1. KARAR YAPILARI

Bir veya birden fazla komutun çalıştırılıp çalıştırılmayacağına karar veren yapılara programlama dilinde karar yapıları adı verilir.

Karar yapıları döngülerle karıştırılmamalıdır. Döngülerde bir koşula bağlı olarak devamlı başa dönüp aynı işlemler tekrarlanır. Karar yapıları ise koşul doğru olduğunda bir işlemi *sadece bir kez* yapar.

C#'ta iki farklı karar yapısı vardır.

- if karar yapısı
- switch karar yapısı

1.1. if

Verilen koşula bağlı olarak bir veya birden fazla komutun çalıştırılıp çalıştırılmayacağına karar verir.

Kullanımı aşağıdaki gibidir.

```
if (koşul) komut;
```

veya

```
if (koşul) { komut; }
```

veya

```
if (koşul)
{
    komut1;
    komut2;
    ...
}
```

Koşul doğru olduğunda sadece bir komut çalıştırılacaksa, bu komut küme parantezleri { } içine alınmadan yazılabilir. Ancak koşul doğru olduğunda birden fazla komut çalıştırılacaksa o zaman küme parantezleri zorunludur.

C#'ta küme parantezleri bir blok belirtir. if karar yapısında koşuldan sonra küme parantezleri kullanılmışsa;

- Koşul doğru ise küme parantezleri ile belirtilen kod bloğundaki bütün kodlar çalıştırılır.
- Koşul yanlış ise küme parantezleri ile belirtilen kod bloğundaki bütün kodlar çalıştırılmadan geçilir.

Örnek

```
if (5 > 2)
{
    Console.WriteLine("Merhaba");
}
```

Yukarıdaki örnekte koşul olarak "5 > 2" yazılmıştır. 5 sayısı 2'den büyük olduğu için bu koşul doğrudur ve ekrana "Merhaba" yazdırılacaktır.

Örnek

```
if (2 > 5)
{
    Console.WriteLine("Merhaba");
}
```

Yukarıdaki örnekteki koşul yanlış olduğu için ekrana "Merhaba" yazdıran komut çalıştırılmadan atlanacaktır.

Örnek

```
if (10 > 9)
{
    Console.WriteLine("Merhaba,");
    Console.WriteLine("Ben C# öğreniyorum!");
}
```

Yukarıdaki örnekte koşul doğru olduğu için ekrana hem “Merhaba,” hem de “Ben C# öğreniyorum!” yazdırılacaktır.

Örnek

```
if (10 > 10)
{
    Console.WriteLine("Merhaba,");
    Console.WriteLine("Ben C# öğreniyorum!");
}
```

Yukarıdaki örnekteki koşul yanlış olduğu için her iki komut da çalıştırılmadan geçilecek, ekrana hiçbir şey yazdırılmayacaktır.

Örnek

```
if (10 > 10)
    Console.WriteLine("Merhaba,");
    Console.WriteLine("Ben C# öğreniyorum!");
```

Yukarıdaki örnekte koşul yanlış olduğu için ekrana “Merhaba” yazdıran komut çalıştırılmadan geçilecek, ama sonrasında ekrana “Ben C# öğreniyorum!” yazdırılacaktır.

Ekran görüntüsü aşağıdaki gibi olacaktır.

```
Ben C# öğreniyorum!
```

Bunu sebebi şudur: Her ne kadar satır girintileri aynı olduğu için bu iki komut bir blok oluşturuyormuş gibi bir göz yanılgısı oluşsa da, küme parantezleri konulmadığı için if karar yapısına sadece ilk komut bağlıdır. En alt satırdaki komut if karar yapısından tamamen bağımsızdır. O yüzden koşul doğru olmadığında sadece ilk komut atlanır, ikinci komut ise çalıştırılır.

Koşullarda kullanılan karşılaştırma operatörleri

Koşullar yazılırken belirli karşılaştırma operatörleri kullanılır.

C#’ta kullanılan karşılaştırma operatörleri aşağıda listelenmiştir.

- > (büyüktür)
- >= (büyük veya eşittir)
- < (küçüktür)
- <= (küçük veya eşittir)
- == (eşittir)
- != (farklıdır)

Koşullarda değişken kullanımı

Koşul yazılırken değişkenler de kullanılabilir. Koşulun doğru olup olmadığı değişkenin o anki değerlerine bağlı olarak hesaplanacak ve işlemin yapılıp yapılmayacağına karar verilecektir.

Örnek

```
var a = 9;
var b = 7;
if (a >= b)
     Console.WriteLine(a);
```

Yukarıdaki örnekte a değişkeninin değeri (9) b değişkeninin değerinden (7) büyük olduğu için koşul sağlanacak ve ekrana a değişkeninin değeri yazdırılacaktır.

```
9
```

Koşullarda aritmetik işlem kullanımı

Koşullar yazılırken aritmetik işlemler kullanılmışsa, önce aritmetik işlemler yapılır, koşulun doğru olup olmadığı aritmetik işlemin sonucuna göre belirlenir.

Örnek

```
var a = 9;
var b = 7;
if (a + b == 16)
     Console.WriteLine(b);
```

Yukarıdaki örnekte a + b işlemi yapılıp sonuç 16 olarak bulunur. Daha sonra sonucun 16'ya eşit olup olmadığı kontrol edilir ve eşit olduğu için koşul sağlanır. Koşul sağlandığı için de ekrana b değişkeninin değerini yazdıran komut çalıştırılır.

```
7
```

Örnek

```
var a = 9;
var b = 7;
if (a + b == 10 + 5 + 1)
    Console.WriteLine(b);
```

Yukarıdaki örnekte a + b işlemi yapılıp sonuç 16 olarak bulunur. Karşılaştırma ifadesinin sağ tarafındaki 10 + 5 + 1 işlemi yapılıp onun sonucu da 16 olarak bulunur. Daha sonra bulunan her iki sonucun eşit olup olmadığı kontrol edilir ve eşit olduğu için koşul sağlanır. Koşul sağlandığı için de ekrana b değişkeninin değerini yazdıran komut çalıştırılır.

```
7
```

Koşul yerine false veya true kullanımı

Bütün karşılaştırma işlemlerinin sonucu true veya false sonucunu üretir. Örneğin 1 > 2 karşılaştırma işleminin sonucu false'tur. Ya da 3 >= 0 karşılaştırma işleminin sonucu true'dur. Bu sebeple, koşul olarak bir karşılaştırma ifadesi yazmak yerine direkt olarak false veya true yazılırsa bu ifade hatalı olmaz.

Örnek

```
if (true) Console.WriteLine("Merhaba");
```

Yukarıdaki örnekte koşul yerine doğru (true) değeri direkt olarak yazıldığı için ekrana "Merhaba" yazılır.

```
Merhaba
```

Örnek

```
if (false) Console.WriteLine("Merhaba");
```

Yukarıdaki örnekte koşul yerine yanlış (false) değeri direkt olarak yazıldığı için ekrana "Merhaba" *yazılmaz*.

Koşul yerine bool tipli değişken kullanımı

Koşul olarak direkt false veya true yazmak hiç de anlamlı değildir. Çünkü if ifadesinde koşul olarak false yazıldığında o if ifadesindeki komutlar asla çalıştırılmaz. Dolayısıyla, hiç çalışmayacak bu if ifadesini yazmak da anlamsız olacaktır.

Koşul olarak direkt true ya da false değerini yazmak yerine, true/false değerlerini alan bool tipli bir değişkenin adı da yazılabilir.

Örnek

```
var buyukMu = 5 > 4;
if (buyukMu) Console.WriteLine("Merhaba");
```

Yukarıdaki örnekte buyukMu isimli bool tipli değişkene 5 > 4 karşılaştırma ifadesinin sonucu (true) değer olarak atanacaktır. if ifadesindeki koşul kontrol edilirken ise buyukMu değişkeninin değeri kontrol edilecektir. Bu değer true olduğu için koşul doğru kabul edilecek, dolayısıyla da ekrana "Merhaba" yazdırılacaktır.

1.2. if – else

Yukarıda da bahsedildiği gibi, if karar yapısına yazılan koşul yanlış ise ona bağlı olan komut ya da komut bloğu işletilmeden atlanır. Ancak bazı durumlarda koşul doğru olduğunda belirli bazı komutları, yanlış olduğunda ise başka bazı komutları çalıştırmak gerekebilir. Bu tür durumlarda if – else karar yapısı kullanılır.

if – else karar yapısında, koşul doğru ise if bloğundaki kodlar, koşul yanlış ise else bloğundaki kodlar işletilir. Yani bu karar yapısında koşula bağlı olarak iki kod bloğundan biri seçilerek sadece o bloktaki kodlar işletilir.

Kelime anlamı olarak else, "aksi takdirde" anlamına gelmektedir. Yani *yukarıdaki koşul doğru değilse* anlamı taşımaktadır.

Örnek

```
var ortalama = 60;
if (ortalama >= 40)
     Console.WriteLine("Geçtiniz.");
else
     Console.WriteLine("Kaldınız.");
```

Yukarıdaki örnekte if koşulu doğrudur. Bu sebeple ekrana "Geçtiniz" yazacak ve else kısmındaki kodlar atlanacaktır.

```
var ortalama = 30;
if (ortalama >= 40)
     Console.WriteLine("Geçtiniz.");
else
     Console.WriteLine("Kaldınız.");
```

Yukarıdaki örnekte görüldüğü gibi eğer ortalama 40'ın altında ise ekranda sadece "Kaldınız" yazacaktır.

Örnek

```
var isim = "Erdal";
if (isim == "Erdal")
    Console.WriteLine("Merhaba Erdal Bey.");
else
    Console.WriteLine("Hoş geldiniz.");
```

Yukarıdaki programda koşul doğru olduğu için ekran çıktısı aşağıdaki gibi olacaktır.

```
Merhaba Erdal Bey.
```

Hatırlatma

if – else yapısında dikkat edilmesi gereken bir husus vardır. if ifadesindeki koşul doğru olduğunda çalıştırılacak kodlardan *hemen sonra* else yazılmalıdır. Eğer araya başka komutlar yazılırsa program hata verecektir.

```
if (4 > 10)
{
    Console.WriteLine("Merhaba.");
}
Console.WriteLine("Nasılsınız?"); // HATA!!!
else
{
    Console.WriteLine("Selamlar.");
}
```

Yukarıdaki program hata verecektir çünkü if bloğu ile else bloğu arasına başka komutlar girmiştir.

1.3. if – else if

Bazı durumlarda, "yukarıdaki koşul doğru değilse" demek yerine, "yukarıdaki koşul doğru değil ama bu koşul doğruysa" demek gerekebilir. Bu durumlarda if – else if yapısı kullanılır.

Örnek

```
var sicaklik = 20;
if (sicaklik >= 30) Console.WriteLine("Hava çok sıcak.");
else if (sicaklik >= 20) Console.WriteLine("Hava sıcak.");
```

Yukarıdaki programda if koşulu doğru değildir. Yani sıcaklık 30'dan büyük veya eşit değildir. Bu yüzden ekranda "Hava çok sıcak." yazmayacaktır. Ancak devamında else if ifadesi vardır. Yani "yukarıdaki koşul doğru değilse ama bu koşul doğruysa" denilmektedir. Yukarıdaki koşul doğru doğru olmadığı ve else if koşulu doğru olduğu için ekranda "Hava sıcak." yazacaktır.

Eğer her iki koşul da doğru olmasaydı, ekranda hiçbir şey yazmayacaktı.

Örnek

```
var sicaklik = 10;
if (sicaklik >= 30) Console.WriteLine("Hava çok sıcak.");
else if (sicaklik >= 20) Console.WriteLine("Hava sıcak.");
```

Yukarıdaki örnekte ekrana hiçbir şey yazılmayacaktır. Çünkü if koşulu da else if koşulu da doğru değildir.

Gerektiği durumlarda birden fazla (sonsuz sayıda) else if koşulu yazılabilir.

Örnek

```
var sicaklik = 7;
if (sicaklik >= 30) Console.WriteLine("Hava çok sıcak");
else if (sicaklik >= 20) Console.WriteLine("Hava sıcak");
else if (sicaklik >= 10) Console.WriteLine("Hava soğuk");
else if (sicaklik >= 0) Console.WriteLine("Hava çok soğuk");
```

Yukarıdaki örnekte, 1. satırdaki if koşulu (sicaklik >= 30) hatalı olduğu için ekrana “Hava çok sıcak” yazmadan atlanacaktır.

Yukarıdaki koşul doğru olmadığı için ikinci satırdaki koşul (sicaklik >= 20) kontrol edilecek, bu da doğru olmadığı için ekrana “Hava sıcak” yazmadan atlanacaktır.

Bir üst satırdaki koşul da doğru olmadığı için bu sefer üçüncü satırdaki koşul (sicaklik >= 10) kontrol edilecek, bu da doğru olmadığı için ekrana “Hava soğuk” yazmadan atlanacaktır.

Sonra, yine bir üst satırdaki koşul da doğru olmadığı için bu sefer dördüncü satırdaki koşul (sicaklik >= 0) kontrol edilecek, bu koşul doğru olduğu için ekrana “Hava çok soğuk” yazılacaktır.

Peş peşe else if koşulları yazılmışsa, bu koşullardan herhangi birinin doğru olması halinde, o koşula bağlı olan komutlar işletilir ve ondan sonra yazılan koşulların hiç biri kontrol edilmez. Çünkü hatırlanacağı üzere else if “yukarıdaki koşul doğru değilse ama bu koşul doğruysa” demektir. Yukarıdaki koşul doğru olduğuna göre, alttaki else if koşullarının hiç birinin hükmü kalmamış olur ve hepsi atlanır.

Örnek

```
var sicaklik = 22;
if (sicaklik >= 30) Console.WriteLine("Hava çok sıcak");
else if (sicaklik >= 20) Console.WriteLine("Hava sıcak");
else if (sicaklik >= 10) Console.WriteLine("Hava soğuk");
else if (sicaklik >= 0) Console.WriteLine("Hava çok soğuk");
```

Yukarıdaki örnekte sıcaklık 22 derece olduğu için, ilk satırdaki koşul sağlanmayacak ancak ikinci satırdaki koşul sağlanacaktır. Bu yüzden ekranda "Hava sıcak" yazacak ve bundan sonraki hiçbir else if koşulu kontrol edilmeden geçilecektir. Yani yukarıdaki örneğin ekran çıktısı aşağıdaki gibi olacaktır.

```
Hava sıcak
```

1.4. if – else if – else

Bazı durumlarda, if – else if koşullarının hiçbiri sağlanmıyorsa belirli bir iş yaptırmak istenebilir. Bu durumda en sona bir else yazılır.

Örnek

```
var sicaklik = -8;
if (sicaklik >= 30) Console.WriteLine("Hava çok sıcak");
else if (sicaklik >= 20) Console.WriteLine("Hava sıcak");
else if (sicaklik >= 10) Console.WriteLine("Hava soğuk");
else if (sicaklik >= 0) Console.WriteLine("Hava çok soğuk");
else Console.WriteLine("Hava aşırı soğuk");
```

Yukarıdaki örnekte if ve else if koşulların hiçbiri sağlanmamaktadır. Bu durumda en sonda else ifadesinden sonra yazılmış olan komut çalıştırılır. Bu programın ekran çıktısı aşağıdaki gibi olacaktır.

```
Hava aşırı soğuk
```

1.5. Birden fazla koşulun aynı anda kullanımı

İki ya da daha fazla sayıdaki koşulun bir arada kullanılması programcıların sıklıkla ihtiyaç duyduğu şeylerden biridir.

Bir if ifadesinde parantez içinde birden fazla koşul birlikte yazılabilir. Bu durumda koşulların arasında && veya || operatörlerinden biri koyulur.

- ✓ && operatörü

  Bu operatör VE anlamı taşımaktadır. İki koşul arasında eğer && operatörü varsa komutların işletilebilmesi için her *bütün koşulların doğru olması* gerekmektedir.

- ✓ || operatörü

  Bu operatör VEYA anlamı taşımaktadır. İki koşul arasında eğer || operatörü varsa *koşullardan en az birinin doğru olması* yeterlidir.

Örnek

```
var sicaklik = 25;
if (sicaklik >= 20 && sicaklik < 30)
    Console.WriteLine("Hava çok güzel");
```

Yukarıdaki örnekte iki koşul birlikte kullanılmış ve aralarına && operatörü konulmuştur. Bu sebeple, yalnızca her iki koşulun birden doğru olması halinde ekranda "Hava çok güzel" yazacaktır. Yukarıdaki örnekte iki koşul da sağlandığı için ekran çıktısı aşağıdaki gibi olacaktır.

```
Hava çok güzel
```

Örnek

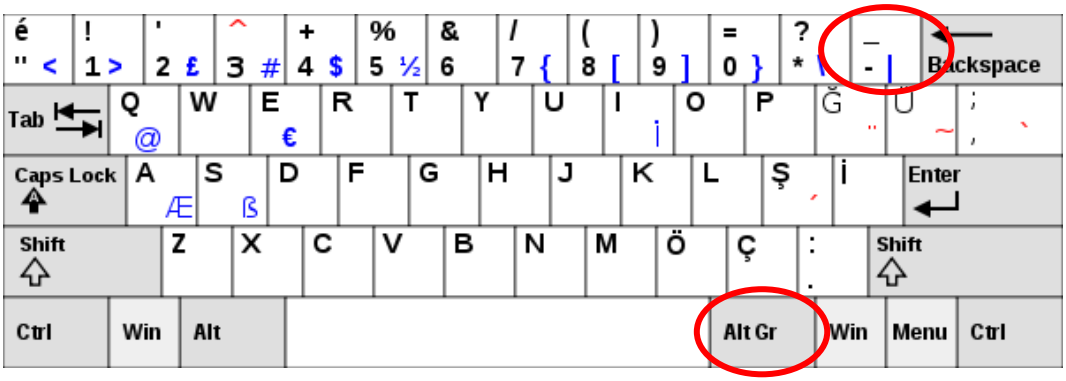
```
Var ortalama = 200;
if (ortalama < 0 || ortalama > 100)
    Console.WriteLine("Ortalama geçersiz");
```

Yukarıdaki örnekte yine iki koşul birlikte kullanılmış ve aralarına || operatörü konulmuştur. Bu sebeple, bu iki koşuldan en az birinin doğru olması halinde ekranda "Ortalama geçersiz" yazacaktır. Yukarıdaki örnekte ilk koşul (ortalama < 0) doğru olmamasına rağmen ikinci koşul (ortalama > 100) doğru olduğu için ekran çıktısı aşağıdaki gibi olacaktır.

```
Ortalama geçersiz.
```

Not

| işareti bazı klavyelerde tuşların üzerinde sembol olarak gösterilmeyebilir. Bu işareti çıkarmak için boşluk tuşunun yanındaki Alt Gr tuşu ile Backspace tuşunun solundaki – tuşuna aynı anda basmak gerekmektedir.



Uyarı

&& ve || operatörlerinin her iki tarafına da koşul yazılmalıdır.

Örnek

Türkçede olduğu gibi “sayı 1 veya 2 ise” şeklinde düşünerek aşağıdaki gibi bir ifade yazılırsa program hata verecektir.

```
if (sayi == 1 || 2)
{
    Console.WriteLine("C#");  // HATA!
}
```

Yukarıdaki örnekte || operatörünün sol tarafında doğru/yanlış (true/false) şeklindeki bir karşılaştırma ifadesi olmasına rağmen sağ tarafında sadece bir tamsayı yer almaktadır. Hâlbuki || ifadesinin sağ tarafında da doğru/yanlış değeri üreten bir koşul yazması gerekirdi. Bu sebeple yukarıdaki kodlar çalışmayacaktır.

Koşulların arasına && veya || operatörlerini konularak ikiden fazla koşul da bir arada kullanılabilir.

Örnek

```
if (vize == 0 && quiz1 == 0 && quiz2 == 0 && final == 0)
{
    Console.WriteLine("Hiçbir sınava girmemişsiniz");
}
```

Yukarıdaki örnekteki kodlar çalıştırıldığında dört koşul da doğru ise yani bütün sınavlardan 0 almışsa ekranda "Hiçbir sınava girmemişsiniz" yazacaktır.

1.6. switch

Bir değişkenin değerinin ne olduğuna bakılarak birden fazla seçenek arasından hangi kodların çalıştırılacağına karar vermek için switch karar yapısı kullanılır.

switch karar yapısında bir koşul ya da karşılaştırma ifadesi yazılmaz. Sadece değişkenin aldığı değer kontrol edilir ve hangi kodların çalıştırılacağına karar verilir.

Kullanımı aşağıdaki gibidir.

```
switch (değişken)
{
    case değer1:
          komutlar;
          break;
    case değer2:
          komutlar;
          break;
    case değer3:
          komutlar;
          break;
    ...
    default:
          komutlar;
          break;
}
```

Eğer değişkenin değeri case ile belirtilen değerlerden hiçbirine eşit değilse bu durumda default kısmında yazılan komut çalıştırılacaktır.

switch karar yapısında istenildiği kadar sayıda case yazılabilir.

Örnek

```
var gun = 6;
switch (gun)
{
     case 1:
           Console.WriteLine("Pazartesi");
           break;
     case 2:
           Console.WriteLine("Salı");
           break;
     case 3:
           Console.WriteLine("Çarşamba");
           break;
     case 4:
           Console.WriteLine("Perşembe");
           break;
     case 5:
           Console.WriteLine("Cuma");
           break;
     case 6:
           Console.WriteLine("Cumartesi");
           break;
     case 7:
           Console.WriteLine("Pazar");
           break;
     default:
           Console.WriteLine("Hatalı gün");
           break;
}
```

Yukarıdaki örnekte gun değişkeninin değeri 6 olduğu için “case 6” yazan kısımdaki kodlar çalıştırılacak ve ekrana “Cumartesi” yazdırılacaktır. Diğer kodların hiç biri çalıştırılmayacaktır.

Eğer gun değişkeninin değeri 3 olsaydı ekranda “Çarşamba”, 5 olsaydı “Cuma” yazacaktı.

Eğer gun değişkeninin değeri 1, 2, 3, 4, 5, 6 ve 7 dışında herhangi bir değer olsaydı (örneğin 10) bu durumda default kısmındaki kodlar çalıştırılacak ve ekranda “Hatalı gün” yazacaktı.

Bazı hallerde case’ler içine hiçbir komut yazılmadan peş peşe sıralanabilir.

Örnek

```
var gun = 6;
switch (gun)
{
     case 1:
     case 2:
     case 3:
     case 4:
     case 5:
           Console.WriteLine("Hafta içi");
           break;
     case 6:
     case 7:
           Console.WriteLine("Hafta sonu");
           break;
     default:
           Console.WriteLine("Hatalı gün");
           break;
}
```

Yukarıdaki örnekte gün 6 olduğu için “case 6” kısmına atlanacaktır. Ancak “case 6” ile “case 7” birleşik yazıldığı için “case 7” altında yazılan komutlar işletilecektir. Dolayısıyla da ekranda “Hafta sonu” yazacaktır. Eğer gün 1 ile 5 arasında herhangi bir değer olsaydı, ekranda “Hafta içi” yazacaktı.

default kullanımı zorunlu değildir. Bir switch ifadesinde default yazılmadığı takdirde, case’lerden hiç biri geçerli değilse hiçbir işlem yapılmadan switch ifadesi sonlandırılır.

Örnek

```
var sayi = 0;
switch (sayi)
{
     case 1:
           Console.WriteLine("Bir");
           break;
     case 2:
           Console.WriteLine("İki");
           break;
}
```

Yukarıdaki program çalıştırıldığında ekrana hiçbir şey yazdırılmayacaktır. Çünkü sayi değişkenin değeri case'lerin hiçbirine uymamaktadır.

1.7. switch ifadelerini if ifadelerine dönüştürme

Her if ifadesi switch ifadesine dönüştürülemez. Çünkü if koşullarında büyük mü ya da küçük mü gibi karşılaştırma ifadeleri yer alabilir.

Örnek

```
if (sayi < 0) Console.WriteLine("Sayı negatif");
else Console.WriteLine("Sayı pozitif");
```

Yukarıdaki if yapısı switch yapısına dönüştürülemez, çünkü switch'teki gibi bir değişken değerinin başka bir değere eşit olup olmadığı kontrol edilmiyor, küçük olup olmadığı kontrol ediliyor. Oysa ki switch ifadeleri yalnızca eşitlik kontrolü yapabilmektedir.

Her if ifadesi switch ifadesine dönüştürülemese de, her switch ifadesi if ifadesine dönüştürülebilir.

Çünkü switch ifadesine yapılan şey değişken değerinin case değerleriyle eşit olup olmadığının bulunması ve ona göre ilgili kodların işletilmesidir. Bu işlem if yapısı ile de yapılabilir.

Örnek

```
switch (sayi)
{
     case 1:
           Console.WriteLine("Bir");
           break;
     case 2:
           Console.WriteLine("İki");
           break;
     case 3:
           Console.WriteLine("Üç");
           break;
     default:
           Console.WriteLine("Hatalı giriş");
           break;
}
```

Yukarıdaki switch ifadesi aşağıdaki if ifadesine dönüştürülebilir. Bu iki ifade birebir aynı işlevi görmektedir.

```
if (sayi == 1)
{
     Console.WriteLine("Bir");
}
else if (sayi == 2)
{
     Console.WriteLine("İki");
}
else if (sayi == 3)
{
     Console.WriteLine("Üç");
}
else
{
     Console.WriteLine("Hatalı giriş");
}
```

Örnek

```
switch (sayi)
{
     case 1:
           Console.WriteLine("Bir");
           break;
     case 2:
           Console.WriteLine("İki");
           break;
}
```

Yukarıdaki switch ifadesi aşağıdaki if ifadesi birebir aynı işlevi görmektedir.

```
if (sayi == 1)
{
     Console.WriteLine("Bir");
}
else if (sayi == 2)
{
     Console.WriteLine("İki");
}
```

Örnek

```
switch (sayi)
{
     case 1:
     case 2:
           Console.WriteLine("Bir veya iki");
           break;
     default:
           Console.WriteLine("Hatalı giriş");
           break;
}
```

Yukarıdaki switch ifadesi aşağıdaki if ifadesine dönüştürülebilir.

```
if (sayi == 1 || sayi == 2)
    Console.WriteLine("Bir veya iki");
else
    Console.WriteLine("Hatalı giriş");
```