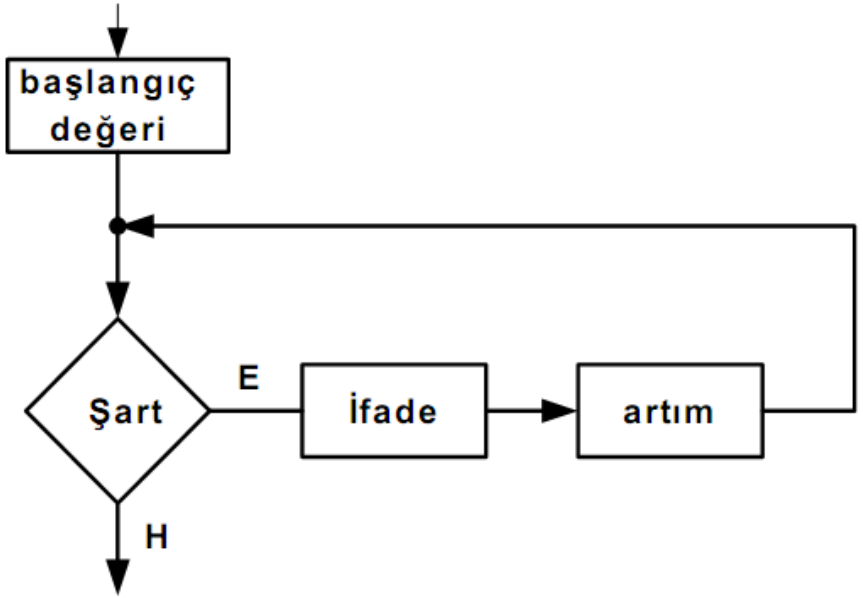
for döngüsü




```
for (başlangıç değeri; şart; artım)
ifade;
```

```
for (başlangıç değeri; şart; artım)
{
ifadeler;
}
```

Başlangıç değeri; koşul içinde tanımladığımız değişkene ilk değerin atanmasını sağlar.

Koşul: Döngünün devam edip etmeyeceğine karar verir. şart doğru ise devam eder. Yanlış ise döngüden çıkılır.

Artım: Koşul değişkenin her bir döngüde arttırılıp azaltılacağı belirtir.

Örnek 1'den 100'e kadar olan sayıların toplamı.

```
j =0;
for (i=1; i<=100; i=i+1)
    j =j+i;
printf("Toplam %d",j);
```

Örnek Girilen sayının faktöriyelini bulunuz.

```
    fact =1;
    for (j=1; j<=i; j++)
            fact =fact*j;
    printf("Faktöriyel  =%f",fact);
}
```

1. Toplam T, sayılar da i diye çağırılsın.
2. Başlangıçta T'nin değeri 0 ve i'nin değeri 1 olsun.
3. i'nin değerini T'ye ekle.
4. i'nin değerini 1 arttır.
5. Eğer i'nin değeri 100'den büyük değil ise 3. adımdan devam et.
6. T'nin değerini yaz.

```
# include <stdio.h>
#include <conio.h>
main ( )
{
int i,T=0;
for (i=1; i<=100; i++)
{
T+=i;
}
printf ("%d",T);
}
```

SORU Çarpım tablosu. (içi içe döngüler)

Çarpım tablosu

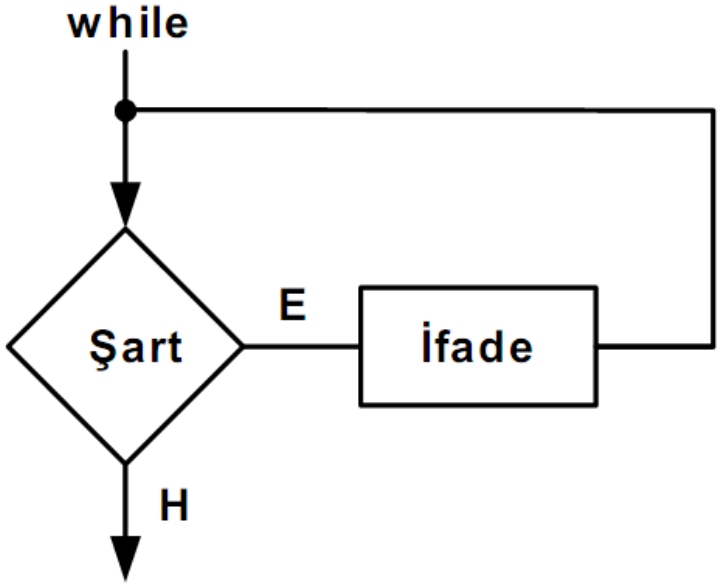
```c
#include<stdio.h>
#include<stdlib.h>
main()
        {
            int i,j;
            for (i=1; i<=10; i++) {
                for (j =1; j<=10; j++)
                        printf("%4.0d",i*j);

            printf("\n");
            }
        }
```

WHILE DÖNGÜSÜ

```
while (şart)
{
ifade 1;
 ifade 2;
}
```

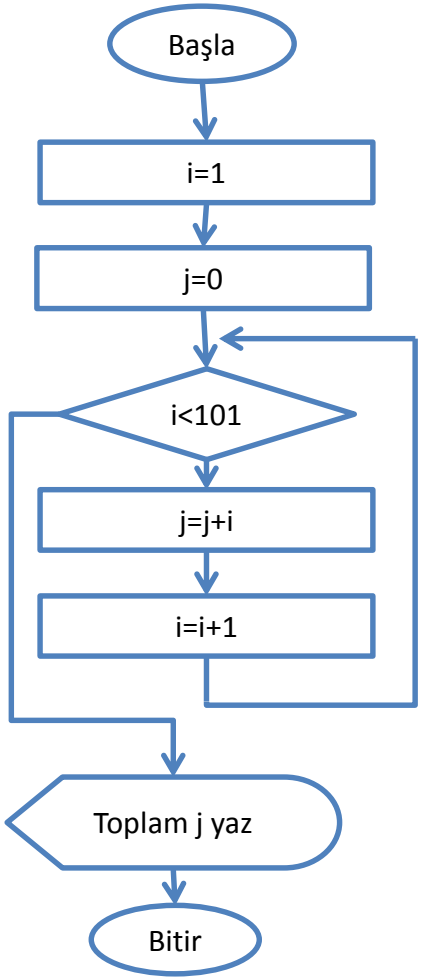
*Aynı işlem if-goto yapısı kullanılarak da gerçekleştirilebilir.




```
x:
 if (şart)
{
ifade 1;
ifade2;
goto x;
}
```

Örnek 1'den 100'e kadar olan sayıların toplamı.

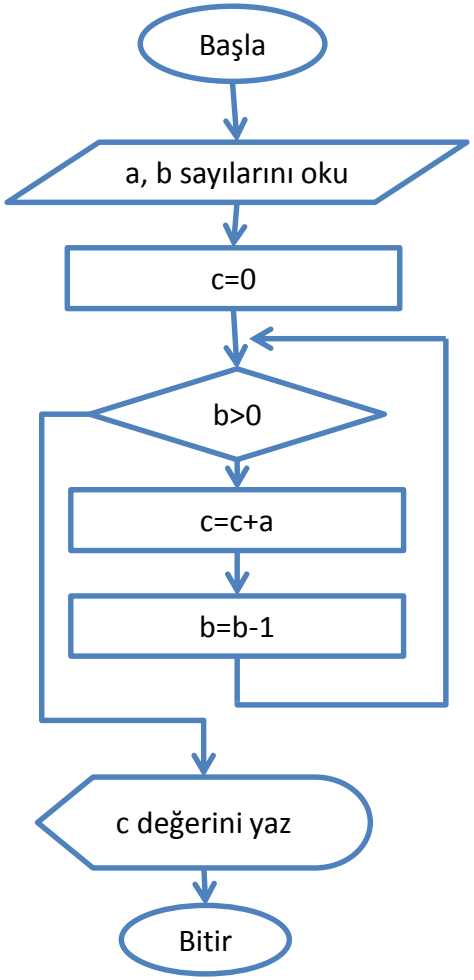
1. i =1
2. j = 0
3. i < 101 olduğu sürece
   - 3.1 j = j + i
   - 3.2 i = i + 1
4. Toplam j ‘ yi yaz




```c
#include<stdio.h>
#include<stdlib.h>
main()
{
int i, j;
i =1;
j = 0;
while (i<101) {
j =j+i;
i =i+1;
}
printf("Toplam = %d",j);
}
```

Örnek **Toplama ve çıkartma kullanarak çarpma işlemini gerçekleyiniz.**

1. a ve b sayılarını oku
2. c =0
3. b>0 olduğu sürece tekrarla
    - 3.1. c=c + a
    - 3.2. b = b-1
4. c değerini yaz ve dur

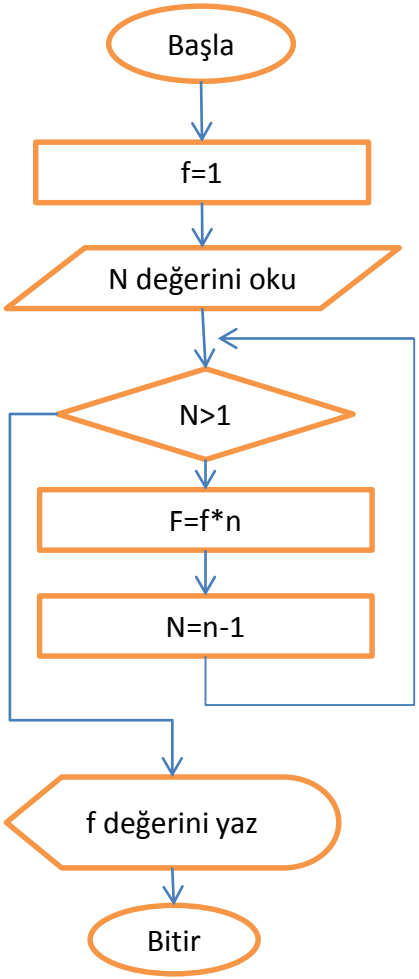


```c
#include<stdio.h>
#include<stdlib.h>
main()
{
  int a, b, c;
  printf ("iki sayıyı giriniz ");
  scanf("%d%d", &a, &b);
  c = 0;
  while (b > 0) {
    c = c + a;
    b = b - 1;
  }
  printf("Sonuç = %d\n", c);
}
```

Örnek : Girilen sayının faktöriyelini hesaplayan programı yazınız.

ALGORİTMA

1. n değerini oku
2. F=1
3. n >1 olduğu sürece tekrarla
   .3.1. F=F*n
   3.2. n= n-1
4. F değerini yaz




```c
#include<stdio.h>
#include<stdlib.h>
main()
{
  int n;
  long f;
  printf ("sayıyı giriniz ");
  scanf("%d", &n);
  f = 1;
  while (n > 1) {
    f = f * n;
    n = n - 1;
  }
  printf("SonuC = %d\n", f);
}
```

Örnek 2014 yılı itibarı ile ülke nüfüsu 76 milyondur. Yıllık nüfüs artış oranı %2.1 dir. Sonraki 10 yılda ülke nüfusunu yıllara göre listeleyen program.

```c
/* Nufus Tablosu  */
#include <stdio.h>
main()
{
  int       i;   /* sayac */
  int     yil;   /* yillar */
  float nufus;   /* nufus miktari */
  float artis;   /* artis orani */
  artis = 0.021;
  yil = 2014;
  nufus = 76000000;
  printf("%d - %10.0f\n",yil,nufus);
  i = 1;
  while (i < 11)
    {
      nufus = nufus * (1 + artis);
      printf("%d - %10.0f\n",yil + i,nufus);
      i = i + 1;
    }
}
```

DO - WHILE DÖNGÜSÜ:

Bu döngü while döngüsünün biraz değiştirilmiş halidir. Do-while döngüsünde karşılaştırma işlemi, döngünün sonunda gerçekleşir. Bunun sonucu olarak döngünün içine en az bir defa girilmiş olur. Yapısı aşağıdaki gibidir.

do

cümle

while (kosul);

do'' nun altındaki cümle kısmındaki komut satırları birden fazla olursa diğer döngülerde olduğu gibi " { } " içine alıyoruz. Bunu kullanmamız kodları okuma da ve ayırma da daha çok isimize yarayacaktir.

```
do
{
cümle
cümle
cümle
...
}
while ( kosul );
```

Kodları yazdığımızda, komut sırası do''ya geldiği zaman, do'' dan sonraki komutun döngünün başı olduğunu belirtiyor. Diğerlerinden farklı (for, While) olarak döngüye giriş yapıyor, yani hiçbir kontrol yapmadan en az bir defa döngünün içine girmiş oluyoruz. While''e geldiğinde ise koşulu kontrol ediyor, eğer doğru ise döngünün başındaki komuta giderek yeniden komutları isliyor. Eğer koşul kontrolü yanlış ise while''den bir sonra ki komutu veya komutları isleyip döngüden çıkıyor. Simdi bu söylediklerimizi örnek üzerinde gösterelim.

```
#include <iostream>
using namespace std;
int main ()
{
unsigned long x;
do {
cout<< "Bir sayi giriniz ( Durdurmak icin 0 ) : ";
cin>> x;
cout<< "Girdiginiz sayi: " << x << "\n";
}
while (x != 0);
return 0;
}
```

ÖDEV 5.

Girilen tamsayının mükemmel sayı olup olmadığının söyleyen programı yazınız.
(mükemmel sayı = tam bölenlerin toplamı sayının kendisine eşit)

Girilen tamsayının kaç basamaktan oluştuğunu söyleyen programı yazınız.

Girilen tamsayı içerisinde kaç tane 1 olduğunu söyleyen programı yazınız.

Girilen tamsayının son üç basamağını yuvarlayan programı yazınız.

son üç basamağı >=500 > 1000 e, <500 ise 0 a yuvarlayacak
(2560 → 3000, 2490 → 2000)

Arka sayfayı takip ediniz

Programınız bir çarpım tablosu programı olacak Açılış ekranında en fazla 20 sayısını girebileceğiniz bir sınır sorusu sorulacak mesela 11 girildiğinde aşağıdaki gibi ekran görüntüsü elde edilecektir. E harfine basarsanız aynı işlem tekrar edecektir. Başka bir harfe bastığınızda program sonlanacak