranan=(char*) malloc (sizeof(argv[1])); aranan değişkenine, char* ile bellekteki boyutunu ve tipini(string) ve malloc fonk ile boyu tunu veriyoruz, argv[1] ile aranan

strcpy(aranan,argv[1]); aranan icine aktardık

fp1= fopen (argv[2], "r"); değer, kısaca buranın her yerine kullanıcının girdiği aranacak kelimenin boyu kadar bellekte yer ayırdık.

while(1) dosya olana kadar fp1 ile aranacak olan dosyayı okuma modunda açıp alıp içine attık

toplam=0;

for(i=0;i<strlen(aranan);i++) kelimenin boyu kadar dönüyoruz

c = fgetc(fp1);

sıra++; en son nerede kaldığını tutar cursor olarak

if(c==EOF) dosya sonuna geldik mi? geldik ise bitti 1 setle ve while ı sonlandır.

bitti=1;

else

{

if(c==aranan[i])

toplam++;

çekersek ve hepsi sırayla eşit olursa bu kelimeyi bulmuşuz demektir.

else

break;

}

if(toplam==strlen(aranan)) eğer toplam değeri aranana eşitse bulduk demektir.

{

konum=sıra-strlen(aranan)+1;

printf("Bulundu %d\n",konum); tamamen çıkartınca kelimenin ilk karakterine denk gelsin

}

if(bitti==1)

break;

}

fclose(fp1);

return 0;

```
neo@ubuntu: ~
neo@ubuntu:~$ gcc -o arabul 6-dosyada-kelime-arama.cpp
neo@ubuntu:~$ ./arabul deneme 27ekim
Bulundu 1 1 tane bulundu
Bulundu 126 126 byte olarak bulundu
neo@ubuntu:~$ ./arabul 21 27ekim
Bulundu 17
Bulundu 19
Bulundu 46
Bulundu 48
Bulundu 52
Bulundu 54
neo@ubuntu:~$
```

Dosyadaki kelimeleri bulan ve deęiřtiren program:

```
6-dosyada-kelime-arama.cpp x 7-dosyada-kelime-bul-degistir.cpp x fork.c x 27ekim x
5 int main(int argc, char* argv[])
6 {
7     FILE *fp1,*fp2;
8     int i,toplam,sira=0,konum,bitti=0;
9     char c;
10    char *aranan;
11    char *degistirilecek;
12    aranan=(char*) malloc (sizeof(argv[1]));
13    degistirilecek=(char*) malloc (sizeof(argv[2]));
14    strcpy(aranan,argv[1]);
15    strcpy(degistirilecek,argv[2]);
16    fp1= fopen (argv[3], "r+");
17    fp2= fopen ("yenidosyamiz","w"); deęistirilmiř řeklinde olan dosyayı iinde tutacak dosya
18    while(1)
19    {
20        toplam=0;
21        int ictur=0;
22        for(i=0;i<strlen(aranan);i++) burası dosyada kelime aramayla aynı soldaki for ds.
23        {
24            c = fgetc(fp1);
25            sıra++; ictur++;
26            if(c==EOF)
27                bitti=1;
28            else
29            {
30                if(c==aranan[i])
31                    toplam++;
32                else
33                    break;
34            }
35        }
36        if(bitti==1)
37            break;
38        if(toplam==strlen(aranan)) aranan kelime bulunursa
39        {
40            konum=sıra-strlen(aranan)+1;
41            printf("Bulundu %d\\n",konum); bulunduęu yeri yazdım
42            fputs(degistirilecek,fp2); kelimenin bulunduęu yere yazılacaktır olarak bu iřlemi yaşıyoruz
43        }
44        else eęer aranan kelime bulunamazsa
45        {
46            konum=sıra-ictur;
47            //printf("sıra=>%d ictur=>%d konum=>%d",sıra,ictur,konum);
48            fseek(fp1,konum,SEEK_SET);
49            for(i=0;i<ictur;i++)
50            {
51                char p=fgetc(fp1);
52                //printf("%c\\n",p);
53                fprintf(fp2,"%c",p);
54            }
55        }
56    }
57    fclose(fp1); dosya aık diyip hata verir o yzden bu iřlemi yaotık
58    fclose(fp2);
59    rename("yenidosyamiz",argv[3]);yenidosyamiz isimli dosyanın ismini argv[3] olarak deęiřtiriyoruz, kullanıcı hangi dosyayı aarsa onu yeni dosyamız olarak grmesidir.
60    return 0;
61 }
```

```
neo@ubuntu:~
neo@ubuntu:~$ gcc -o buldegis 7-dosyada-kelime-bul-degistir.cpp
neo@ubuntu:~$ ./buldegis satirın mısram 27ekin
Bulundu 60
Bulundu 133
neo@ubuntu:~$
```

satırmları mısram olarak deęiřtirdi.

Process Management sf 3

Sistem aęrılarını nelerle uęrařır?

- *Bir processin oluřturulması.
- *Bir processi icra edecek řekilde ayarlamak.
- *Bir bařka iřlem bir processin sonlandırılmasını beklemek

- *Processler arası haberleşme
 - *Processi bitirmek
 - *Bir processe sinyaller göndermek.
-

Process ID sadece bir sayıdır getpid() ile ulaşılabilir

ps ile Id li processleri listeleyebiliyoruz.

her processin bir parenti var getppid() ile ulaşılabilir o id ye. (Get process parent id kısaltması)

getpgrp() her process bir grubun üyesidir, bunun ile de group id getirebiliyoruz.

Processlerin Oluşturulması:

pid = fork() ile bir tane yeni çocuk process üretiyor, o çocuk process i o anda koşan bir processin kopyasını oluşturuyor. Parent ve çocuk aynı konumda koşturmaya devam eder.

Aynı değişken değerlerini alarak aynı dosyaları açabiliyorlar.

Tek fark çocuk yeni bir PID değerine sahip olur.

Eğer forkdan dönen değer 0 ise çocuk process anlamına gelir.

Eğer parentin içerisinde ise childin PID değeri döner.

Pid = wait(&status) ile, çağıran processin icrasını durduruyor, yani parent processin icrası durdurulur. Kısaca, parentin icrasını, üretilen çocuklardan herhangi birinin icrası sonlanırsa onun process id sini almış oluyoruz.

Pid = wait (pid,&status,options)Çağırılan processin icrasını, belirli bir çocuk process sonlanırsa durdurur.(spesifik bir çocuk)

```
pid = fork();
if( pid == 0)
    execl("./program", "program",
        arg1, NULL);
else
    pid = wait(& status);continue
    execution
```

yandaki kod parçasında, process çocuk ise programın versiyonunu oluştur, eğer değilse çocuk sonlanıncaya kadar bekle.

Exit(status) açılmış tüm processleri kapatır.

Bütün dosya açıklamalarını da kapatır.

Parent processe de processin durumunu dönderir.

SİNYAL İSİMLERİ:

INT Kesme isteği yollar

KILL Processi zorla bitiriyoruz

ALRM Alarm

TERM Yakalanabilir bir icra sonucu düzgün bir giriş

QUIT Çıkış

ABRT iptal

HUP Asılı kalmak

Sinyaller, sinyal tutucular tarafından yakalanır. (Signal Handler)

Sf12-----

Sending signals to process

SIGINT adında bir

- Send a signal using kill function
 - `retval = kill(pid, signal)`
- Example
 - `kill(SIGINT, 1234);`
 - Sends an interrupt (signal) of type SIGINT
 - Does not block sending process
 - The process, whose ID is 1234, gets the signal.

fork.c (-) - gedit



Open Save Undo redo

6-dosyada-kelime-arama.cpp x 7-dosyada-kelime-bul-degistir.cpp x fork.c x 27ekim x

```
1 #include <stdio.h>
2 #include <unistd.h>
3 #include <stdlib.h>
4 #include <sys/wait.h>
5 int main ( void )
6 {
7     int f;
8     f= fork();
9     if(f==0)
10 {
11 printf("Cocuk : Proses no: %d \n" , getpid() );
12 sleep(10);
13 printf("Cocuk : Anne proses no : %d \n" , getppid() );
14 }
15 else
16 {
17 printf("Anne : Proses no : %d \n", getpid() );
18 printf("Anne : Cocuk proses no : %d \n" , f );
19 printf("Anne : Anne proses no : %d \n" , getppid() );
20 wait(NULL);
21 printf("Anne : Sonlanıyorum . . . \n");
22 exit(0);
23 }
24 return 0;
25 }
```