

C# OPERATÖRLER

+ - / * = ? ! ++ -- != <

Adem AKKUŞ
Bilgisayar Mühendisi
Bilişim Teknolojileri Öğretmeni
Eğitmen

+ - / * = ? !

OPERATÖR NEDİR?

operatör :Programlama dillerinde tek başlarına herhangi bir anlamı olmayan ancak programın işleyişine katkıda bulunan karakter ya da karakter topluluklarına **operatör** denir.

Örneğin $a+b$ ifadesinde **+** işareti bir operatördür.

operand :Operatörlerin etki ettikleri sabit ya da değişkenlere ise **operand** denir.

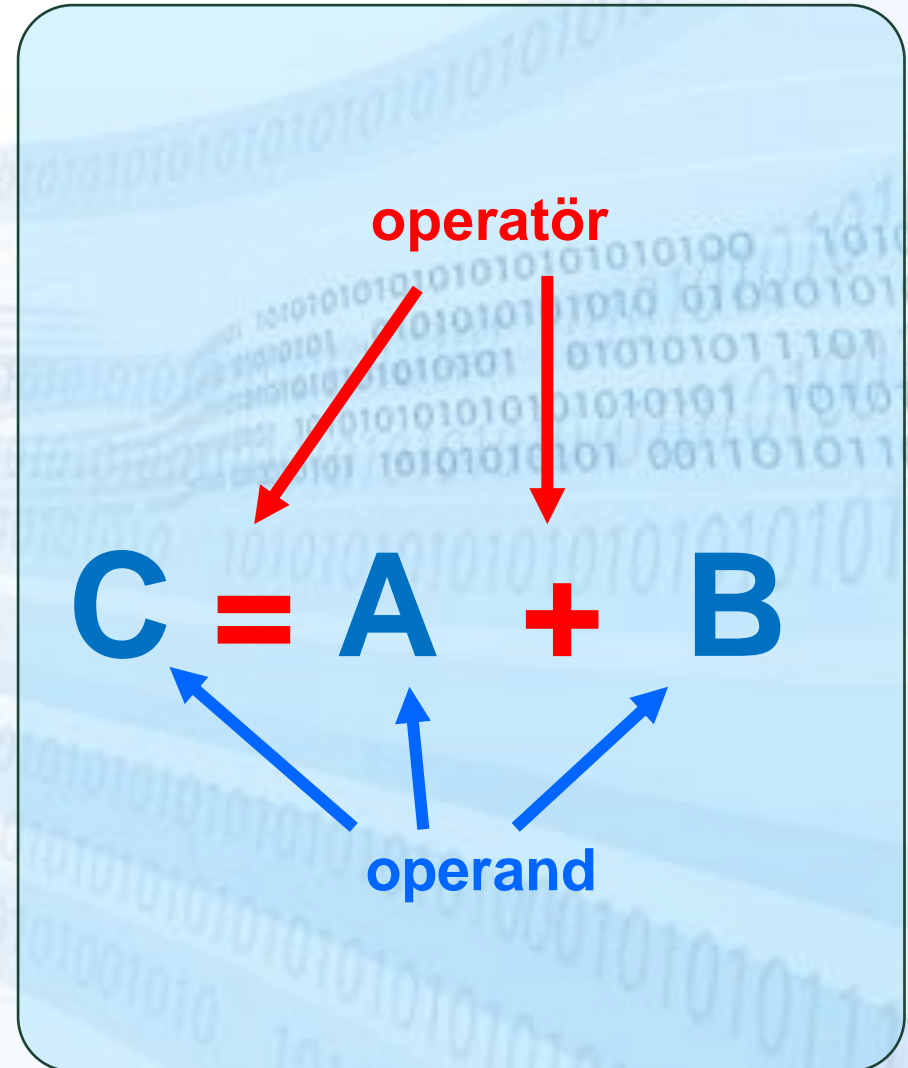
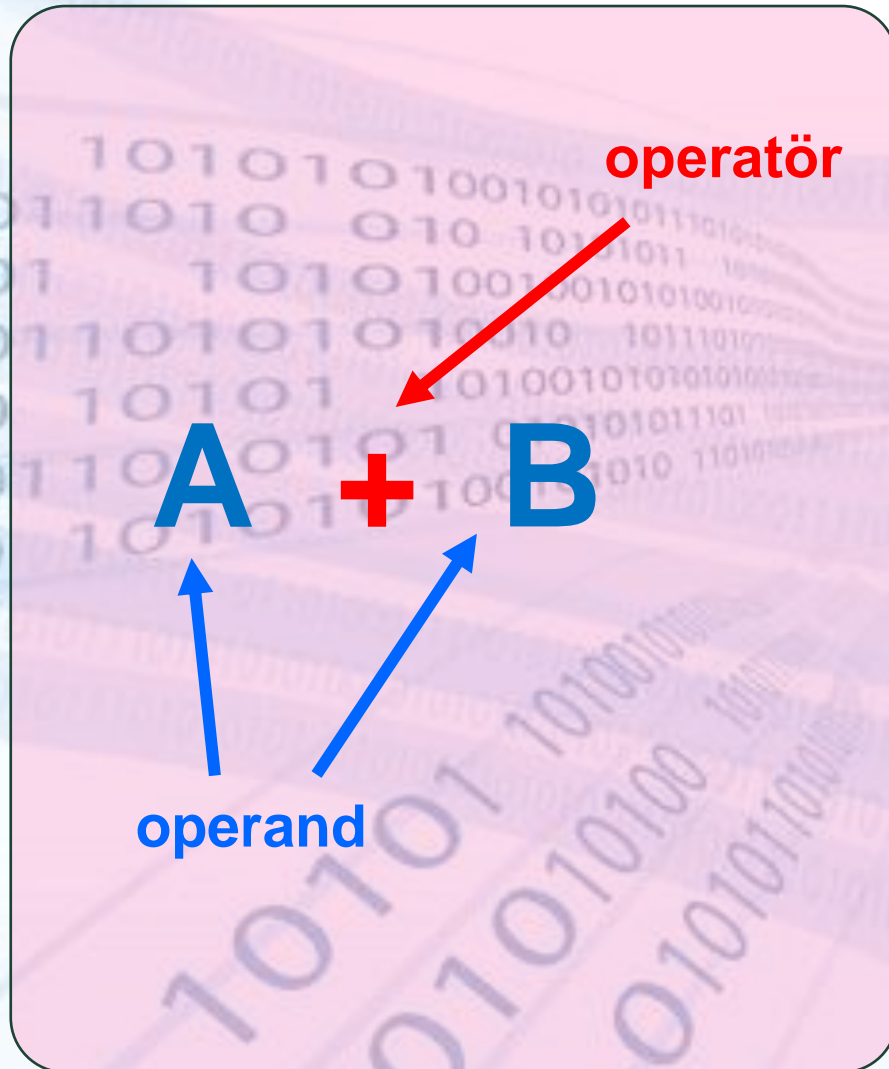
Örneğin $a+b$ ifadesinde **a** ve **b** birer operandır.

C#'ta operatörlerin bazıları tek, bazıları çift, bazıları da üç operand alır.

+ - / * = ? ! ++ -- != <

+ - / * = ? !

OPERAND NEDİR?



OPERATÖR NEDİR?

+ - / * = ? !

Önceden tanımlanmış olan ve birtakım

- matematiksel ,
- mantıksal ,
- ilişkisel ,
- bitisel (bitwise),
- atama,
- özel amaçlı,

işlemleri gerçekleştirmek için kullanılan özel karakter ya da karakterler topluluğudur.

Bazı operatörler tek karakterden (unary) bazıları iki karakterden (binary) ve bazıları üç (ternary) karakterden oluşur.

Operatörler işlemi yapacağı operandlara ihtiyaç duyarlar.

+ - / * = ? ! ++ -- != <

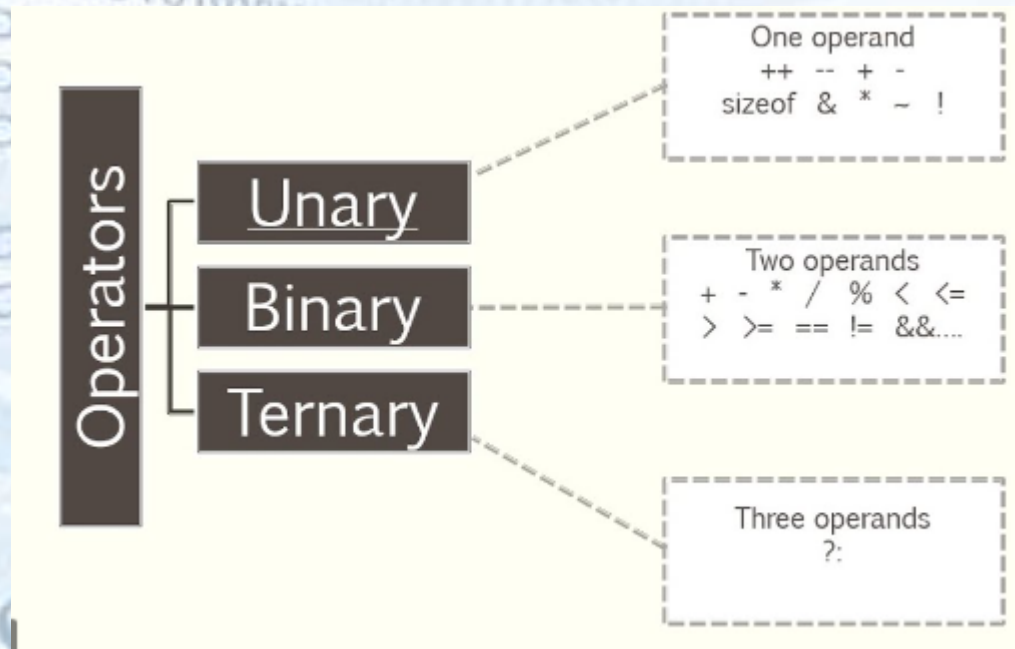
OPERATÖRLERİN ÇEŞİTLERİ

S.No	Operatör Grubu	Örnekler
1	Aritmetiksel Operatörler	+ , - , * , / , ++ , --
2	Karşılaştırma Operatörleri	< , <= , > , >= , == , != ,
3	Bitsel Operatörler	& (and), (or) , ~ (not) , ^ (xor)
4	Mantıksal Operatörler	&& (and), (or) , ! (not)
5	Atama Operatörleri	= , *= , /= , += , -= , &= , ^= , =
6	Özel Amaçlı Operatörler	?: , () , [] , typeof , sizeof , new

+ - / * = ? !

unary binary ternary ?

Aldıkları operand sayısına göre operatörler 3'e ayrılır.



+ - / * = ? ! ++ -- != <

unary

+ - / * = ? !

Unary: tek operand üzerinde işlem yapan operatörlerdir..

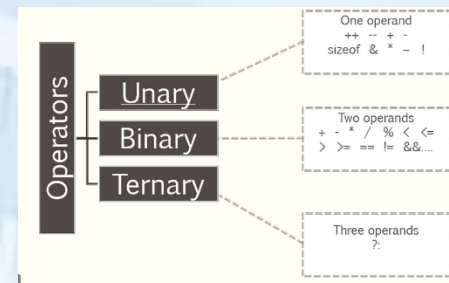
```
int negative = -1;
int positive = 1;
int result;

result = +negative;    // result = -1
result = +positive;    // result = 1
```

```
int count;
int index = 6;

count = index++; // count = 6, index = 7
```

+ - / * = ? ! ++ -- != <



unary

+ - / * = ? !

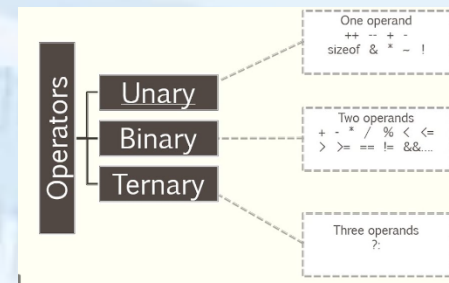
Unary: tek operand üzerinde işlem yapan operatörlerdir..

```
int count;  
int index = 6;  
  
count = --index;    // count = 5, index = 5
```

```
int count;  
int index = 6;  
  
count = index--;    // count = 6, index = 5
```

```
bool bexpr    = true;  
bool bresult  = !bexpr;    // bresult = false  
bresult       = !bresult;  // bresult = true
```

+ - / * = ? ! ++ -- != <



unary

+ - / * = ? !

```
#region Unary Operatörler
//unary operator: tekli operatör demektir. tek operand alır ++a, --b
int a = 10;
Console.WriteLine(a);
++a; //a nın değerini bir artır
Console.WriteLine(a);
byte x = 99;
byte y = ++x; //önce x in değerini bir artır sonra y'ye ata x=100 ,y=100
Console.WriteLine($"x={x} ve y={y}");
//
byte m = 100;
byte n = m++; //önce m nin değerini n ye ata; sonra m nin değerini bir artır m=101, n=100
//--
byte k = 5;
byte l = --k; //önce k in değerini bir azalt sonra l'ye ata k=4 ,l=4
Console.WriteLine($"k={k} ve l={l}");
//
short p = 1000;
short r = p--; //önce p nin değerini r ye ata; sonra p nin değerini bir azalt p=999 ,r=1000
Console.WriteLine($"p={p} ve r={r}");
//! :not değili anlamında gelir . true-->false, false-->true
bool durum = false;
bool digerdurum = !durum;
Console.WriteLine($"durum={durum} ve diğer durum={digerdurum}");

#endregion
```

binary

+ - / * = ? !

binary: iki operand üzerinde işlem yapan operatörlerdir..

```
int one = 1;
int two;

two = one + one; // two = 2
```

```
int dividend = 33;
int divisor = 10;
int remainder;

remainder = dividend % divisor; // remainder = 3
```

```
int dividend = 45;
int divisor = 5;
int quotient;

quotient = dividend / divisor; // quotient = 9
```

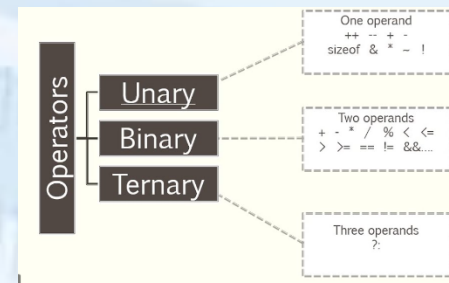
```
int expr1 = 3;
int expr2 = 7;
int product;

product = expr1 * expr2; // product = 21
```

```
decimal debt    = 537.50m;
decimal payment = 250.00m;
decimal balance;

balance = debt - payment; // balance = 287.5
```

+ - / * = ? ! ++ -- != <



binary

+ - / * = ? !

binary: iki operand üzerinde işlem yapan operatörlerdir..

```
bool bresult;  
decimal debit  = 1500.00m;  
decimal credit = 1395.50m;  
  
bresult = debit == credit; // bresult = false
```

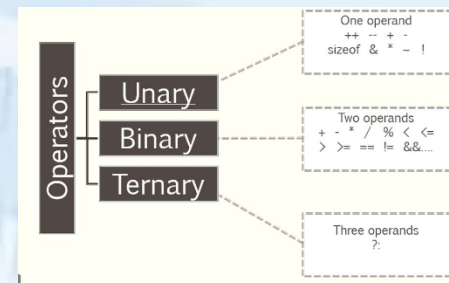
```
short redBeads   = 2;  
short whiteBeads = 23;  
bool bresult;  
  
bresult = redBeads < whiteBeads; // bresult=true,
```

```
float limit      = 4.0f;  
float currValue  = 3.86724f;  
bool bresult;  
  
bresult = currValue <= limit; // bresult = true
```

```
bool bresult;  
decimal debit  = 1500.00m;  
decimal credit = 1395.50m;  
  
bresult = debit != credit; // bresult = true  
bresult = !(debit == credit); // bresult = true
```

```
short redBeads   = 13;  
short whiteBeads = 12;  
bool bresult;  
  
bresult = redBeads > whiteBeads; // bresult=true
```

+ - / * = ? ! ++ -- != <



binary

+ - / * = ? !

```
//binary operator: ikili operatör. iki tane operand alır
#region binary operatör ilk
//a+b, a-b,a*b,a/b,a%b
int x = 10;
int y = 15;
Console.WriteLine($"x={x}, y={y} , z={z}");
int t = ++x + y; //x=12,y=15, t=27
Console.WriteLine($"x={x}, y={y} , t={t}");
#endregion
```

```
//beş işlem
//değişken tanımlama
int sayi1, sayi2;
int toplama, cikarma, carpma, modalma;
double bolme;

//değişkenlere değer ataması
Console.Write("Birinci sayıyı giriniz:");
sayi1 = Convert.ToInt32(Console.ReadLine());
Console.Write("İkinci sayıyı giriniz:");
sayi2 = Convert.ToInt32(Console.ReadLine());
//işlemler
toplama = sayi1 + sayi2;
cikarma = sayi1 - sayi2;
carpma = sayi1 * sayi2;
bolme = (double) sayi1 / (double) sayi2;
modalma = sayi1 % sayi2;
```

+ - / * = ? !

binary

+ - / * = ? !

```
#region 1.yöntem
Console.WriteLine("-- Beş işlem -- ");
Console.WriteLine("***** 1. yol*****");
Console.WriteLine($"{sayi1} + {sayi2} = {toplama}");
Console.WriteLine($"{sayi1} - {sayi2} = {cikarma}");
Console.WriteLine($"{sayi1} * {sayi2} = {carpma}");
Console.WriteLine($"{sayi1} / {sayi2} = {bolme}");
Console.WriteLine($"{sayi1} % {sayi2} = {modalma}");
```

```
#region 2.yol
Console.WriteLine("*****2. yol*****");
Console.WriteLine("{0}+{1}={2}\n{0}-{1}={3}\n{0}*{1}={4}\n{0}/{1}={5}\n{0}%{1}={6}", sayi1, sayi2, toplama, cikarma, carpma, bolme, modalma);
#endregion
```

```
#region 3.yol
Console.WriteLine("***3.yol***");
Console.WriteLine($"{sayi1} + {sayi2} = {sayi1 + sayi2}");
Console.WriteLine($"{sayi1} - {sayi2} = {sayi1 - sayi2}");
Console.WriteLine($"{sayi1} * {sayi2} = {sayi1 * sayi2}");
Console.WriteLine($"{sayi1} / {sayi2} = {(double)sayi1 / sayi2}");
Console.WriteLine($"{sayi1} % {sayi2} = {sayi1 % sayi1}");
#endregion
```

+ - / * = ? ! ++ -- != <

ternary

+ - / * = ? !

ternary:

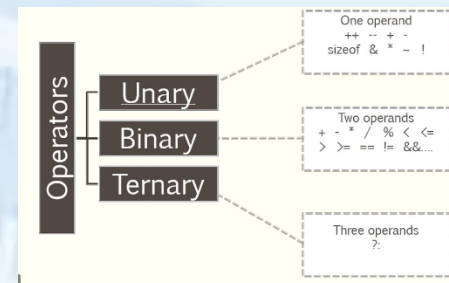
İlk ifade bir Boolean ifadesi olmalıdır. İlk ifade **true** olarak değerlendirildiğinde, değer ikinci ifadenin değeri döndürülür. İlk ifade **false** olarak değerlendirildiğinde, üçüncü ifadenin değeri döndürülür. İlk ifade **false** olarak değerlendirildiğinde, üçüncü ifadenin değeri döndürülür.

```
Condition ? Expression1 : Expression2;
```

```
if (number % 2 == 0)
{
    isEven = true;
}
else
{
    isEven = false;
}
```

```
isEven = (number % 2 == 0) ? true : false ;
```

+ - / * = ? ! ++ -- < > <= >= != ==



ternary

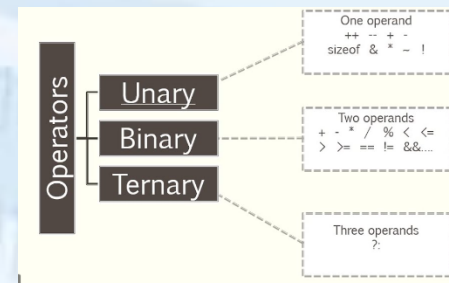
+ - / * = ? !

ternary:

İlk ifade bir Boolean ifadesi olmalıdır. İlk ifade **true** olarak değerlendirildiğinde, değer ikinci ifadenin değeri döndürülür. İlk ifade **false** olarak değerlendirildiğinde, üçüncü ifadenin değeri döndürülür. İlk ifade **false** olarak değerlendirildiğinde, üçüncü ifadenin değeri döndürülür.

```
result = a > b ? "a is greater than b" : a < b ? "b is greater than a" : "a is equal to b";
```

```
if (a > b)
{
    result = "a is greater than b";
}
else if (a < b)
{
    result = "b is greater than a";
}
else
{
    result = "a is equal to b";
}
```



+ - / * = ? !

ternary

+ - / * = ? !

```
//sonuc=şart ? ifade1 : ifade2 ;  
//ternary operator:3 oprerand alır .  
//şartın true olması durumunda ,ifade1 ;false ise ifade2 çalışacaktır  
//bool result = (27< 5) ? true : false;  
//Console.WriteLine(result);  
//kullanıcı tarafından girilen sayının tek mi çift mi olduğunu yazan program  
int sayi;  
int sayi2;  
Console.Write("Bir sayı giriniz:");  
sayi = Convert.ToInt32(Console.ReadLine());  
//sayi = int.Parse(Console.ReadLine());  
//sayi = Int32.TryParse(Console.ReadLine(), out sayi2);  
string sonuc = (sayi % 2 == 0) ? "Sayı Çift" : "Sayı Tek.";   
Console.WriteLine(sonuc);  
//iki sayıdan büyük olanı bulan program  
Console.Write("İkinci sayıyı giriniz:");  
sayi2 = Convert.ToInt32(Console.ReadLine());  
string sonuc2 = (sayi < sayi2) ? "İkinci sayı büyüktür" : "Birinci sayı büyüktür.";   
Console.WriteLine(sonuc2);
```

+ - / * = ? ! ++ -- != <

Aritmetiksel Operatörler ?

+ , **-** , ***** , **/** , **++** , **--** , **%**

Aritmatiksel işlemler için kullanılır.

```
byte a=10; string s;
```

```
byte b=3,c=20;
```

```
int d=a+c*b%2;
```

Sonuçta d=10 olur.

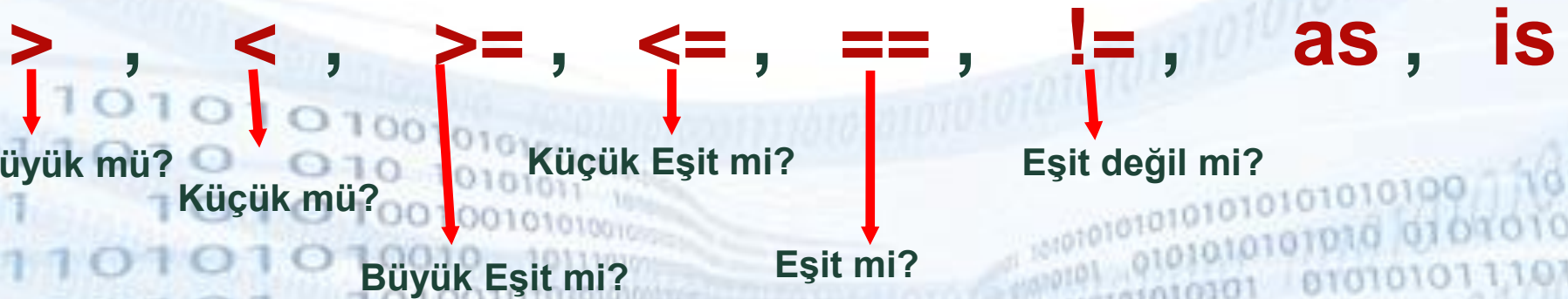
```
byte a=10;
```

```
byte b=++a;
```

```
byte c=a--;
```

a=? b=? c=?

Karşılaştırma Operatörleri ?



Mantıksal karşılaştırmalar için kullanılır.

```
byte a=10; string s;  
byte b=3,c=20;  
Console.WriteLine(a==b);
```

```
byte a=10;  
Console.WriteLine(a is System.Byte);
```

Karşılaştırma Operatörleri ?

```
// <, >, <=, >=, ==, !=  
//sonuc boolean dir. True ya da False değerini verecektir
```

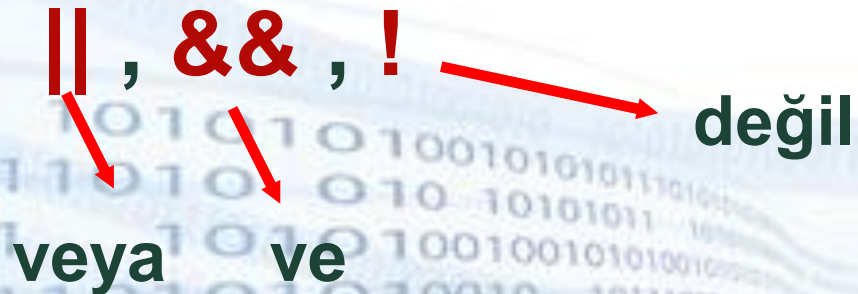
```
int s1 = 12;  
int s2 = 13;  
bool result1 = s1 < s2;  
Console.WriteLine(result1);  
Console.WriteLine(s1<s2);  
Console.WriteLine(s1>s2);  
Console.WriteLine(s1<=s2);  
Console.WriteLine(s1>=s2);  
Console.WriteLine(s1==s2);  
Console.WriteLine(s1!=s2);  
Console.WriteLine("-----");  
Console.WriteLine(100<103);  
Console.WriteLine("*****");  
Console.WriteLine("adem"=="adem");  
Console.WriteLine('a'>'A');  
Console.WriteLine(12.33 < 22.44);  
Console.ReadKey();
```

E:\MCSD\MCS8 Ocak\5.Hafta\Uyg

```
True  
True  
False  
True  
False  
False  
True  
-----  
True  
*****  
True  
True  
True
```


Mantıksal Operatörler ?

|| , **&&** , **!**

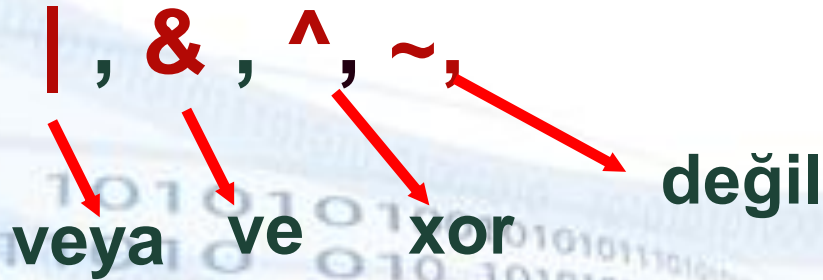
A diagram showing three red arrows pointing from the operators to their names: one from '||' to 'veya', one from '&&' to 've', and one from '!' to 'değil'.

veya ve değil

Mantıksal karşılaştırmalar için kullanılır.

```
byte a=10;  
string s;  
byte b=3,c=20;  
Console.WriteLine(!(a==b)&&(b==3));
```


Bitsel Operatörler ?



Bitsel işlemler için kullanılırlar. Sonuçta matematiksel bir sonuç üretirler.

```
int a = 60;          /* 60 = 0011 1100 */
int b = 13;          /* 13 = 0000 1101 */
```

```
int c = a & b;        /* 12 = 0000 1100 */
int d = a | b;        /* 61 = 0011 1101 */
int e = a ^ b;        /* 49 = 0011 0001 */
```

Bitisel Operatörler ?

Bitisel işlemler için kullanılırlar. Sonuçta matematiksel bir sonuç üretirler.

Bitisel operatör, bitler üzerinde çalışır ve bit bit işlem gerçekleştirir. $\&$, $|$ ve \wedge için doğruluk tabloları aşağıdaki gibidir

p	q	$p \& q$	$p q$	$p \wedge q$
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

Bitisel Operatörler ?

Bitisel işlemler için kullanılırlar. Sonuçta matematiksel bir sonuç üretirler.

Bitisel operatör, bitler üzerinde çalışır ve bit bit işlem gerçekleştirir. $\&$, $|$ ve \wedge için doğruluk tabloları aşağıdaki gibidir

AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

XOR

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

NOT

A	B
0	1
1	0

Bitisel Operatörler ?

Bitisel işlemler için kullanılırlar. Sonuçta matematiksel bir sonuç üretirler.

Bitisel operatör, bitler üzerinde çalışır ve bit bit işlem gerçekleştirir. **&**, **|** ve **^** için doğruluk tabloları aşağıdaki gibidir

```
int a = 60;          /* 60 = 0011 1100 */
int b = 13;          /* 13 = 0000 1101 */
```

```
int c = a & b;        /* 12 = 0000 1100 */
int d = a | b;        /* 61 = 0011 1101 */
int e = a ^ b;        /* 49 = 0011 0001 */
```


Diğer Operatörler

[] , () , typeof

```
string mesaj = "Merhaba";  
char c = mesaj[2];  
Console.WriteLine(c);
```

```
Console.WriteLine(typeof(int));  
//System.Int32  
  
int s=5;  
Console.WriteLine(s.GetType());  
//System.Int32
```

Üç tane operand alan bir tane operatör vardır.

O da **?:** operatörüdür.

```
int a=10; string s;  
int b=20;
```

```
(a==b)? s="Doğru":s="Yanlış";
```

Operatör Önceliği ?

Matematiksel operatörler için geçerli olan kurallar geçerlidir. Önce parantez içleri halledilir. Daha sonra çarpma ve bölme ve en son toplama çıkarma

```
byte a=10; string s;
```

```
byte b=3,c=20;
```

```
Console.WriteLine((a+b)*(b+c%5));
```