Increment / Bir Variable'in Degerini Artirma Yontemleri

```
int numA = 2 ;
numA  = numA + 3;
```

veya

```
numA + = 3
```

?

```
int numB = 10 ;
numB  = numB * 7;
```

veya

```
numB * = 7
```

?

```
int numC = 7 ;
numC++;
```

?

```
int numD = 11 ;
numD++ ;
```

?

Decrement / Bir Variable'in Degerini Azaltma Yontemleri

```
int numA = 2 ;
numA  = numA - 3;
```

veya

```
numA - = 3
```

?

```
int numB = 20 ;
numB  = numB / 5;
```

veya

```
numB / = 5
```

?

```
int numD = 7 ;
numD - - ;
```

?

```
int numE = 11 ;
numE - - ;
```

?

Pre-Increment & Post Increment

- Pre-Increment ve Post Increment operatorlerinin her ikisi de artirma islemi icin kullanilir
- Pre-Increment isleminde variable statement'da kullanilmadan once artirilir veya azaltilir

```
public static void main(String[] args) {
    int a=15;
    int b=++a;
    System.out.println(b);
}
```

Output : 16

- Post Increment isleminde variable statement'da kullanilir, sonra artirilir veya azaltilir

```
public static void main(String[] args) {
    int a=15;
    int b=a++;
    System.out.println(b);
}
```

Output : 15

Javada Matematiksel Operatorler

1- Ustel islemler
2- Parantez ici
3- Carpma-Bolme
4- Toplama-cikarma

Ornek 1 :

```
38 / 2 * ( 4 + 3 ) * 2 =
```

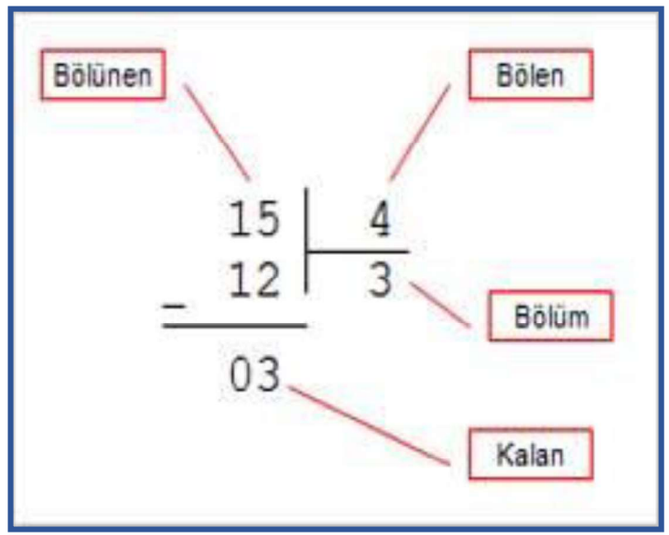
Ornek 2 :

```
8 + 2 * ( 14 – 6 / 2 ) - 12 =
```

Modulus %

Modulus islemi bir bolme isleminde kalan sayiyi bize verir

```
public static void main(String[] args) {
    int a=15 % 4;
    System.out.println(a);
}
```



Modulus %

Soru) Kullanicidan 4 basamakli bir sayi alin ve rakamlar toplamini bulup yazdirin

Ipucu 1:

Sayi % 10 => Bize son basamagi verir

538 % 10 = 8

Ipucu 2:

Int Sayi /10 => Bize son basamak haric sayiyi verir

int sayi=538;

sayi = sayi / 10 =>

sayi'ya 53 degerini atar

Wrapper Class

Java primitive data turleri ile methodlari kullanabilmemiz icin Wrapper class'lari olusturmustur.

Character,Byte,Integer,Short,Float,Double primitive data turleri icin olusturulan wrapper class'lardir.

```
public  class Example {
public static void main(String[] args) {

    int num1 = Integer.MIN_VALUE;
    System.out.println(num1);

    int num2 = Integer.MAX_VALUE;
    System.out.println(num2);

    int num3 = Byte.MIN_VALUE;
    System.out.println(num3);

    int num4 = Byte.MAX_VALUE;
    System.out.println(num4);

}
}
```

```
-2147483648

2147483647

-128

127
```

Concatenation / (String Datalari Birlestirme)

Birden cok String'i + isareti ile topladiginizda Java bu String degiskenleri birlestirerek yeni bir String olusturur

```
String a = "Hello";
String b = "World";
System.out.println(a+b);
System.out.println(a+" "+b);
```

```
HelloWorld
Hello World
```

Not : Eger matematiksel bir islemin icinde String kullanilirsa, matematikteki oncelikler dikkate alinarak islem yapilir. Sira String ile toplamaya geldiginde toplama yerine Concatenation uygulanir

```
String a = "Hello";
int b = 2;
int c = 3;

System.out.println(a+b+c);
System.out.println(c+b+a);
System.out.println(a+(b+c));
System.out.println(a+b*c);
```

```
Hello23
5Hello
Hello5
Hello6
```

Concatenation

Soru 1) Asagida verilen variable'lari kullanarak istenen sonuclari yazdiran programlari yaziniz.

Variables

```
String str1= “Java”;
String str2= “Guzel”;
int sayi1=5;
int sayi2=4;
```

Istenen Yazilar

1) Java Guzel 54
2) Java 5 Guzel
3) Java 94
4) Java 19
5) 54 Guzel

Relational Operators /(Karsilastirma Operatorleri)

= Assignment (Atama yapar) operatoru

int num1=3;	num1 degiskenine 3 degerini atar
String str1 = “Ali” + “ ” + “Can”;	str1’e Ali Can degeri atar
c = c+5;	c’nin degerini 5 artirir ve son degeri c’ye atar

== Cift esittir isareti / karsilastirma (Comperison) operatoru

boolean sonuc1 = 5+2 == 7;	sonuc1 degeri true olur
boolean sonuc2 = 5*2 == 15;	sonuc2 degeri false olur

Relational Operators /(Karsilastirma Operatorleri)

!= Esit degildir isareti

```
boolean sonuc1= 5+2 != 7;                    sonuc1 degeri false olur
System.out.println(5*2 != 15);               true yazdirir
```

> Buyuktur , >= Buyuk veya esittir

```
boolean sonuc1= 5+2 >= 7;                    sonuc1 degeri true olur
System.out.println(5*2 > 15);                false yazdirir
```

< Kucuktur , <= Kucuk veya esittir

```
boolean sonuc1= 5+2 < 7;                     sonuc1 degeri false olur
System.out.println(5*2 < 15);                true yazdirir
```

Conditional Operators / (Sart Operatorleri)

&& AND (ve) isareti

&& isareti ile birlestirilen tum ifadeler dogru ise sonuc true olur.
Diger tum durumlarda false doner. (&& operatoru mukemmeliyetcidir)

```
boolean sonuc1= (5+2 == 7) && (4+3 !=5) ;
System.out.println((5*2 != 15) && (5>7));
```

sonuc1 degeri true olur
false yazdirir

|| OR (veya) isareti

|| isareti ile birlestirilen tum ifadeler yanlis ise sonuc false olur.
Diger tum durumlarda truee doner. (|| operatoru iyimserdir)

```
boolean sonuc1= (5+2 == 7) || (4+3 !=5) ;
System.out.println((5*2 == 15) || (5>7));
```

sonuc1 degeri true olur
false yazdirir

& Ile && Arasindaki Fark

& isareti kullanildiginda Java isaretin iki yanindaki mantiksal ifadelerin ikisini de kontrol eder. Bu islem kodumuzu yavaslatir

40<30 & 50==50 & 60>50

ilk karsilastirma yanlis olmasina ragmen Java tum karsilastirmalari kontrol etmeye devam eder.

&& isareti kullanildiginda ise Java en bastan kontrol etmeye baslar, mantiksal ifadelerin birinde yanlisi bulursa sonrakileri kontrol etme ihtiyaci duymaz. Bu islem kodumuzu hizlandirir

40<30 && 50==50 && 60>50

ilk karsilastirma yanlis oldugunu gorunce Java diger karsilastirmalari kontrol etmeden alt satira gecer.