

\*Fprintf formatlı yazmamıza yarıyor

```
1 /* fprintf example */
2 #include <stdio.h>
3
4 int main ()
5 {
6     FILE * pFile;
7     int n;
8     char name [100];
9
10    pFile = fopen ("myfile.txt","w");
11    for (n=0 ; n<3 ; n++)
12    {
13        puts ("please, enter a name: ");
14        gets (name);
15        fprintf (pFile, "Name %d [%-10.10s]\n",n+1,name);
16    }
17    fclose (pFile);
18
19    return 0;
20 }
```

dümdüz yazmak yerine araya boşluklar koy, tab koy

vs onun için kullanılıyor. 3 parametre alır, 1. Parametre nereye yazılacak, 2. Parametre formatlı yazım tipi, sonra da yazdırılacak elemanlar.

---

\*Fputs 2 parametre alıyor, 1. Parametre yazdırılacak pointer dizisi, 2. Parametre ise yazdırılacak dosya.

```
1 /* fputs example */
2 #include <stdio.h>
3
4 int main ()
5 {
6     FILE * pFile;
7     char sentence [256];
8
9     printf ("Enter sentence to append: ");
10    fgets (sentence,256,stdin);
11    pFile = fopen ("mylog.txt","a");
12    fputs (sentence,pFile);
13    fclose (pFile);
14    return 0;
15 }
```

FILE tipinde bir dosya değişken oluşturulmuş. 256 karakterlik bir dizi oluşturulmuş. Eklemek istediğiniz diziyi ekleyin diyor, standart girişten (stdin) alıyor ve 256 birimlik olarak. Fgets 3 parametre alıyor. 1. Parametre dizi gelmesi lazım pointer veya normal dizi, 2. Parametrede dosyadan çekilecek karakter sayısını belirtiyoruz.

konsoldan yazılanın 256 karakterini alıyor, aldıktan sonrada sentence isimli diziye aktarıyor fgets ile. Eğer 256 karakteri aşmazsam konsoldan girilen ifade diziye yerleşmiş oluyor. File tipinde bir pointer tanımlanmıştı, fopen ile onu açtı

---

\*Fopen 2 parametre alıyor, 1. Si dosya adı, 2.si ise hangi modda açılacağı. var olan dosya varsa içini boşaltır, yoksa içini

---

\*Fputc ve fgetc karakter okuma ve koyma yapıyorlar.

\*fopen dosya varsa açar yoksa oluşturup açar

\* while(c!=EOF) dosya sonuna gelene kadar demek.

```

1 /* fscanf example */
2 #include <stdio.h>
3
4 int main ()
5 {
6     char str [80];
7     float f;
8     FILE * pFile;
9
10    pFile = fopen ("myfile.txt","w+");
11    fprintf (pFile, "%f%s", 3.1416, "PI");
12    rewind (pFile);
13    fscanf (pFile, "%f", &f);
14    fscanf (pFile, "%s", str);
15    fclose (pFile);
16    printf ("I have read: %f and %s \n",f,str);
17    return 0;
18 }

```

Rewind pozisyon indikatörünü yani cursoru başa konumlandırır dosyada.

%f float değeri okuyacağımı gösterir yukarıdaki kodda, &i ile de değişkenin adresini veriyoruz. Ampersant ile kısaca adres bilgisi verilir.

%d %i integer

%u işaretli yani pozitif tam sayı

%o oktal sayı

%x hexadecimal bir sayı

%c karakter

%s karakter veya string dizisi

%f float

%e double

%g %G double veya float

%% yüzde karakteri yazmaya yarar.

Sqrt(2ç0) ile karakök alınır.

### AKREP YELKOVAN ÖRNEĞİ:

```
2-dizi-olusturma-ve-donguler x
1 #!/ bin/ bash
2
3 declare -a dizi
4 j=0
5 for i in $*
6 do
7     echo $i
8     dizi[j]=$i
9     ((j+=1))
10 done
11
12 k=0
13 while [ $k -lt ${#dizi[*]} ]
14 do
15     printf "%s" "${dizi[k]}"
16     ((k+=1))
17 done
18 printf "\n"
19
20 ilkeleman=$1
21 ilkelemanboy=${#ilkeleman}
22 echo $ilkeleman
23 echo $ilkelemanboy
24
```

## DİZİ OLUŞTURMA VE DÖNGÜLER:

```
celal5 x 1-akrep-yelkovan-aci-hesaplama.c x
1 #include <stdio.h>      Yelkovan ilerledikçe akrep te ilerler. buna hesaplama yapmamız lazım.
2 #include <string.h>
3 #include <stdlib.h>      Atoi tanımı için include yapılır stdlib,
4 int main(int argc, char* argv[])
5 {
6     if(argc==3)
7     {
8         double akrep=atoi(argv[1])*12; atoi, string türünü int çeviriyor
9         double yelkovan=atoi(argv[2])*60;
10        double aciclakrep=0;
11        double acictiyelkovan=0;
12        acictiyelkovan+=(yelkovan)*6; tüm saat 360 derecedir, aslında yelkovani 6 ile çarpınca aç buluyoruz, mesela 15:00 için 6 ile çarparsak 90 buluyoruz. yani sağa dönük 90 derec
13        aciclakrep+=akrep*30+((yelkovan*6))*30/360; her 1 saatlik ara 30 derece olduğu için bu işlemi yaptık.
14        double fark=acictiyelkovan-aciclakrep;
15        if(fark<0)        aç farkını bulmak için, biri diğerinden ileride ise -1 ile çarpıyoruz ki pozitif olsun.
16        fark=fark*-1;
17        if(fark>180)
18        fark=360-fark;
19        printf("Aradaki Fark:%.2f\n",fark);
20    }
21    else
22        printf("Akrep ve Yelkovan Değerlerini Giriniz\n");
23    return 0;
24 }
```

## BU GÜN OLUŞTURULAN DOSYALAR:

```
3-bugun-olusturulan...alar-dizin-argumanli x
1 dizin=$1
2 if [[ $dizin != "" ]] arguman boş mu dolu mu
3 then
4 if [[ -d $dizin ]] dizi n mi
5 then
6 cd $dizin
7 else
8 echo "Dizin Değil"
9 exit 1 hata varsa işlemi burada sonlandırıyor
10 fi
11 fi
12 s=0
13 tarih=$(date) bash kodundan gelen date i tarih e atadık
14 ay=${tarih:4:3} 4 satır sağa gidip ayı yakaladı
15 gun=${tarih:8:2} 8 den sonra 2 tane gitti ve günü yakaladı
16 ls -al > temp_file_bugun tempfile adında bir dosya oluşturdu ve bulunduğumuz dizine yazıldı
17 while read a1 a2 a3 a4 a5 a6 a7 a8 a9 a1 e okuma yazma izinlerini, a2 ye root falan vs sırayla içlerine atıyor.
18 do
19 if [[ $a6 == $ay && $a7 == $gun && $a9 != "temp_file_bugun" ]] günümüzdeki tarih değerleri ile dosyaları karşılaştırıyor. ve kod sırasında yazdığımız "temp_file_bugun" dosyası
20 then
21 s=$((s+1)) # s="expr "$s+$a5" | bc"
22 echo $a9 dosyanın ismini yazdırıyoruz
23 # printf "%s %s\n" "$a5 $a9"
24 fi
25 done < temp_file_bugun Bu dosyayı döngünün içine satır bazlı olarak veriyor, her 1 turda 1 satır çekecek
26 rm temp_file_bugun bizim oluşturduğumu geçici dosyayı siliyoruz.
```

```
neo@ubuntu:~$ bash 3-bugun-olusturulan-dosyalar-dizin-argumanli zczczczx
Dizin Değil
neo@ubuntu:~$ bash 3-bugun-olusturulan-dosyalar-dizin-argumanli
.
1-akrep-yelkovan-aci-hesaplama.c
1-akrep-yelkovan-aci-hesaplama.c-
.bash_history
saatcl
neo@ubuntu:~$ bash 3-bugun-olusturulan-dosyalar-dizin-argumanli Downloads/
.
1-akrep-yelkovan-aci-hesaplama.c
neo@ubuntu:~$
```

## DOSYA BOYUTU HESAPLAMA

```
3-bugun-olusturulan...alar-dizin-argumanli x 4-dosya-hesap-baska x
1 s=0
2 ls -al > list
3 while read a1 a2 a3 a4 a5 a6 a7 a8 a9
4 do
5 if [[ ( -f $a9 ) && $a5 != 0 ]] Dosya mı değil mi onu kontrol ediyoruz, yani dosyası al
6 then
7 s=$((s+1)) s="expr "$s+$a5" | bc" ilk turda boyutları toplamış ve dönüştürme işlemi yapmış bc ile, ve s değişkenine atmış
8 printf "%s %s\n" "$a5 $a9" yakalanan dosyanın ismini ve boyutunu yazdırdım s ler string demek değişken olan s ile ilgisi yok
9 fi
10 done < list
11 rm list
12 echo "Total size: $s byte"
```

```
neo@ubuntu:~$ ls -al
total 188
-rwxr-xr-x 1 neo neo 8646 Jan 12 04:47 saatcl
-rwxr-xr-x 1 neo neo 278 Jan 16 2019 soru2.sh-
-rwxr-xr-x 2 neo neo 4096 Dec 7 2018 Templates
-rwxr-xr-x 2 neo neo 4096 Dec 7 2018 Videos
-rw-r--r-- 1 neo neo 51 Jan 24 2020 .Xauthority
-rw-r--r-- 1 neo neo 108 Jan 24 2020 .xsession-errors
-rw-r--r-- 1 neo neo 828 Jan 24 2020 .xsession-errors.old
-rwxr-xr-x 15 neo neo 4096 Jan 12 05:22 .
-rwxr-xr-x 3 root root 4096 Nov 30 2018 .
-rw-r--r-- 1 neo neo 469 Jan 12 04:47 1-akrep-yelkovan-aci-hesaplama.c
-rw-r--r-- 1 neo neo 491 Jan 12 04:45 1-akrep-yelkovan-aci-hesaplama.c-
-rw-r--r-- 1 neo neo 489 Jan 11 15:46 1-akrep-yelkovan-aci-hesaplama.cpp-
-rw-r--r-- 1 neo neo 244 Jan 11 15:50 2-dizi-olusturma-ve-donguler
-rw-r--r-- 1 neo neo 419 Jan 12 05:19 3-bugun-olusturulan-dosyalar-dizin-argumanli
-rw-r--r-- 1 neo neo 392 Jan 12 05:19 3-bugun-olusturulan-dosyalar-dizin-argumanli-
-rw-r--r-- 1 neo neo 213 Jan 11 16:30 4-dosya-hesap-baska
-rw-r--r-- 1 neo neo 391 Jan 11 16:33 5-dosyadan-farkliokuma.cpp
-rw-r--r-- 1 neo neo 696 Jan 11 16:38 6-dosyada-kelime-arama.cpp
-rw-r--r-- 1 neo neo 153 Dec 20 2018 awkkodu-
-rw-r--r-- 1 neo neo 3494 Jan 12 05:20 .bash_history
-rw-r--r-- 1 neo neo 220 Nov 30 2018 .bash_logout
-rw-r--r-- 1 neo neo 3637 Nov 30 2018 .bashrc
-rwxr-xr-x 14 neo neo 4096 Jan 28 2019 .cache
-rw-r--r-- 1 neo neo 401 Nov 24 01:24 celal2
-rw-r--r-- 1 neo neo 385 Nov 2 2015 celal3
-rw-r--r-- 1 neo neo 589 Dec 15 04:37 celal4
-rw-r--r-- 1 neo neo 601 Dec 15 04:36 celal4-
-rw-r--r-- 1 neo neo 247 Dec 15 05:21 celal5
-rw-r--r-- 1 neo neo 247 Dec 15 05:21 celal5-
```

```
neo@ubuntu:~$ bash 4-dosya-hesap-baska
469 1-akrep-yelkovan-aci-hesaplama.c
491 1-akrep-yelkovan-aci-hesaplama.c-
489 1-akrep-yelkovan-aci-hesaplama.cpp-
244 2-dizi-olusturma-ve-donguler
419 3-bugun-olusturulan-dosyalar-dizin-argumanli
392 3-bugun-olusturulan-dosyalar-dizin-argumanli-
213 4-dosya-hesap-baska
391 5-dosyadan-farkliokuma.cpp
696 6-dosyada-kelime-arama.cpp
153 awkkodu-
3494 .bash_history
220 .bash_logout
3637 .bashrc
401 celal2
385 celal3
589 celal4
601 celal4-
247 celal5
247 celal5-
88 cikttcl
25 .dmrc
54 dosya-
229 dosya1-
1272 .ICEAuthority
675 .profile
8646 saatcl
278 soru2.sh-
51 .Xauthority
108 .xsession-errors
828 .xsession-errors.old
Total size: 6032 byte
neo@ubuntu:~$
```

## DOSYA OKUNABİLİYOR MU:

```
3-bugun-olusturulan...alar-dizin-argumanli x 4-dosya-hesap-baska x 5-dosyadan-farkli-okuma.c x
1 #include <stdio.h>
2
3 int main(int argc, char* argv[])
4 {
5     FILE *fpr;
6
7     char str[10];
8     fpr = fopen(argv[1], "r"); dosyayi okuma modunda açtım
9
10
11     if (fpr == NULL)
12     {
13         puts("Dosyayı açarken problem oldu");
14     }
15
16     while(1)
17     {
18         if(fgets(str, 10, fpr) == NULL)
19             break;
20         else
21         {
22             printf("%s", str);
23         }
24     }
25
26     fclose(fpr);
27     return 0;
28 }
```

## KELİMEYİ ARAYARAK KAÇ TANE OLDUĞUNU BULAN 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     FILE *fp1;
7     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"
8     char c;
9     char *aranan; char tipinde pointer değişken tanımladık. (c de dinamik bir dizi oluşturabilmek için lazım çünkü ilkeldir C dili
10    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
11    strcpy(aranan,argv[1]);
12    fp1= fopen (argv[2], "r");
13    while(1) dosya olana kadar fp1 ile aranacak olan dosyayı okuma modunda açıp alıp içine attık
14    {
15        toplam=0;
16        for(i=0;i<strlen(aranan);i++) kelimenin boyu kadar dönüyoruz
17        {
18            c = fgetc(fp1);
19            sıra++; en son nerede kaldığımızı tutar cursor olarak
20            if(c==EOF) dosya sonuna geldik mi? geldik ise bitti 1 setle ve while ı sonlandır.
21            bitti=1;
22            else
23            {
24                if(c==aranan[i])
25                {
26                    toplam++;
27                    else
28                    break;
29                }
30            }
31            if(toplam==strlen(aranan)) eğer toplam değeri aranana eşitse bulduk demektir.
32            {
33                konum=sıra-strlen(aranan)+1;
34                printf("Bulundu %d\n",konum);
35            }
36            if(bitti==1)
37            break;
38        }
39        fclose(fp1);
40        return 0;
41    }
```

```
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:~$
```