



Akış Kontrol Mekanizmaları

Koşul İfadeleri

-  Programlar içersinde bazı durumlarda programın akışının değişmesi ya da farklı işlemlerin yapılması gerekebilir. Bazen de seçimlerin yapılması ya da belirli işlemlerin belli sayıda tekrar edilmesi gerekebilir.
-  Bu gibi işlemleri gerçekleştirmek için koşul ifadeleri ve döngü yapıları kullanılır.



C# dilinde koşula bağlı olarak işlemler gerçekleştirmek için iki farklı deyim kullanılabilir:

- if deyimi
- switch deyimi

If Deyimi



Program akış kontrol deyimlerinin başında gelir. Koşula göre değişik işlemlerin yapılmasını sağlar. Basit kullanımı:

- if (Koşul)
 Koşul_doğru_ise_yapılacaklar
else
 Koşul_yanlış_ise_yapılacaklar

```
using System;

class if_1
{
    static void Main()
    {
        int x, y;

        Console.Write("1. Sayıyı Girin:");
        x = Convert.ToInt32(Console.ReadLine());

        Console.Write("2. Sayıyı Girin:");
        y = Convert.ToInt32(Console.ReadLine());

        if (x > y)
            Console.WriteLine("\n1. Sayı Büyük");
        else
            Console.WriteLine("\n2. Sayı Büyük");

    }
}
```

- ✓ if deyimi else olmadan da kullanılabilir.
- ✓ if ya da else ifadelerinden sonra birden fazla işlem yapılmak isteniyorsa blok kullanılabilir.
- ✓ Programın akışı birden fazla koşula göre değişecek ise “else if” ifadeleri eklenebilir.
- ✓ Birden fazla if deyimi iç içe kullanılarak karmaşık koşulların kontrol edilmesi sağlanabilir.



if deyiminin en ayrıntılı kullanımı:

```
if (koşul)
    { ifadeler }
else if (koşul)
    { ifadeler }
else if (koşul)
    { ifadeler }
.....
else
    { ifadeler }
```

```

using System;

class if_2
{
    static void Main()
    {
        int not;

        Console.Write("Notunuzu Girin (0-100) :");
        not = Convert.ToInt32(Console.ReadLine());

        if (not >= 0 && not < 40)
            Console.WriteLine("FF");
        else if (not >= 40 && not < 50)
            Console.WriteLine("FD");
        else if (not >= 50 && not < 55)
            Console.WriteLine("DD");
        else if (not >= 55 && not < 60)
            Console.WriteLine("DC");
        else if (not >= 60 && not < 70)
            Console.WriteLine("CC");
        else if (not >= 70 && not < 80)
            Console.WriteLine("CB");
        else if (not >= 80 && not < 85)
            Console.WriteLine("BB");
        else if (not >= 85 && not < 90)
            Console.WriteLine("BA");
        else if (not >= 90 && not <= 100)
            Console.WriteLine("AA");
        else
            Console.WriteLine("Geçersiz Not Girdiniz...");
    }
}

```



```

using System;
class if_3
{
    static void Main()
    {
        string secim;
        int s1, s2;

        Console.Write("1.Sayıyı Girin:");
        s1 = Convert.ToInt32(Console.ReadLine());
        Console.Write("2.Sayıyı Girin:");
        s2 = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("\nİşleminizi Seçin:\n=====");
        Console.WriteLine("Toplama (t)");
        Console.WriteLine("Çıkarma (ç)");
        Console.WriteLine("Çarpma (x)");
        Console.WriteLine("Bölme (b)");
        Console.WriteLine("Mod Alma (m)");
        secim = Console.ReadLine();

        if (secim == "t")
            Console.WriteLine("Toplam = {0}", s1 + s2);
        else if (secim == "ç")
            Console.WriteLine("Fark = {0}", s1 - s2);
        else if (secim == "x")
            Console.WriteLine("Çarpım = {0}", s1 * s2);
        else if (secim == "b")
        {
            if (s2==0)
                Console.WriteLine("Bölen 0 olamaz...");
            else
                Console.WriteLine("Bölüm = {0}",(float)s1 / (float)s2);
        }
        else if (secim == "m")
        {
            if (s2 == 0)
                Console.WriteLine("Bölen 0 olamaz...");
            else
                Console.WriteLine("Mod = {0}", s1 % s2);
        }
        else
            Console.WriteLine("Yanlış İşlem Seçtiniz...");
    }
}

```

Switch Deyimi



Bir ifadenin alabileceği değişik değerlere göre işlemler gerçekleştirmek için kullanılan deyimdir. Karmaşık if ifadelerini daha sade bir şekilde ifade etmeyi de sağlayabilir. Kullanımı:

```
switch (ifade)
{
    case sabit1 :
        deyimler;
        break;
    case sabit2 :
        deyimler;
        break;
    .....
    default :
        deyimler;
        break;
}
```



Bir switch yapısında **break**, **case**, **switch** ve **goto** anahtar sözcükleri kullanılır.



Switch bloğu bir ifadenin üreteceği değerlere göre programın akışını dallandırır.

```
using System;

class switch_1
{
    static void Main()
    {
        Console.Write("Bir sayı girin:");
        int sayi = Convert.ToInt32(Console.ReadLine());

        switch (sayi)
        {
            case 1:
            case 2:
                Console.WriteLine("Case 1 ve 2");
                break;
            case 3:
                Console.WriteLine("Case 3");
            case 4:
                Console.WriteLine("Case 4");
                goto case 5;
            case 5:
                Console.WriteLine("Case 5");
                break;
            default:
                Console.WriteLine("Default");
                break;
        }
    }
}
```



switch ifadesi kullanırken dikkat edilecek bazı kurallar vardır:

- case sözcüğünden sonra gelen ifadeler sabit olmak zorundadır.
- **case ifadeleri tamsayı, karakter ya da string sabitler olabilir.**
- default ve case ifadeleri istenilen sırada yazılabilir.
- Aynı switch bloğu içersinden birden fazla aynı case ifadesi bulunamaz.
- default kullanmak zorunlu değildir.
- Akışı bir case ifadesinden bir başka case ifadesine yönlendirmek için mutlaka goto anahtar sözcüğü kullanılır.

Döngüler



Program içerisinde belirli işleri tekrar tekrar yapılmasını sağlayan kod bloklarıdır. Döngüler sonsuz olabileceği gibi belli bir koşul ile kontrol etmek de mümkündür. C# dilinde dört tip döngü yapısı bulunur:

- for
- while
- do while
- foreach

for Döngüsü



En çok kullanılan döngü deyimlerinden biridir. Kullanımı:

- for (ifade1; ifade2; ifade3) işlem;

Ya da

- for (ifade1; ifade2; ifade3)

```
{
```

```
    işlemler...
```

```
}
```



for döngü tanımlamasında iki “;” ile ayrılmış üç ifade bulunur. Bunlardan bazıları boş olabilir fakat mutlaka “;” işaretleri kullanılmalıdır.



İlk ifade bir defaya mahsus olmak üzere çalıştırılır. Genelde döngü değişkeninin tanımlanması ya da ilk değer ataması için kullanılır.



İkinci ifade ise döngünün kontrol edildiği kısımdır. Buradaki ifade “true” değer ürettiği sürece döngü devam eder.



Son ifade ise genelde döngü değişkeninin değerinin değiştirildiği kısımdır.


```
using System;

class for_1
{
    static void Main()
    {
        Console.Write("Bir sayı girin:");
        int n = Convert.ToInt32(Console.ReadLine());
        int toplam = 0;

        for (int i = 1; i <= n; i++)
            toplam = toplam + i;

        Console.WriteLine("1\'den {0}\'e kadar olan
            sayıların toplamı {1}\'dir.", n, toplam);
    }
}
```

```
using System;

class for_2
{
    static void Main()
    {
        string s;

        for (s = Console.ReadLine(); s != "Çıkış";
            s = Console.ReadLine())

            Console.WriteLine(s);
    }
}
```

```
using System;

class for_3
{
    static void Main()
    {
        int i=0, a, n;

        Console.Write("Bir Sayı Girin :");
        n = Convert.ToInt32(Console.ReadLine());

        Console.Write("Artım Miktarı  :");
        a = Convert.ToInt32(Console.ReadLine());

        for (; i < n; )
        {
            Console.Write("{0} ", i);
            i += a;
        }
    }
}
```

```

using System;

class for_4
{
    static void Main()
    {
        int k, t, toplam, n1, n2;

        Console.Write("Aralık başlangıcı :");
        n1 = Convert.ToInt32(Console.ReadLine());

        Console.Write("Aralık Sonu      :");
        n2 = Convert.ToInt32(Console.ReadLine());

        if (n1 <= n2)
        {
            for (k = n1; k <= n2; k++)
            {
                toplam = 0;
                for (t = 1; t <= k; t++) { if (k % t == 0) toplam = toplam + t; }
                if (toplam == k + 1) Console.WriteLine(k);
            }
        }
        else
        {
            Console.WriteLine("Geçerli bir aralık giriniz...");
        }
    }
}

```

```
using System;

class for_5
{
    static void Main()
    {
        int sayi;

        Console.Write("Bir tamsayı giriniz :");
        sayi = Convert.ToInt32(Console.ReadLine());

        for (int bit = 32; bit >= 1; bit--)
        {
            Console.Write("{0}", (sayi >> bit - 1) & 1);
        }
        Console.WriteLine();
    }
}
```

while Döngüsü



Bir başka döngü deyimidir. Belirtilen koşul doğru olduğu sürece çalışmasına devam eder.

- while (koşul)

işlem;

veya

- while (koşul)

{

işlemler...

}

```
using System;

class while_1
{
    static void Main()
    {
        int i=0;

        while (i < 1000)
        {
            i += 5;
            Console.Write("{0,5}",i);
            if (i % 50 == 0) Console.WriteLine();
        }
    }
}
```

do-while Döngüsü



for ve while döngülerinde koşul sağlanmadığı durumlarda döngü bloğu hiç çalıştırılmaz. Bazı durumlarda ise döngünün en az bir kez çalıştırılması gerekebilir. Bu durumda do while kullanılabilir.

- do
 işlem
 while (koşul);
ya da
- do
 {
 işlemler...
 } while (koşul);


```

using System;

class do_while_1
{
    static void Main()
    {
        int secim;

        do
        {
            Console.WriteLine("İşlemler\n=====");
            Console.WriteLine("1 - Toplama");
            Console.WriteLine("2 - Çıkarma");
            Console.WriteLine("3 - Çarpma");
            Console.WriteLine("4 - Bölme");
            Console.WriteLine("0 - Çıkış\n");
            Console.Write("İşleminizi Seçin:");
            secim = Convert.ToInt32(Console.ReadLine());
            switch (secim)
            {
                case 1: Console.WriteLine("Toplama işlemi seçildi."); break;
                case 2: Console.WriteLine("Çıkarma işlemi seçildi."); break;
                case 3: Console.WriteLine("Çarpma işlemi seçildi."); break;
                case 4: Console.WriteLine("Bölme işlemi seçildi."); break;
                case 0: Console.WriteLine("Çıkış seçildi."); break;
                default: Console.WriteLine("Yanlış seçim."); break;
            }
        }
        while(secim != 0);
    }
}

```

foreach Döngüsü

- ✓ Farklı bir döngü yapısıdır. Koleksiyon tabanlı nesnelere içersinde adım adım dolaşılmasını sağlar.
- ✓ Bu döngü elemanlara tek tek ulaşım sağlar fakat ulaşılan elemanlar sadece okunabilir özelliğe sahiptir.

Break ve Continue Anahtar Sözcükleri

- ✓ Çalışan bir döngüden “break” sözcüğü kullanılarak çıkılabilir. Program akışı döngüden sonraki satırlardan devam eder. break sözcüğü sadece döngü ve switch ifadelerinde kullanılabilir.
- ✓ continue sözcüğü ise döngünün bir sonraki tekrarına geçilmesini sağlar.

```

using System;

class break_continue_1
{
    static void Main()
    {
        int sayi, t=0;

        while (true)
        {
            Console.Write("Negatif bir sayı girin:");
            sayi = Convert.ToInt32(Console.ReadLine());
            if (sayi == 0) break;
            if (sayi > 0)
            {
                Console.WriteLine("Pozitif sayı girdiniz,
tekrar deneyin...");
                continue;
            }
            t = t + sayi;
        }
        Console.WriteLine("Toplam :{0}", t);
    }
}

```

Goto Anahtar Sözcüğü



Programın etiket ile belirlenmiş herhangi bir kısmına atlamak için kullanılır. Nesneye yönelik programlamaya uygun bir yapı değildir. Switch ifadesindeki kullanımı dışında mümkün olduğunca kullanılmaktan kaçınılmalıdır.

- Etiket:

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

Goto Etiket:

Return Sözcüğü



return Anahtar Sözcüğü

- Metotların herhangi bir anda sonlandırılması için kullanılır. Metot sonlandırıldıktan sonra programın akışı metodu çağıran fonksiyondan devam eder.