KOD ÖRNEKLERİ:

Dosyadaki aranan kelimeyi bularak ondan kaç tane olduğunu veren program:

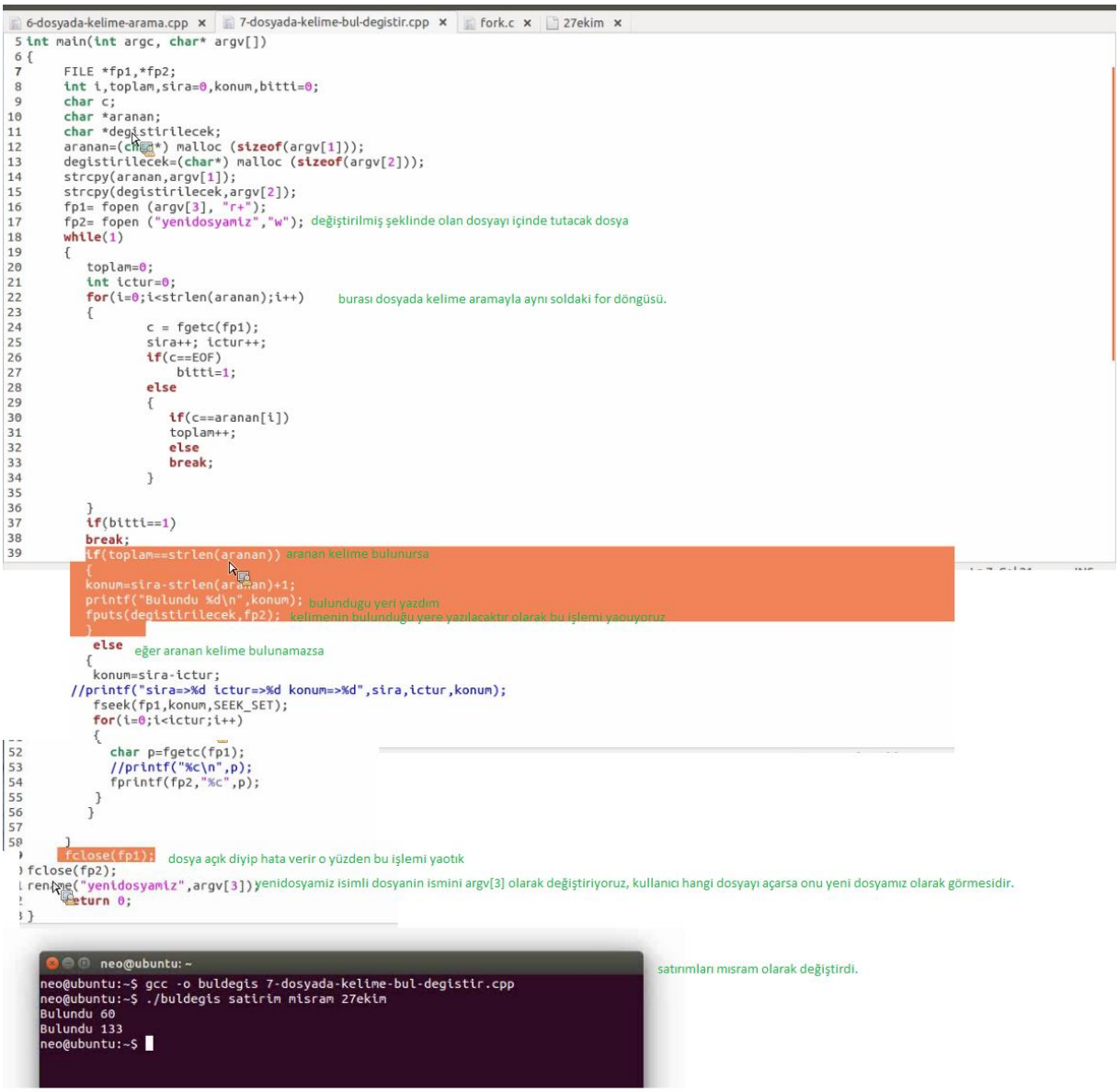
6-dosyada-kelime-arama.cpp × 7-dosyada-kelime-bul-degistir.cpp × fork.c ×

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main(int argc, char* argv[])
{

    FILE *fp1; file tipinde bir değişken tanımladık(pointer tipinde)
    int i,toplam,sira=0,konum,bitti=0;"sira" 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]); değeri aranan icine aktardık değer, kısaca buranın her yerine  kullanıcının girdiği aranacak kelimenin boyu kadar bellekte yer ayırdık.
    fp1= fopen (argv[2], "r"); fp1 ile aranacak olan dosyayı okuma modunda açıp alıp içine attık
    while(1) dosya olana kadar
    {
       toplam=0;
       for(i=0;i<strlen(aranan);i++) kelimenin boyu kadar dönüyoruz
       {
               c = fgetc(fp1);
               sira++; en son nerede kaldığımı 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]) gelen karakteri aranılan karakterin ilk harfiyle karşılaştırıyoruz aynı ise, toplam ı 1 artırdık ki aradığımız kelimenin boyu kadar karakter
                  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=sira-strlen(aranan)+1; tamamen çıkartınca kelimenin ilk karakterine denk gelsin
       printf("Bulundu %d\n",konum);
       }
       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 değiştiren program:



Process Management sf 3

Sistem çağrıları nelerle uğraşır?

*Bir processin oluşturulması.

*Bir processi icra edecek şekilde ayarlamak.

*Bir başka işlem 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 getppied() 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 koşmaya 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)Çağrılan 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

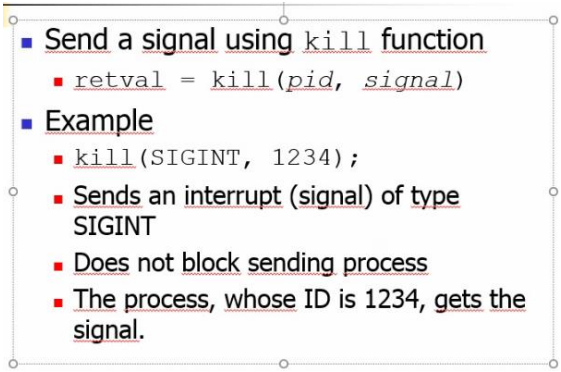
HUP Asılı kalmak

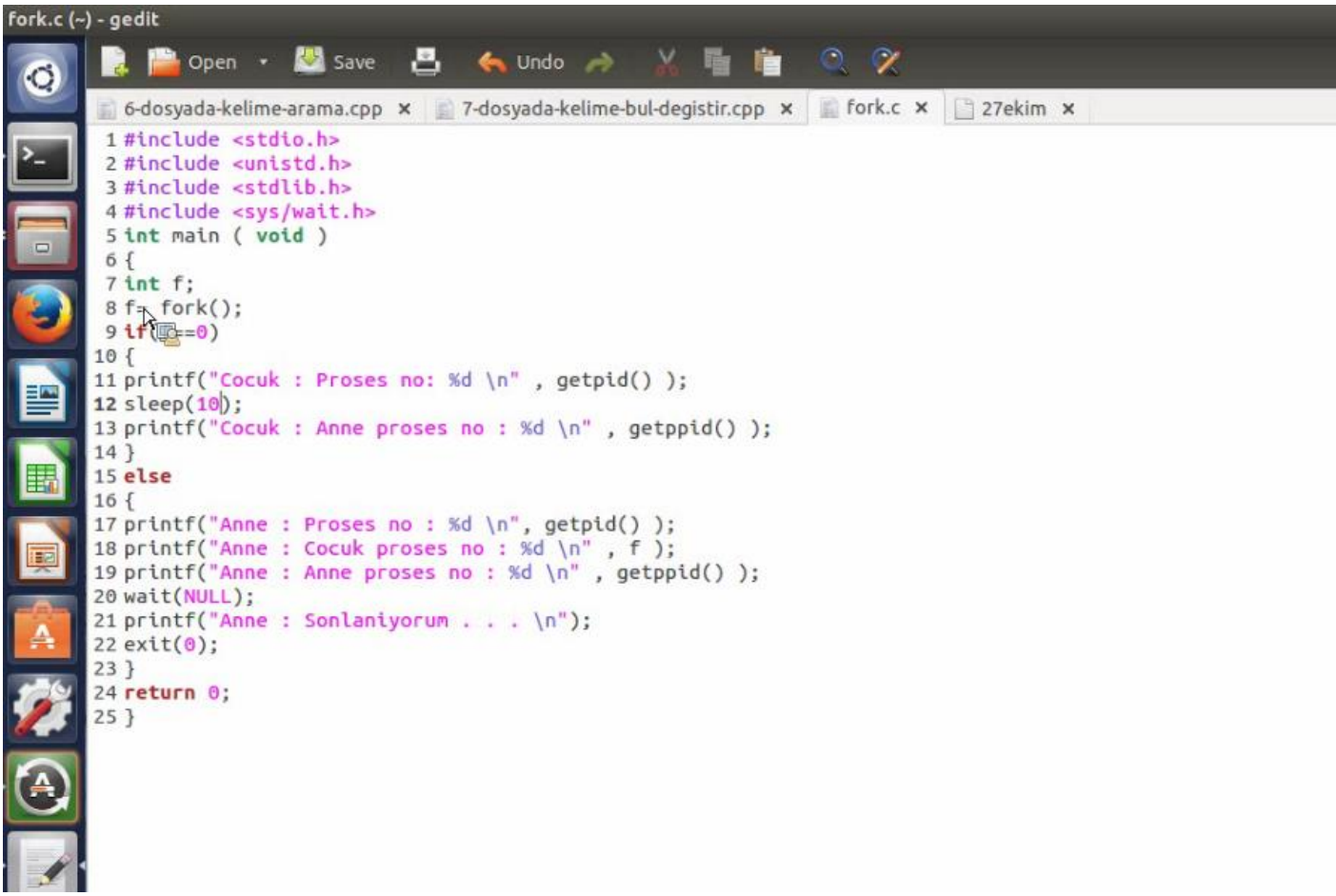
Sinyaller, sinyal tutucular tarafından yakalanır. (Signal Handler)

Sf12----------

Sending signals to process

SIGINT adında bir




fork.c (~) - gedit
Open
Save
Undo
6-dosyada-kelime-arama.cpp
7-dosyada-kelime-bul-degistir.cpp
fork.c
27ekim
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>
int main (void)
{
int f;
f= fork();
if(f==0)
{
printf("Cocuk : Proses no: %d \n" , getpid());
sleep(10);
printf("Cocuk : Anne proses no : %d \n" , getppid());
}
else
{
printf("Anne : Proses no : %d \n", getpid());
printf("Anne : Cocuk proses no : %d \n" , f);
printf("Anne : Anne proses no : %d \n" , getppid());
wait(NULL);
printf("Anne : Sonlaniyorum . . . \n");
exit(0);
}
return 0;
}