1. for DÖNGÜSÜ

Hemen hemen bütün programlama dillerinde döngü denilince akla ilk olarak for döngüsü gelir. Güncel programlama dillerinin neredeyse tamamında for döngüsünün yazım ve çalışma şekli ise birebir aynıdır.

Kullanım şekli şöyledir.

```
for (Başlangıç Değeri Atama ; Koşul ; Değer Değiştirme)
{
     Tekrarlanacak Kodlar
}
```

for döngüsünde { } küme parantezleri yazılmazsa sadece döngüden sonra yazılan ilk kod tekrarlanır. Ancak küme parantezleri kullanılırsa parantezlerin içinde kalan bütün kodlar tekrarlanır.

for döngüsünde bir ***döngü değişkeni*** kullanılır. Döngü değişkeni olarak genellikle "i" adı kullanılsa da bu isim zorunlu değildir. Herhangi bir değişken adı da kullanılabilir.

for döngüsünün çalışma akışı şu şekildedir:

1. Döngü değişkenine bir başlangıç değeri atanır.
2. Koşul kontrol edilir.
    - Koşul doğru ise tekrarlanacak kodlar çalıştırılır.
    - Koşul doğru değilse döngüden çıkılır (döngü sona erdirilir).
3. Döngü değişkeninin değeri değiştirilir ve 2. adıma dönülür.

Yukarıda da gösterildiği üzere, for döngüsünde ( ) parantezleri arasında iki adet noktalı virgül vardır.

- İlk noktalı virgülden önce döngü değişkenine başlangıç değeri atanır. Buraya yazılan kodlar döngü çalıştırılmaya başlanacağı zaman ***sadece bir kez*** işletilir.
- İlk noktalı virgülden sonra bir koşul yer alır. Bu koşul doğru ise döngü tekrarlanacak, doğru değilse döngü sona erdirilecektir. Buraya yazılan koşul, { } küme parantezlerinin arasındaki kodlar işletilmeden önce ***her zaman*** kontrol edilir.

- İkinci noktalı virgülden sonra ise döngü değişkeninin değeri değiştirilir. Yani duruma göre arttırılır veya azaltılır. Buraya yazılan kodlar { } küme parantezleri arasındaki kodlar *işletildikten sonra* ***her zaman*** çalıştırılır.

Örnek

```
for (var i = 1; i <= 3; i++)
{
    Console.WriteLine(i);
}
```

Yukarıdaki for döngüsünde başlangıç değeri atama kısmında şu kod yer almaktadır:

```
var i = 1
```

Burada döngü değişkeni olarak tamsayı tipinde i adında bir değişken tanımlanmış ve başlangıç değeri olarak 1 değeri atanmıştır. Bu komut for döngüsü işletilmeden hemen önce ***sadece bir kez*** çalıştırılır.

Yukarıdaki döngüde aşağıdaki koşul yazılmıştır.

```
i <= 3
```

Döngüde tekrarlanması istenen kodlar kaçıncı kez işletilecek olursa olsun (ilk işletilişinde bile) önce bu koşul kontrol edilir, koşul doğru ise çalıştırılır. Yani her seferinde i değişkeninin değerinin 3'ten küçük veya 3'e eşit olup olmadığı kontrol edilecektir.

Yukarıdaki döngüde değer değiştirme kısmında şu kod yer almaktadır:

```
i++
```

Döngüde tekrarlanması istenen kodlar işletildikten sonra, her seferinde bu kısımdaki kod çalıştırılır. Bu döngüde, her seferinde i değişkeninin değeri 1 arttırılacaktır.

Son olarak, tekrarlanması istenen kod olarak aşağıdaki kod yer almaktadır:

```
Console.WriteLine(i);
```

Yani eğer koşul doğru ise döngünün her adımında "i" değişkeninin değeri ekrana yazdırılacaktır. Tabi her seferinde "i" değişkeninin değeri farklı olacağı için, döngünün her adımında ekrana farklı bir değer yazdırılacaktır.

Yukarıdaki döngünün çalışması şu şekilde olacaktır:

- i değişkenine 1 değeri atanır.
- Koşul kontrol edilir. i=1 olduğu için i <= 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana i değişkeninin değeri olan **1** yazdırılır.
- i++ kodları çalıştırılır, yani i değişkeninin değeri 1 arttırılır, böylece 2 olur.
- Koşul kontrol edilir. i=2 olduğu için i <= 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana i değişkeninin değeri olan **2** yazdırılır.
- i++ kodları çalıştırılır, yani i değişkeninin değeri 1 arttırılır, böylece 3 olur.
- Koşul kontrol edilir. i=3 olduğu için i <= 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana i değişkeninin değeri olan **3** yazdırılır.
- i++ kodları çalıştırılır, yani i değişkeninin değeri 1 arttırılır, böylece 4 olur.
- Koşul kontrol edilir. i=4 olduğu için i <= 3 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Böylelikle ekran çıktısı aşağıdaki gibi olur.

```
1
2
3
```

Örnek

```
for (var i = 1; i <= 3; i++)
{
    Console.WriteLine("C#");
}
```

Fark edeceğiniz üzere, yukarıdaki for döngüsünde bir önceki örneğe göre farklı olan tek şey, ekrana i değişkeninin değerinin yerine "C#" yazılıyor olmasıdır.

Bu döngünün işletilişi de aşağıdaki gibi olacaktır:

- i değişkenine 1 değeri atanır.
- Koşul kontrol edilir. i=1 olduğu için i <= 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani "**C#**" yazdırılır.
- i++ kodları çalıştırılır, yani i değişkeninin değeri 1 arttırılır, böylece 2 olur.
- Koşul kontrol edilir. i=2 olduğu için i <= 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani "**C#**" yazdırılır.
- i++ kodları çalıştırılır, yani i değişkeninin değeri 1 arttırılır, böylece 3 olur.
- Koşul kontrol edilir. i=3 olduğu için i <= 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani "**C#**" yazdırılır.
- i++ kodları çalıştırılır, yani i değişkeninin değeri 1 arttırılır, böylece 4 olur.
- Koşul kontrol edilir. i=4 olduğu için i <= 3 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Böylelikle ekran çıktısı aşağıdaki gibi olacaktır.

```
C#
C#
C#
```

Örnek

```
for (var sayi = 3; sayi >= 1; sayi--)
{
    Console.WriteLine(sayi);
}
```

Fark edeceğiniz üzere, yukarıdaki for döngüsünde döngü değişkeni olarak i değil sayi kullanılmıştır. Yine fark edeceğiniz üzere, sayi değişkeninin değeri 3'ten başlayıp birer birer azalmaktadır.

Bu döngünün işletilişi aşağıdaki gibi olacaktır:

- sayi değişkenine 3 değeri atanır.
- Koşul kontrol edilir. sayi=3 olduğu için sayi >= 1 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **3** yazdırılır.
- sayi-- kodları çalıştırılır, yani sayi değişkeninin değeri 1 azaltılır, böylece 2 olur.
- Koşul kontrol edilir. sayi=2 olduğu için sayi >= 1 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **2** yazdırılır.
- sayi-- kodları çalıştırılır, yani sayi değişkeninin değeri 1 azaltılır, böylece 1 olur.
- Koşul kontrol edilir. sayi=1 olduğu için sayi >= 1 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **1** yazdırılır.
- sayi-- kodları çalıştırılır, yani sayi değişkeninin değeri 1 azaltılır, böylece 0 olur.
- Koşul kontrol edilir. sayi=0 olduğu için sayi >= 1 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Böylelikle ekran çıktısı aşağıdaki gibi olacaktır.

```
3
2
1
```

Örnek

```
for (var sayi = 0; sayi < 10; sayi+=2)
{
    Console.WriteLine(sayi);
}
```

Yukarıdaki for döngüsünde, sayi değişkeninin değeri 0'dan başlayıp ikişer ikişer artmaktadır.

Bu döngünün işletilişi aşağıdaki gibi olacaktır:

- sayi değişkenine 0 değeri atanır.
- Koşul kontrol edilir. sayi=0 olduğu için sayi < 10 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **0** yazdırılır.
- sayi+=2 kodları çalıştırılır, yani sayi değişkeninin değeri 2 arttırılır ve 2 olur.
- Koşul kontrol edilir. sayi=2 olduğu için sayi < 10 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **2** yazdırılır.
- sayi+=2 kodları çalıştırılır, yani sayi değişkeninin değeri 2 arttırılır ve 4 olur.
- Koşul kontrol edilir. sayi=4 olduğu için sayi < 10 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **4** yazdırılır.
- sayi+=2 kodları çalıştırılır, yani sayi değişkeninin değeri 2 arttırılır ve 6 olur.
- Koşul kontrol edilir. sayi=6 olduğu için sayi < 10 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **6** yazdırılır.
- sayi+=2 kodları çalıştırılır, yani sayi değişkeninin değeri 2 arttırılır ve 8 olur.
- Koşul kontrol edilir. sayi=8 olduğu için sayi < 10 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana **8** yazdırılır.
- sayi+=2 kodları çalıştırılır, yani sayi değişkeninin değeri 2 arttırılır ve 10 olur.
- Koşul kontrol edilir. sayi=10 olduğu için sayi < 10 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Böylelikle ekran çıktısı aşağıdaki gibi olacaktır.

```
0
2
4
6
8
```

Örnek

```
for (var sayi = 10; sayi < 10; sayi++)
{
    Console.WriteLine(sayi);
}
```

Yukarıdaki döngünün işletilişi aşağıdaki gibi olacaktır:

- sayi değişkenine 10 değeri atanır.
- Koşul kontrol edilir. sayi=10 olduğu için sayi < 10 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Bu döngüde Console.WriteLine komutu hiç çalıştırılmadığı için ***ekrana hiçbir şey yazdırılmayacaktır***.

Örnek

```
for (var sayi = 5; sayi < 10; sayi++)
{
    Console.WriteLine(sayi);
}
```

Yukarıdaki döngüde sayi değişkeninin başlangıç değeri 5'tir. Bu değer döngünün her adımı işletildikten sonra bir arttırılacaktır. Yani döngü değişkenin değerleri 5, 6, 7, … şeklinde gidecektir. Koşul olarak sayi < 10 yazıldığı için, bu koşulu sağlayan en büyük değer 9 olacaktır. Yani döngü değişkeninin değerleri 5, 6, 7, 8 ve 9 iken Console.WriteLine komutu çalıştırılacak, böylelikle de ekran çıktısı aşağıdaki gibi olacaktır.

```
5
6
7
8
9
```

Örnek

```
for (var sayi = 5; sayi > 0; sayi++)
{
    Console.WriteLine(sayi);
}
```

Yukarıdaki döngüde sayi değişkeninin başlangıç değeri 5'tir. Bu değer döngünün her adımı işletildikten sonra bir arttırılacaktır. Yani döngü değişkenin değerleri 5, 6, 7, … şeklinde gidecektir.

Koşul olarak sayi > 0 yazıldığı için sayı değişkeninin değeri ne kadar artarsa artsın bu koşul sonsuza kadar sağlanıyor olacaktır. Bu yüzden de döngü asla sona ermeyecektir.

Bu tür döngülere ***sonsuz döngü*** adı verilir. Bir programda sonsuz döngüye girildiğinde, o döngü hiçbir zaman bitmeyeceği için program orada dönüp duracak, yani kilitlenip kalacak ve yanıt vermeyecektir.

Örnek

```
for (var i = 0; i < 4; i++)
{
    Console.WriteLine(i * 2);
}
```

Fark edeceğiniz üzere, yukarıdaki döngüde ekrana i değişkeninin kendi değeri değil, iki ile çarpıldıktan sonra bulunan sonuç yazdırılmaktadır.

Burada her ne kadar i değişkeninin değerleri 0, 1, 2, 3 iken koşul sağlanıyor olsa da, ekrana bu değerlerin kendisi değil, bu değerlerin iki katı yazdırılacaktır.

Ekran çıktısı aşağıdaki gibi olacaktır.

```
0
2
4
6
```

Örnek

```
for (var i = 3; i <= 5; i++)
{
    Console.WriteLine(i + 10);
}
```

Yukarıdaki döngüde i değişkeni 3'ten başlayıp birer birer artmaktadır. Koşul i <= 5 olduğu için, i değişkeninin bu koşulu sağlayan değerleri 3, 4, 5 olacaktır.

Ancak bir önceki örnekte olduğu gibi, ekrana bu değerlerin kendisi yazdırılmayacak, bu değerlerin 10 fazlası yazdırılacaktır.

Bu yüzden de ekran çıktısı aşağıdaki gibi olacaktır.

```
13
14
15
```

Örnek

```
for (var i = 1; i <= 2; i++)
    Console.WriteLine(i);
```

Yukarıdaki for döngüsünde { } küme parantezleri kullanılmamış olsa bile, ilk komut Console.WriteLine olduğu için bu komut tekrarlanacak ve ekran çıktısı aşağıdaki gibi olacaktır.

```
1
2
```

Örnek

```
for (var i = 1; i <= 2; i++)
{
    Console.WriteLine(i);
    Console.WriteLine("C#");
}
```

Yukarıdaki for döngüsünde { } küme parantezleri arasında iki farklı komut yer almaktadır. Yani döngünün her tekrarlanışında bu iki komut peş peşe çalıştırılacaktır.

Döngü değişkeninin değerleri 1 ve 2 iken koşul sağlanacağı için, ekran çıktısı aşağıdaki gibi olacaktır.

```
1
C#
2
C#
```

Örnek

```
for (var i = 1; i <= 2; i++)
    Console.WriteLine(i);
    Console.WriteLine("C#");
```

Yukarıdaki döngüde küme parantezleri kullanılmadığı için, sadece ilk Console.WriteLine komutu for döngüsüne dahildir. İkinci yazılan Console.WriteLine komutunun döngü ile hiçbir ilgisi yoktur. Bu komut, döngü sona erdirildikten sonra çalıştırılacaktır.

Bu yüzden, yukarıdaki döngü sanki aşağıdaki gibi yazılmış şekilde çalıştırılacaktır.

```
for (var i = 1; i <= 2; i++)
{
    Console.WriteLine(i);
}
Console.WriteLine("C#");
```

Yukarıdaki kodlar çalıştırıldığında önce döngü çalıştırılacak, ekranda 1 ve 2 yazacak, döngü sonlandıktan sonra da ekrana “C#” yazdırılacaktır.

Bu programın ekran çıktısı aşağıdaki gibi olacaktır.

```
1
2
C#
```

2. while DÖNGÜSÜ

while döngüsü for döngüsüne göre daha sade bir döngüdür. Bu döngüde parantez içine başlangıç değeri atama ve değer değiştirme kısımları yazılmaz, sadece koşul yazılır.

Kullanılışı aşağıdaki gibidir.

```
while (koşul)
{
    Tekrarlanacak Kodlar
}
```

while döngüsünün işletilmesi şu şekildedir:

1. Koşul kontrol edilir.
    - Koşul doğru ise tekrarlanacak kodlar çalıştırılır.
    - Koşul doğru değilse döngüden çıkılır (döngü sona erdirilir).
2. Tekrarlanacak kodlar çalıştırıldıktan sonra 1. adıma geri dönülür.

Örnek

```
var sayi = 0;
while (sayi < 3)
{
    Console.WriteLine(sayi);
    sayi++;
}
```

Yukarıdaki kodların çalışması şu şekilde olacaktır.

- sayi değişkenine 0 değeri atanır.
- Koşul kontrol edilir. sayi=0 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **0** yazdırılır. Sonra da sayi değişkeninin değeri 1 arttırılır ve 1 olur.
- Koşul kontrol edilir. sayi=1 olduğu için sayi < 3 koşulu ***doğrudur***.

- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **1** yazdırılır. Sonra da sayi değişkeninin değeri 1 arttırılır ve 2 olur.
- Koşul kontrol edilir. sayi=2 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **2** yazdırılır. Sonra da sayi değişkeninin değeri 1 arttırılır ve 3 olur.
- Koşul kontrol edilir. sayi=3 olduğu için sayi < 3 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Ekran çıktısı aşağıdaki gibi olacaktır.

```
0
1
2
```

Örnek

```
var sayi = 0;
while (sayi < 3)
{
    sayi++;
    Console.WriteLine(sayi);
}
```

Fark edileceği üzere, yukarıdaki örneğin bir önceki örnekten tek farkı sayi++ komutunun Console.WriteLine komutundan sonra değil önce çalıştırılmasıdır.

Yukarıdaki kodların çalışması şu şekilde olacaktır.

- sayi değişkenine 0 değeri atanır.
- Koşul kontrol edilir. sayi=0 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani sayi değişkeninin değeri 1 arttırılır ve 1 olur. Sonra da ekrana sayi değişkeninin değeri olan **1** yazdırılır.
- Koşul kontrol edilir. sayi=1 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani sayi değişkeninin değeri 1 arttırılır ve 2 olur. Sonra da ekrana sayi değişkeninin değeri olan **2** yazdırılır.

- Koşul kontrol edilir. sayi=2 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani sayi değişkeninin değeri 1 arttırılır ve 3 olur. Sonra da ekrana sayi değişkeninin değeri olan **3** yazdırılır.
- Koşul kontrol edilir. sayi=3 olduğu için sayi < 3 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Ekran çıktısı aşağıdaki gibi olacaktır.

```
1
2
3
```

Örnek

```
var sayi = 0;
while (sayi < 3)
{
    sayi++;
    Console.WriteLine(sayi);
    sayi++;
                                                    }
```

Yukarıdaki kodların çalışması şu şekilde olacaktır.

- sayi değişkenine 0 değeri atanır.
- Koşul kontrol edilir. sayi=0 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani sayi değişkeninin değeri 1 arttırılır ve 1 olur. Sonra da ekrana sayi değişkeninin değeri olan **1** yazdırılır. Sonra sayi değişkeninin değer 1 arttırılır ve 2 olur.
- Koşul kontrol edilir. sayi=2 olduğu için sayi < 3 koşulu ***doğrudur***.
- Tekrarlanacak kodlar çalıştırılır. Yani sayi değişkeninin değeri 1 arttırılır ve 3 olur. Sonra da ekrana sayi değişkeninin değeri olan **3** yazdırılır. Sonra sayi değişkeninin değer 1 arttırılır ve 4 olur.
- Koşul kontrol edilir. sayi=4 olduğu için sayi < 3 koşulu ***yanlıştır***.

- Koşul yanlış olduğu için döngü sonlandırılır.

Ekran çıktısı aşağıdaki gibi olacaktır.

```
1
3
```

Örnek

```
var sayi = 0;
while (sayi > 5)
{
    Console.WriteLine(sayi);
}
```

Yukarıdaki kodların çalışması şu şekilde olacaktır.

- sayi değişkenine 0 değeri atanır.
- Koşul kontrol edilir. sayi=0 olduğu için sayi > 5 koşulu ***yanlıştır***.
- Koşul yanlış olduğu için döngü sonlandırılır.

Bu program da Console.WriteLine komutu hiç çalıştırılmadığı için ekrana hiçbir şey yazdırılmayacaktır.

Örnek

```
var sayi = 0;
while (sayi < 5)
{
    Console.WriteLine(sayi);
}
```

Yukarıdaki kodlar çalıştırıldığında while döngüsünün çalışması hiç bitmez, yani sonsuz döngüye girilir. Çünkü sayi değişkeninin değeri 0 olarak atandıktan sonra hiç değiştirilmemektedir. Bu değişkenin değeri değişmediği için de sayi < 5 koşulu hep sağlanacaktır.

Örnek

```
var sayi = 0;
while (sayi < 5)
{
    Console.Write(sayi);
    sayi++;
}
```

Bu örnekte Console.Write komutunun kullanıldığı fark edilecektir.

Hatırlanacağı üzere, Console.WriteLine komutu ekrana yazdıktan sonra alt satıra geçerken, Console.Write komutu geçmemektedir. Böylelikle bir sonraki yazılan çıktı öncekinin *yanına* yazdırılmaktadır. Bu sebeple, yukarıdaki programın ekran çıktısında sayılar yan yana bitişik olarak ekrana yazdırılacaktır.

Yukarıdaki programın ekran çıktısı aşağıdaki gibi olacaktır.

```
01234
```

3. do-while DÖNGÜSÜ

do-while döngüsü while döngüsü ile çok benzerdir. while döngüsünde koşul tekrarlanacak kodlardan önce yazılırken, do-while döngüsünde koşul tekrarlanacak kodlardan sonra yazılır. Bu yüzden de, koşul tekrarlanacak kodlar işletilmeden önce değil, işletildikten sonra kontrol edilir.

Kullanılışı aşağıdaki gibidir.

```
do
{
    Tekrarlanacak kodlar
} while (koşul);
```

do-while döngüsünün işletilmesi şu şekildedir:

1. Tekrarlanacak kodlar çalıştırılır.
2. Koşul kontrol edilir.
    - Koşul doğru ise 1. adıma geri dönülür.
    - Koşul doğru değilse döngüden çıkılır (döngü sona erdirilir).

Dikkat edilirse, döngü ilk başlatıldığında tekrarlanacak kodlar koşul kontrol edilmeden çalıştırılır. Yani yazılan koşul baştan hatalı olsa bile tekrarlanacak kodlar en az bir kere çalıştırılacaktır.

Örnek

```
var sayi = 0;
do
{
    Console.Write(sayi);
    sayi++;
} while (sayi < 4);
```

Yukarıdaki kodların işletilişi aşağıdaki gibi olacaktır.

- sayi değişkenine 0 değeri atanır.

- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **0** yazdırılır (alt satıra geçilmez), sonra sayi değişkeninin değeri 1 arttırılarak 1 olur.
- Koşul kontrol edilir. sayi=1 olduğu için sayi < 4 koşulu **doğrudur**.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **1** yazdırılır (alt satıra geçilmez), sonra sayi değişkeninin değeri 1 arttırılarak 2 olur.
- Koşul kontrol edilir. sayi=2 olduğu için sayi < 4 koşulu **doğrudur**.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **2** yazdırılır (alt satıra geçilmez), sonra sayi değişkeninin değeri 1 arttırılarak 3 olur.
- Koşul kontrol edilir. sayi=3 olduğu için sayi < 4 koşulu **doğrudur**.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **3** yazdırılır (alt satıra geçilmez), sonra sayi değişkeninin değeri 1 arttırılarak 4 olur.
- Koşul kontrol edilir. sayi=4 olduğu için sayi < 4 koşulu **yanlıştır**.
- Koşul yanlış olduğu için döngü sonlandırılır.

Ekran çıktısı aşağıdaki gibi olacaktır.

```
0123
```

Örnek

```
var sayi = 0;
do
{
    Console.Write(sayi);
    sayi++;
} while (sayi > 100);
```

Yukarıdaki kodların işletilişi aşağıdaki gibi olacaktır.

- sayi değişkenine 0 değeri atanır.
- Tekrarlanacak kodlar çalıştırılır. Yani ekrana sayi değişkeninin değeri olan **0** yazdırılır (alt satıra geçilmez), sonra sayi değişkeninin değeri 1 arttırılarak 1 olur.
- Koşul kontrol edilir. sayi=4 olduğu için sayi > 100 koşulu **yanlıştır**.

- Koşul yanlış olduğu için döngü sonlandırılır.

Dikkat edilirse, koşul daha en başından yanlış olmasına rağmen tekrarlanacak kodlar bir kez çalıştırılmıştır.

Ekran çıktısı aşağıdaki gibi olacaktır.

```
0
```