



## Decimal Veri Tipi

Bazı programlama dillerinde değişkenlerin kullanımı veri tiplerine bağlı olmayabilir. Basic dili böyle idi. Bazı dillerde ise türlerin tanımlanması ve doğru karar verilmiş olması gerekir. Bir türün diğer bir türe dönüştürülmesi bizzat programcı tarafından açık olarak yapılmalıdır. Bu yüzden, Basic programlama dilinde veri tipleri üzerindeki kontrollerin artması en önemli gelişmelerden birisi olmuştur.

Programcılarının değişkenler için veri tiplerini seçmesi gerekir. Tamsayı olmayan sayılar için C# programlama dili floating point ve decimal olmak üzere iki ayrı alternatif sunar. Decimal veri tipi diğer dillerde olmayan farklı bir veridir.

Gerçek sayılar için decimal veya floating point türünde değişken kullanmamız gerekir. Bilimsel veya mühendislik uygulamaları için tanımlanacak değişkenler için floating point veri tipi kullanılır. Floating point veri tipi çok büyük ve çok küçük sayılar için kullanılır. Bazı durumlarda, floating point türü kullanıldığında çok küçük hatalı sonuçlar oluşabilir. Decimal veri tipi kesin sonuçlar verdiği için floating point kadar esnek değildir.

Decimal veri tipi, System.Decimal sınıfından türetilir. Kullanımı:

```
decimal cebimdekiPara;
```

Eğer cebinizde bozuk para yok ise decimal tipinde değişken kullanabilirsiniz.

```
cebimdekiPara = 70;
```

Eğer bozuk paranız da varsa, şu şekilde kullanmak isterseniz:

```
cebimdekiPara = 70.25; //Problem olabilir.
```

Karışıklığa mahal vermemek için sonuna M veya m eklemeniz gerekir.

```
cebimdekiPara = 70.25m; //Düzgün çalışacaktır
```

Aşağıdaki programı inceleyelim.

```
//-----  
//biletHesabi.cs  
//-----  
using System;  
class biletHesabi  
{  
    static void Main()  
    {  
        const decimal yuksekFiyat = 12.50m, dusukFiyat = 8.25m;  
  
        Console.Write("Pahalı bilet fiyatını giriniz: ");  
        int pahaliBiletSayisi = Int32.Parse(Console.ReadLine());
```

```
Console.Write("Ucuz bilet fiyatını giriniz: ");
int ucuzBiletSayisi = Int32.Parse(Console.ReadLine());

decimal toplamFiyat = pahaliBiletSayisi * yuksekFiyat +
ucuzBiletSayisi * dusukFiyat;
Console.WriteLine("Toplam fiyat:" + toplamFiyat + " YTL");

}
}
```

Gördüğümüz gibi, decimal sayıları ve tamsayıları aynı ifade içinde kullanabiliyoruz. C# herhangi bir tamsayı değişkeni decimal türüne çevirebilir. Decimal değerler ve tamsayı değerler arasındaki hesaplama sonucu decimal değerdir. Fakat C#, decimal tipleri tamsayı tipine dönüştürmez.

```
decimal toplamPara = 54.25m;
int para = toplamPara; //Derleme Hatası
```

C# derleyicisi bu şekildeki çevirmeyi önlemektedir. Decimal veri tipinden tamsayı veri tipine çevirmede kayıp oluşabilir.

casting işlemi yaparak decimal tipte bir sayıyı kayıplarla birlikte tamsayı türüne dönüştürmek mümkündür.

```
int para = (int)toplamPara;
```

Bir bilme işlemin sonucunun decimal olabilmesi için işlem yapılacak sayılardan en az birisinin decimal olması gerekir, ya da decimal türüne dönüştürülmesi gerekir.

```
int a = 25, b = 10;
decimal c = a/b; //sonuc 2 çıkacaktır.
```

Çünkü bir tamsayının diğer bir tamsayıya bölünmesi sonucu bölüm yine bir tamsayıdır. İşleme giren sayılardan birisini açık dönüştürme ile decimal sayıya dönüştürelim.

```
int a = 25, b = 10;
decimal c = (decimal) a/b; //sonuc 2.5
```

Çıkan sonucu yuvarlamak isteyebiliriz. Bunu da kaç basamak yuvarlayacağımızı belirterek yapabiliriz.

```
decimal x = 25.77m;
decimal y = .05m;
decimal sonuc = x * y; // sonuc 1.2885
decimal yuvarlanmis = decimal.Round(sonuc, 2); //yuvarlanmis = 1.29 olur.
```