



# Bölüm 3

## Atama ve Girdi/Çıktı Komutları

Atama Komutu

Operatörler

İsim Sabitleri

Veri Tipi Dönüşümü

Çıktı Fonksiyonu - printf()

Girdi Fonksiyonu - scanf()

Matematik Kütüphanesi

# Atama Komutu

Değişkenlerin yani bellek hücrelerinin içine veri saklamak için kullanılan yöntemlerden birisi **atama komutudur**.

*değişken = ifade;*

## Komut

## Bellek Görüntüsü

<code>int x;</code>	<div><div>x</div><div>?</div></div>
<code>x=5;</code>	<div><div>x</div><div>5</div></div>

# Atama Komutu

## Tanımlama Komutu

```
double y;  
char ch;
```

## Bellek Görüntüsü

ch	y
?	?

## Atama Komutu

```
y=10.23;  
ch='M';
```

## Bellek Görüntüsü

ch	y
M	10.23

# Atama Komutu

int k;  
double z;

<u>Atama komutu</u>	<u>Değişken değeri</u>
k=14;	k tamsayı değişkenidir, atama operatörünün sağ tarafındaki 14 tamsayı sabiti, k'ya atanır.
k=3.7;	k tamsayı değişkenidir, ancak atama operatörünün sağ tarafında 3.7 reel sayı sabiti vardır. Bu durumda 3.7 değerinin ondalık kısmı otomatik olarak atanır ve 3 tamsayı değeri, k'ya atanır.
z=10.8;	z reel sayı değişkenidir, atama operatörünün sağ tarafındaki 10.8 reel sayı sabiti, z'ye atanır.
z=5;	z reel değişkenidir, ancak atama operatörünün sağ tarafında 5 tamsayı sabiti vardır. Bu durumda 5 tamsayısı otomatik olarak reel sayıya dönüştürülür ve 5.0 değeri, z'ye atanır.

## Değişkenlere İlk Değer Ataması

Tanımlama ve Atama Ayır Komut Olarak:

	1. komut sonrası	2. komut sonrası
<pre>int x; x=5;</pre>	<div>x ?</div>	<div>x 5</div>

Tanımlama ve Atama Tek Komut Olarak:

<pre>int x=5;</pre>	<div>x 5</div>
---------------------	--------------------

## Aritmetik Operatörler

**Tekli (unary) eksi (-):** Sayıyı negatif hale getirir :  $-3$     $-9$

**Tekli (unary) artı (+):** Sayıyı pozitif hale getirir :  $+5$     $+7$  .  $8$

**Çıkarma Operatörü (-):** İki değerin birbirinden çıkarılmasını sağlar,  
 $13 - 1 \rightarrow 12$     $7 - 9 \rightarrow -2$     $2.9 - 0.3 \rightarrow 2.6$

**Toplama Operatörü (+):** İki değerin toplanmasını sağlar,  
 $3 + 1 \rightarrow 4$     $5 + 2 \rightarrow 7$     $1.1 + 0.3 \rightarrow 1.4$

**Bölme Operatörü (/):** Bir değerin diğer bir değere bölünmesini sağlar,  
 $5 / 2 \rightarrow 2$     $-3.0 / 2 \rightarrow -1.5$     $6 / 2 \rightarrow 3$

## Aritmetik Operatörler

**Mod Operatörü (%):** İki tamsayı değerinin birbirine bölünmesinden kalan değeri verir. Sadece tamsayı değerleri için tanımlıdır.

$$5 \% 2 \rightarrow 1$$

$$10 \% 3 \rightarrow 1$$

$$4 \% 2 \rightarrow 0$$


$$\begin{array}{r|l} 5 & 2 \\ -4 & \\ \hline 1 & \end{array} \quad \begin{array}{l} \rightarrow 5/2 \\ \rightarrow 5 \% 2 \end{array}$$

## Aritmetik Operatör Kuralları

- İki operatör yan yana kullanılamaz.  $(2 + / 3)$  geçersiz
- İki tamsayı işleminin sonucu tamsayıdır.  $2 + 3 \rightarrow 5$        $5 / 2 \rightarrow 2$
- Sayılardan birisi reel ise sonuç reel sayıdır.  
 $2.0 + 3 \rightarrow 5.0$        $5 / 2.0 \rightarrow 2.5$
- İşlem sırası parantez kullanılarak belirtilebilir.
- Parantez kullanıldığı durumlarda, işlem içten dışa doğru ilerler.
- Parantezlerin olmadığı durumda öncelik tablosu geçerlidir



# Aritmetik Operatörlerin Öncelik Sırası

Öncelik sırası	Operatör	Özellik
En yüksek	( )	İçten dışa
	- +	Tekli operatör (sağdan sola)
	* / %	İkili operatör (soldan sağa)
	+ -	İkili operatör (soldan sağa)
En düşük		

## Aritmetik Operatörlerin Öncelik Sırası

$9/2*4.2+5/2-1.1$  işleminin sonucunu bulalım.

$$\underline{9 / 2} * 4.2 + 5 / 2 - 1.1$$



$$\underline{4} * 4.2 + 5 / 2 - 1.1$$



$$16.8 + \underline{5 / 2} - 1.1$$



$$\underline{16.8 + 2} - 1.1$$



$$\underline{18.8 - 1.1}$$



$$17.7$$

```
#define sabit_adı değer
```

## Örnek:

Pi sayısının isim sabiti olarak tanımlayan komutu yazalım.

```
#define PI 3.1415
```

Değişkenlerin değerlerinin veya sabitlerin veri tiplerinin başka veri tiplerine dönüştürülmesi veri tipi dönüşümü olarak adlandırılır.

## Otomatik Veri Tipi Dönüşümü

```
double r=0.5, p=5.2, s;  
int i=15, q=10, w;  
char ch;
```

```
s = i/q;           /*s 1.0 değerini alır. * /  
w = r * p;        /*w 2 değerini alır. * /  
ch=5*i;           /*ch 75 değerini alır * /
```

## Tanımlanan Veri Tipi Dönüşümü

*(istenilen\_veri\_tipi) değişken\_ismi*

```
int sayi1, sayi2;  
double bolum;  
sayi1=2;  
sayi2=4;  
bolum=sayi1/sayi2;    /*bolum 0.0 değerini alır */  
bolum=(double)sayi1/ (double)sayi2;  
                        /*bolum 0.5 değerini alır */  
sayi1=(int) 3.6;      /*sayi1 3 değerini alır */
```

# Çıktı Fonksiyonu - printf()

`printf()` fonksiyonu program sonuçlarının ekranda gösterilmesini sağlayan bir kütüphane fonksiyonudur.

```
printf("format dizgisi");
```

**Örnek:** `printf("Bu bir ciktidir.");`

**Çıktı:** Bu bir ciktidir.

# Çıktı Fonksiyonu - printf()

**Örnek:**

```
#include <stdio.h>

int main(void)
{ printf("gecen ogrenci sayisi");
  printf("=30,");
  printf(" kalan ogrenci sayisi=");
  printf("10");
  return(0);
}
```

**Çıktı:**

gecen ogrenci sayisi=30, kalan ogrenci sayisi=10

# Çıktı Fonksiyonu - printf()

Çıktıların ayrı satırlarda gösterilmek isteniyorsa yeni satır karakteri '\n' kullanılmalıdır.

## Örnek:

```
printf("Bu 1. satır. \nBu 2. satır.");
```

**Çıktı:** Bu 1. satır.  
Bu 2. satır.



# Çıktı Fonksiyonu - printf()

`printf()` değişkenlerin veya ifadelerin değerlerinin ekranda gösterilmesini sağlar.

`printf("format dizgisi", çıktı listesi);`

## Örnek:

```
int x = 75;  
printf("%d", x);
```

**Çıktı:** 75

# Çıktı Fonksiyonu - printf()

`printf("x=%d y=%d", x, y);`

Yer belirleyici	Veri tipi
%c	Karakter
%d	Tamsayı
%e	Bilimsel gösterim (scientific notation)
%f %lf	Reel sayı (decimal, floating point)
%g	%e ve %f'den hangisi daha kısa ise onu kullanır
%s	Dizgi (string)
%u	İşaretsiz ondalık (Unsigned decimal)
%x	Hexadecimal

# Çıktı Fonksiyonu - printf()

## Formatlı Çıktı

	Örnek	Çıktı
<i>%nd</i>	<code>printf("%4d", 33);</code>	<code>33</code>
<i>%nc</i>	<code>printf("%3c", 'M');</code>	<code>M</code>
<i>%ns</i>	<code>printf("%10s", "Merhaba");</code>	<code>Merhaba</code>
<i>%n.mf</i>	<code>printf("%f", 12.236);</code>	<code>12.236000</code>
<i>%n.me</i>	<code>printf("%10.3e", -0.0536);</code>	<code>-5.350e-02</code>

# Çıktı Fonksiyonu - printf()

## Ters Eğik Çizgi Karakter Sabitleri (\)

Kod	Açıklama
\b	Geriye doğru boşluk (backspace)
\f	Form besleme (form feed)
\n	Yeni satır
\r	Satır başı (carriage return)
\t	Sekme (horizontal tab)
\'	Tek tırnak karakteri
\0	Boş (null)

# Çıktı Fonksiyonu - printf()

## Ters Eğik Çizgi Karakter Sabitleri (\)



```
printf ("%s\b%s", "Merhaba", "Nasilsin?");
```

Geriye doğru bir boşluk ver

**Çıktı:** MerhabNasilsin?



```
printf ("%s\t\t\t%s", "Merhaba", "Nasilsin?");
```

Üç sekme kadar ilerle

**Çıktı:** Merhaba

Nasilsin

# Girdi Fonksiyonu - scanf()

`scanf()` fonksiyonu kullanıcı tarafından veri girişinin yapılmasını ve bu verilerin girdi listesinde belirtilen değişkenlerde saklanmasını sağlayan bir fonksiyondur.

`scanf("format_dizisi", girdi_listesi);`

```
int a,b;  
...  
scanf("%d%d", &a, &b);
```

**Girdi:** 3 5

3

a

5

b

Tablo 3.4 Bazı Matematik Kütüphane Fonksiyonları

<u>Fonksiyon</u>	<u>Açıklaması</u>
<code>cos (x)</code>	x parametresi ve fonksiyonun sonucu double veri tipindedir. Verilen x değerinin kosünüs karşılığını hesaplar.
<code>log (x)</code>	x parametresi ve fonksiyonun sonucu double veri tipindedir. Verilen x değerinin logaritma karşılığını hesaplar.
<code>pow(x,y)</code>	x, y parametreleri ve fonksiyonun sonucu double veri tipindedir. Verilen x değerinin $x^y$ değerini hesaplar.
<code>sqrt (x)</code>	x ve fonksiyonun sonucu double veri tipindedir. Verilen x değerinin kara kökünü hesaplar.

**Örnek:** `sqrt (a*b-c/6.0) ;`  
`pow (p*q, 5.0)`