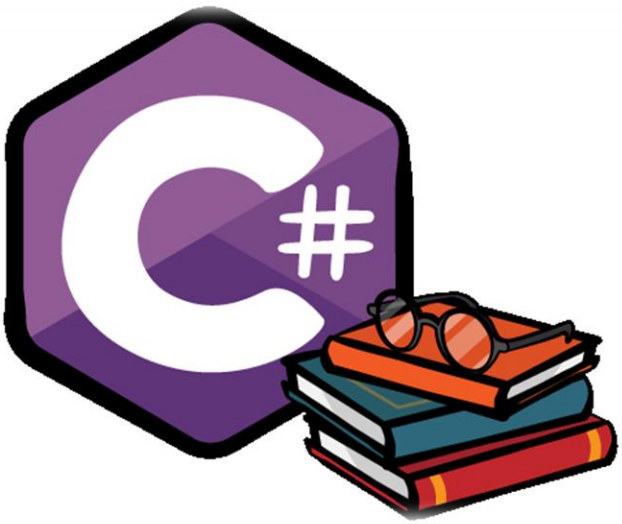


CONSOLE EKRANI-4

CONSOLE EKRANI

- ✓ ***Döngüler***

KAZANIMLAR

- ✓ *Döngü deyimlerinin hangi durumlarda kullanıldığını bilir*
- ✓ *Döngü deyimlerinin kullanım mantığını bilir*
- ✓ *Döngü Deyimlerıı ile ilgili problemleri çözer.*

DÖNGÜ DEYİMLERİ

Döngüler bir program içerisinde belirli işlerin defalarca yapılmasını sağlayan komut bloklarıdır. Sonsuz döngüler yapılabildiği gibi belirli kriterler sağlanana kadar devam eden döngüler de yapılabilir.

4 tip döngü vardır. Bunlar:

- ✓ *for döngüleri*
- ✓ *while döngüleri*
- ✓ *do while döngüleri*
- ✓ *foreach döngüleri' dir.*

Belirlenen başlangıç değerinden itibaren belirtilen koşul sağlanana kadar içine yazıldığı kod parçasını ardı ardına çalıştıran bir döngü çeşididir.

For döngüsünün kullanımı şu şekildedir;

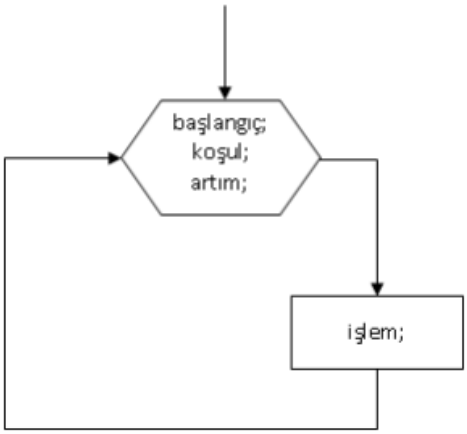
Kullanımı:

```
for(başlangıç;koşul;artım)
{
yapılacak işler;
}
```

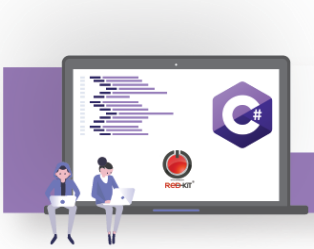
Başlangıç, *döngü kontrol değişkeni olarak da ifade edilebilir. Döngü içerisinde bir sayaç görevini görür.*

Koşul, *döngünün ne kadar çalışacağını denetleyen mekanizmadır. Koşul sağlanıyorken döngü çalışmaya devam eder. Koşul sağlanmadığında ise döngü durur. Koşulda genellikle başlangıç değerinin durumu denetlenir.*

Artım, *başlangıç değerinin döngünün her adımda artma ya da azaltma miktarını belirler. Eğer başlangıç değeri hiç değişmez ise sonsuz döngü oluşur.*



Akış diyagramlarıyla for döngüsünün gösterimi de şu şekildedir.



Şimdi basit bir örnekle for döngüsünün çalışmasını inceleyelim.

Örnek 1: *1'den 10'a kadar olan sayıları ekrana yazdırınız.*

```
byte i;

for(i=1;i<=10;i++)

{

Console.WriteLine(i);

}
```

Yukarıdaki kodu incelediğimizde;

- ✓ *Döngü kontrol değişkenimiz olan i'ye 1 değerini atayarak başlangıç değerimizi,*
- ✓ *Döngümüzün ne zamana kadar döneceğini belirlediğimiz koşulumuzu i<=10 ifadesini,*
- ✓ *i++ ile de i değerimizi döngümüzün her dönüşünde 1 arttıracağımızı belirliyoruz.*

Döngü her seferinde koşul kısmını kontrol eder ve buradaki koşul false(yanlış) olana kadar küme parantezleri ({ }) ile sınırlandırılan kod bloğunu çalıştırmaya devam edecektir. For terimiyle döngü kurarken başlangıç değerimiz herhangi bir tam sayı olabileceği gibi char türünde bir değişkende olabilir.

For döngüsüyle sonsuz bir döngü oluşturulmak istenirse şu şekilde kodlanması gerekir;

```
for(;;)

{

//.....

}
```

Dikkat : *Bu şekilde bir sonsuz döngüyü bilgisayarınızda çalıştırdığınız zaman uygulamanız sonsuza kadar devam eder.*

For döngüleri ileriye doğru sayabildiği gibi geriye dönük sayma işlemlerinde de kullanılırlar.

Örnek 2: *10'dan 0'a geriye doğru sayan ve sayıları ekrana yazdıran programı yazdırınız.*

```
int i;

for (i=10;i>=0;i--)

Console.WriteLine(i);
```

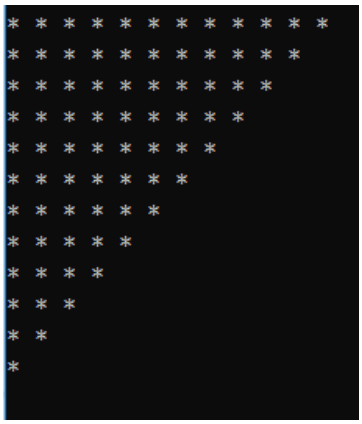
Örnek 3: ***0'dan klavyeden girilen sayıya kadar olan sayıların toplamını ekrana yazdıran programı yazınız.***

```
int bitis,i,toplam;
Console.Write("Bir sayı giriniz:");
bitis = Convert.ToInt32(Console.ReadLine());
toplam = 0;
for (i = 0; i <= bitis; i++)
{
toplam = toplam + i;
}
Console.WriteLine("Toplam={0}", toplam );
```

İpucu: *Kod satırlarımızda yer alan "toplam=toplam+i;" kod bloğumuzu "toplam+=i;" diye de ifade edebiliriz. İkisi toplam değerinin içerisine i değerini ekler.*

Şimdiye kadar gördüğümüz örneklerde for döngüsünü hep tek başına kullandık. Aynı koşul kontrol mekanizmalarında olduğu gibi döngüler de iç-içe kullanılabilirler. İç-içe kullanılacak döngü sayılarında herhangi bir kısıtlama söz konusu değildir. İstediğimiz kadar sayıda döngüyü iç-içe kullanabiliriz Sıradaki örneklerimizde de iç içe for döngüsü nasıl kullanılır buna göz atalım.

Örnek 4: ***Yandaki ekran çıktısını verecek programın kodunu yazalım***



```
for (int i = 0; i <5 ; i++)
    {
   for (int j = 0; j <=i ; j++)
   {
   Console.Write("*");
   }
   Console.Write("\n");
   }
   Console.ReadLine();
```

```
}
```

While Döngüsü

While döngüsü bir koşul sağlanıyorken dönmeye devam eder. Koşul yanlış (false) sonucunu verdiği zaman ise sonlandırılır. Genel yazım şekli şöyledir.

Kullanımı:

```
while(koşul)

{

yapılacak işler;

}
```

Örnek 2: 0'dan 20'ye kadar olan çift sayıları ekrana yazdırınız

```
class Program
{
    static void Main(string[] args)
    {
        int i = 0;
        while (i <= 20)
        {
            Console.WriteLine(i);
            i = i + 2;
        }
    }
}
}
```

Örnek 5: Sayı tahmin oyunu

```
static void Main(string[] args)
    {
        int tahmin=0, tutulan, sayac=0;
        Random rasgele = new Random();
        tutulan = rasgele.Next(1, 100);
        while (tahmin!=tutulan)
        {
            sayac++;
            Console.WriteLine("Sayı giriniz");
            tahmin = Convert.ToInt32(Console.ReadLine());
            if(tahmin &gt; tutulan)
            {
                Console.WriteLine("Sayıyı küçült");

            }
            else if (tahmin &lt; tutulan)
            { Console.WriteLine("Sayıyı büyüt");
        }
```

```
        }
            Console.WriteLine("Tebrikler.");
            Console.WriteLine("{0}.hakkınızda bildiniz", sayac);
            Console.ReadKey();
        }
    }
```

Örnek6 : Kullanıcı tarafından girilen bir sayının kaç basamaklı olduğunu bulup ekrana yazdıran program

```
int sayi = Convert.ToInt32(Console.ReadLine());
int basamak = 0;
while (sayi > 0)
{
    basamak++;
    sayi = sayi / 10;
}
Console.WriteLine("Girdiğiniz sayı " + basamak.ToString() + "basamaklıdır.");
```

Örnek7: Kullanıcıdan devam etmek istiyor musunuz sorusuna alınan yanıta göre sayı alıp alınan sayıların toplamını bulan programı yazınız. (Evet "E" veya Hayır "H")

```
using System;

namespace ConsoleApp3
{
    class Program
    {
        static void Main(string[] args)
        {
            int toplam = 0, sayac = 1;
            char cevap = 'e';

            while (cevap == 'e')
            {
                Console.WriteLine("{0}. Sayıyı girin.", sayac);
                toplam += Convert.ToInt32(Console.ReadLine());

                Console.WriteLine("Devam istiyor musu?(e,h)");
                cevap = Convert.ToChar(Console.ReadLine());
                sayac++;
            }
            Console.WriteLine("Sonuç={0}", toplam);
            Console.ReadLine();
        }
    }
}
```

Do…While Döngüsü

For ve while döngülerinde döngü bloklarının koşul sağlanmadığı takdirde hiç çalıştırılmama ihtimali vardır. Ancak döngünün en az bir kere çalıştırılması istenilen durumlarda do-while döngüleri kullanılırlar.

Do-While döngülerinde koşul döngü içerisindeki işlemler bir kez gerçekleştirildikten sonra kontrol edilir. Koşul doğru olduğu müddetçe de döngü içerisindeki işlemler tekrarlanmayı sürdürür.

Genel yazım şekli şöyledir.

Kullanımı:

```
do
{
yapılacak işler;
}
while(koşul);
```

Örnek 8: 1'den 20'ye kadar olan tek sayıları ekrana yazdırınız.

```
int i=1;
do{
Console.WriteLine(i);
i = i + 2;
} while (i < 20);
```

Yukarıdaki örnek kod incelendiğinde koşulun do-while döngüsünün en sonunda kontrol edildiğini görebilirsiniz.

Örnek 8: Sayı tahmin oyunu

Bilgisayar hafızasında rastgele bir sayı tutucak,kullanıcıda o sayıyı tahmin edicek.Kullanıcı her sayıyı girdiğinde tahmin edilen sayıya yakınlığını yazıcak.

```
int sayi, tahmin, defa=1;
        Console.WriteLine("Sence ben kaç yaşındayım ?");
        tahmin = Convert.ToInt32(Console.ReadLine());
        Random rnd = new Random();
        sayi = rnd.Next(0, 5);
        do
        {
           defa++;
```

```
        if (sayi > tahmin)
        {
            Console.WriteLine("daha büyük sayı tahmin et");
            tahmin= Convert.ToInt32(Console.ReadLine());
        }
        else if (tahmin > sayi)
        {
            Console.WriteLine("daha küçük bir sayı tahmin et");
            tahmin = Convert.ToInt32(Console.ReadLine());
        }
    } while (sayi != tahmin);
    Console.WriteLine("tebrikler doğru tahmin :)");
    Console.WriteLine(defa + ".defada bildin");
    Console.ReadKey();
```
