

JAVADA METOTLAR

YAZM-209 NESNE TABANLI

Programlama

DR. ÖĞR. ÜYESİ SERPİL ASLAN

Açık problem

Amacımız sırasıyla 1'den 10, 10'dan 20'ye ve 35'ten 45'e kadarki sayıların toplamını bulmak olsun.

Ne yapmak gerekir?

Problem

```
int toplam = 0;
for (int i = 1; i <= 10; i++)
    toplam += i;
System.out.println("1-10 araliginin toplami:"+toplam);

toplam = 0;
for (int i = 20; i <= 30; i++)
    toplam += i;
System.out.println("20-30 araliginin toplami:"+toplam);

toplam = 0;
for (int i = 35; i <= 45; i++)
    toplam += i;
System.out.println(" 20-30 araliginin toplami:"+toplam);
```

Problem

```
int toplam = 0;  
for (int i = 1; i <= 10; i++)  
    toplam += i;
```

```
System.out.println("1-10 araliginin toplamı:"+toplam);
```

```
toplam = 0;  
for (int i = 20; i <= 30; i++)  
    toplam += i;
```

```
System.out.println("20-30 araliginin toplamı:"+toplam);
```

```
toplam = 0;  
for (int i = 35; i <= 45; i++)  
    toplam += i;
```

```
System.out.println(" 20-30 araliginin toplamı:"+toplam);
```

Çözüm

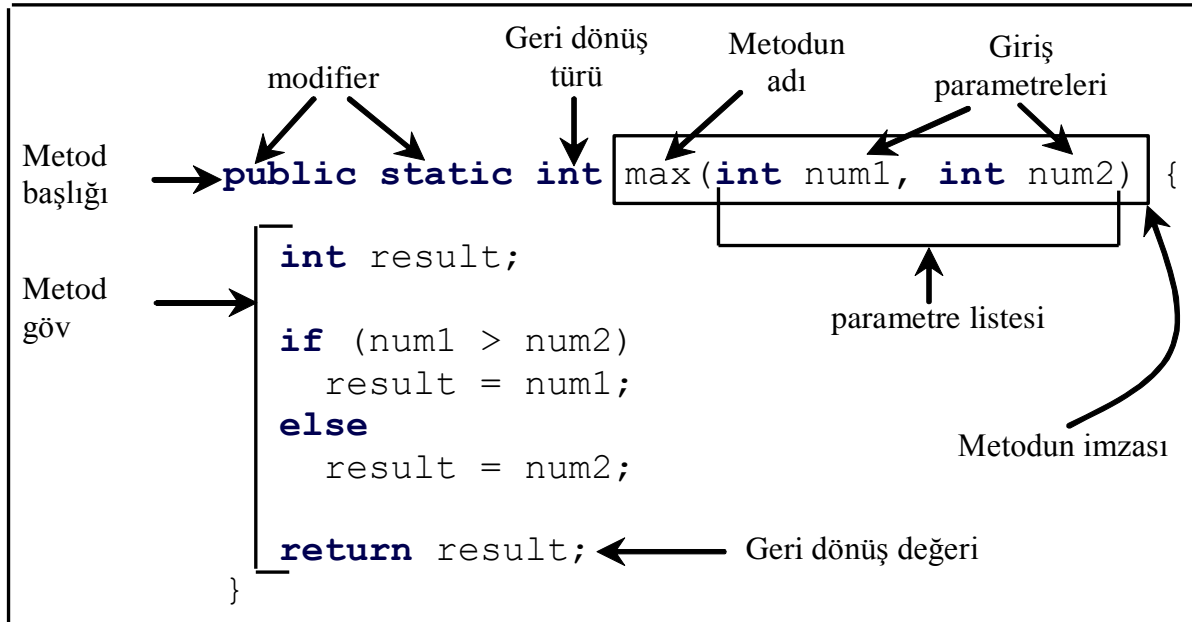
```
public static int toplam(int i1, int i2) {  
    int top = 0;  
    for (int i = i1; i <= i2; i++)  
        top += i;  
    return top;  
}
```

```
public static void main(String[] args) {  
    System.out.println("1-10 aralık toplamı: " + toplam(1, 10));  
    System.out.println("20-30 aralık toplamı: " + toplam(20, 30));  
    System.out.println("35-45 aralık toplamı: " + toplam(35, 45));  
}
```

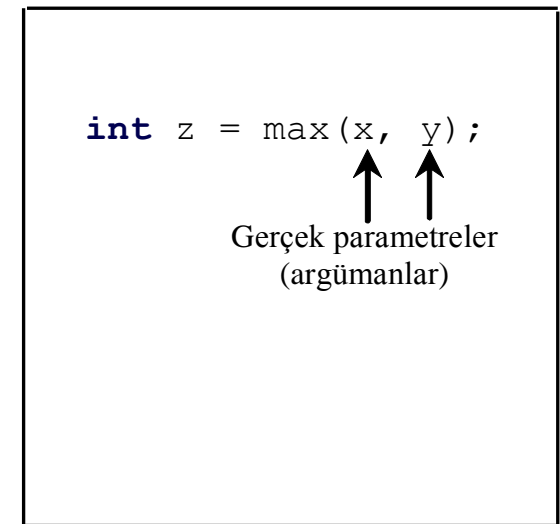
Metotların Tanımlanması

Bir metot bir işlemi gerçekleştirmek için yazılan komutların bir koleksiyonudur.

Metodun tanımlanması



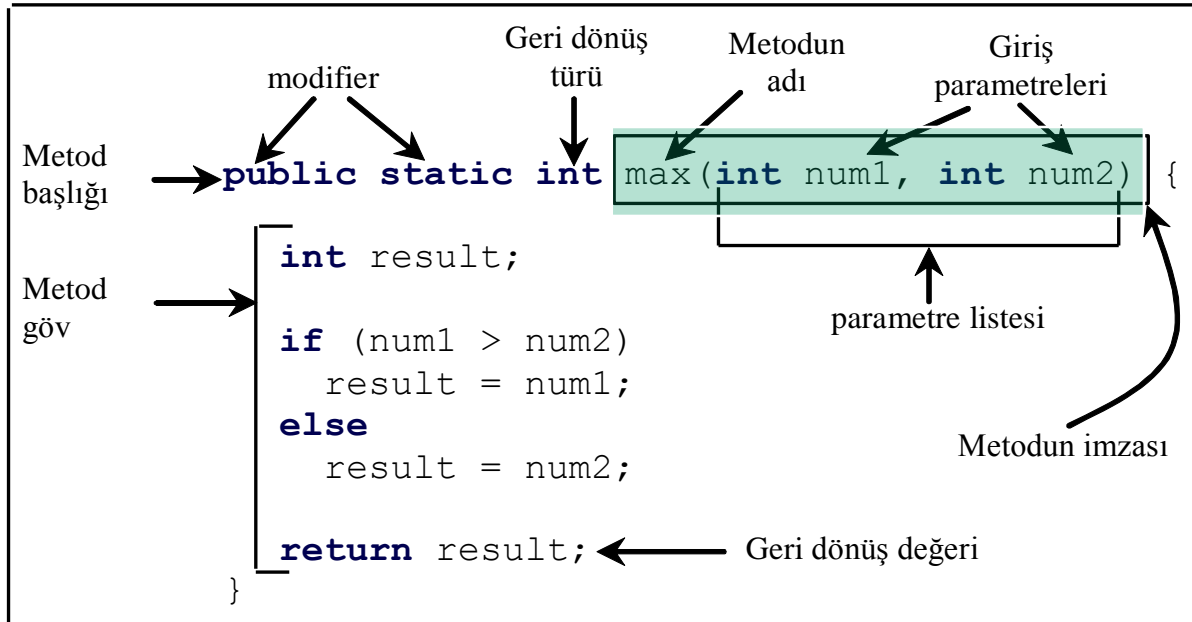
Metodun çağırılması



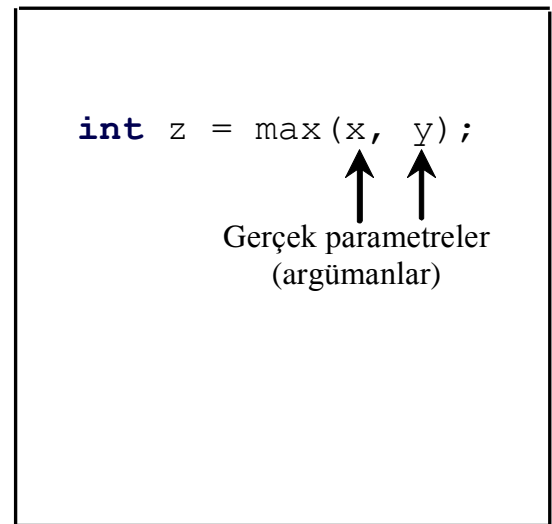
Metot İmza

Metot imzası metodun adı ve parametre listesinin birleşimidir.

Metodun tanımlanması



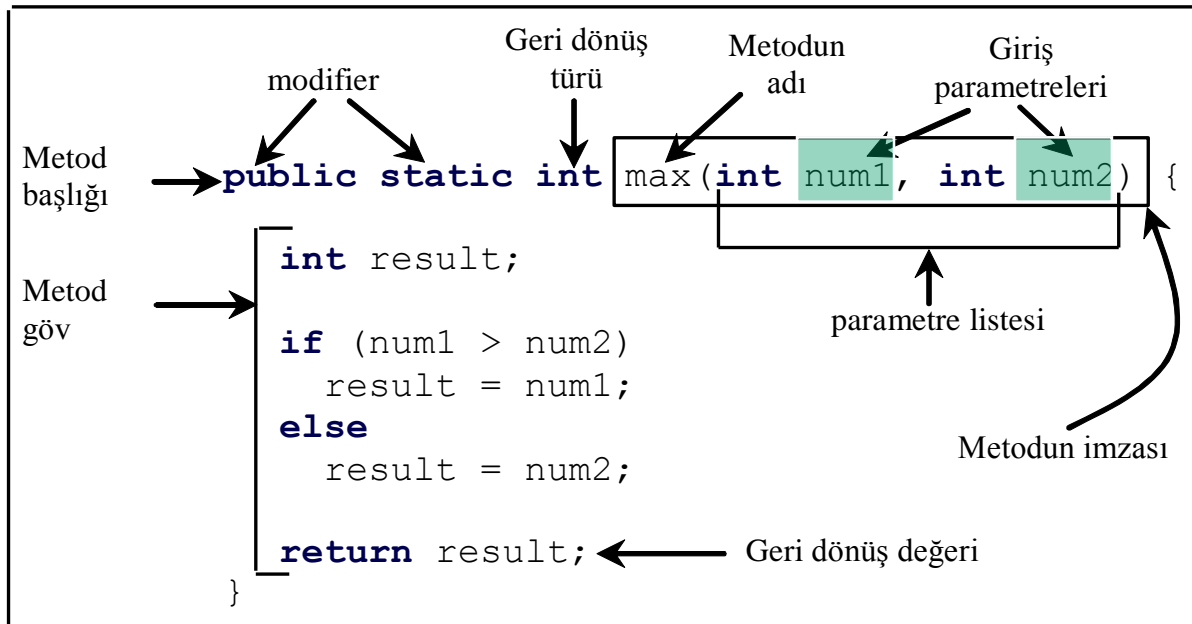
Metodun çağırılması



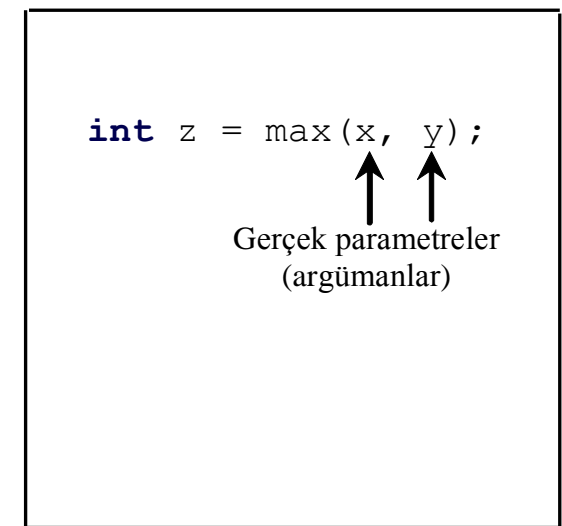
Giriş Parametreleri

Metot başlığında tanımlanan değişkenler giriş parametreleridir.

Metodun tanımlanması



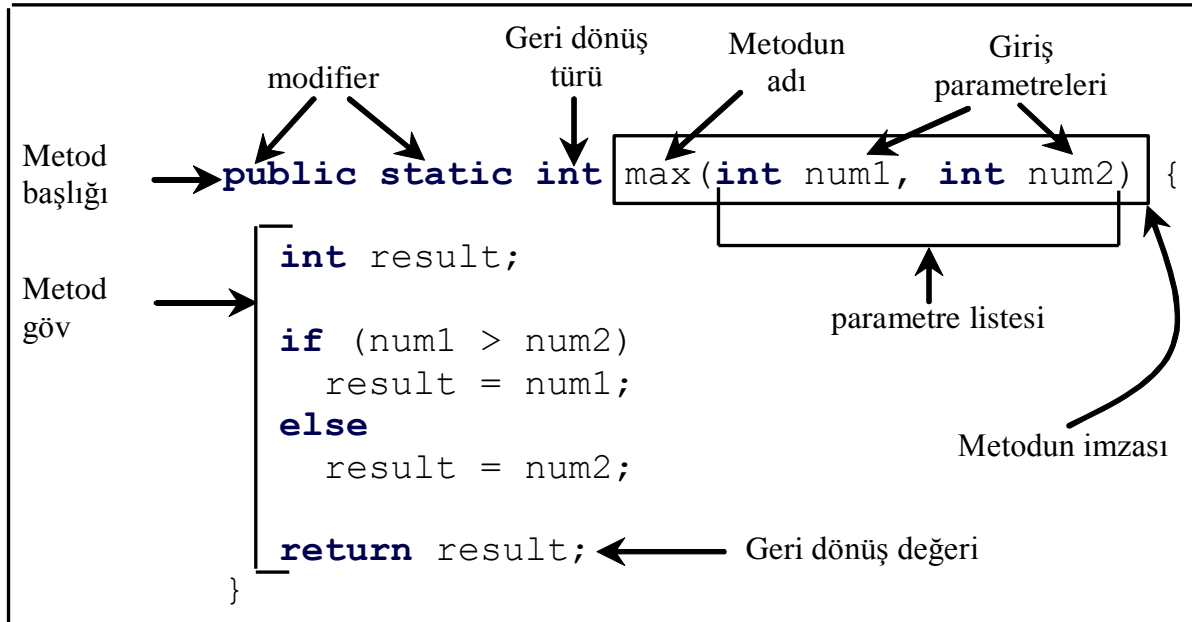
Metodun çağırılması



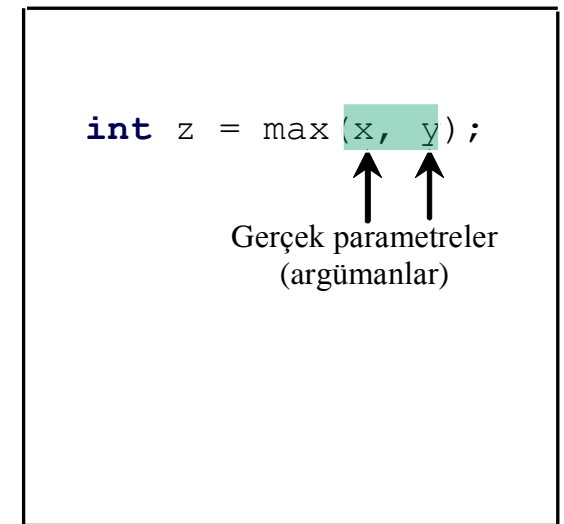
Gerçek parametreler

Bir metod çağrıldığında parametreye bir değer gönderilir. Bu değer argüman veya gerçek parametre olarak isimlendirilir.

Metodun tanımlanması



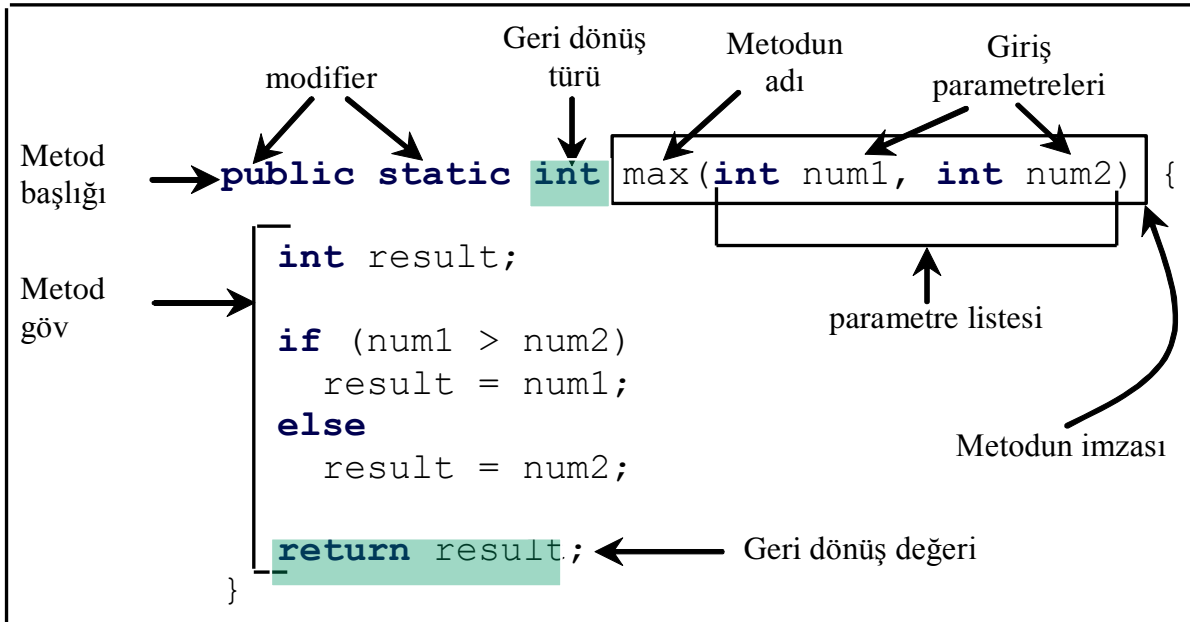
Metodun çağırılması



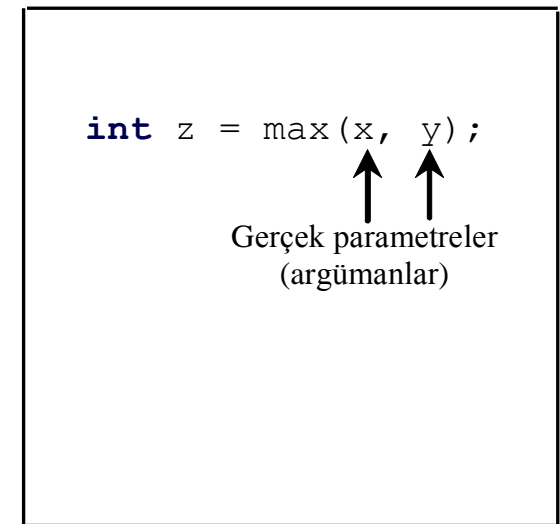
Geri dönüş değeri tipi

Bir metod herhangi bir değer geri döndürebilir. Geri dönüş değerinin tipi metodun return ile geri gönderdiği değer tipidir. Eğer method bir değer geri göndermiyorsa geri dönüş tipi void olur. Örneğin main metodunun geri dönüş türü void'tir.

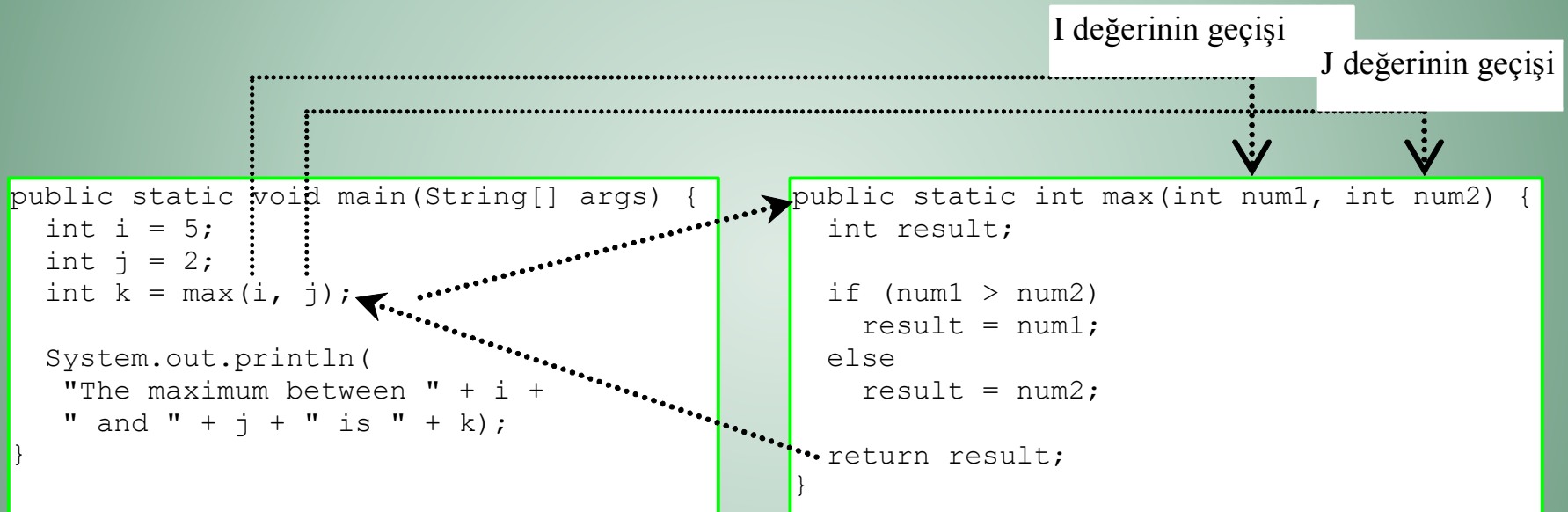
Metodun tanımlanması



Metodun çağırılması



Metotların çağrılması



Metodun çağrılması

i şimdi 5

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

j şimdi 2

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

max(i, j) metodunu çağır

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

max(i, j) metodunu çağır
i değerini num1'e aktar
j değerini num2'ye aktar

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

Result degiskenini tanımla

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```


Trace Method Invocation

(num1 > num2) şartı doğru. Çünkü num1=5 ve num2=2 dir.

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

result şimdi 5

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

Result değerini ana programa gönder

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

max(i, j) metodundan dönen
değeri k değişkenine ata

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Metodun çağrılması

Execute the print statement

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

DİKKAT

Değer döndüren bir metotta return komutu olmak zorundadır. Aşağıda (a)'da verilen metot mantıksal olarak doğru olmasına rağmen derleme hatası verir çünkü java ilgili metodun herhna because the Java compiler thinks it possible that this method does not return any value.

```
public static int sign(int n) {  
    if (n > 0)  
        return 1;  
    else if (n == 0)  
        return 0;  
    else if (n < 0)  
        return -1;  
}
```

(a)

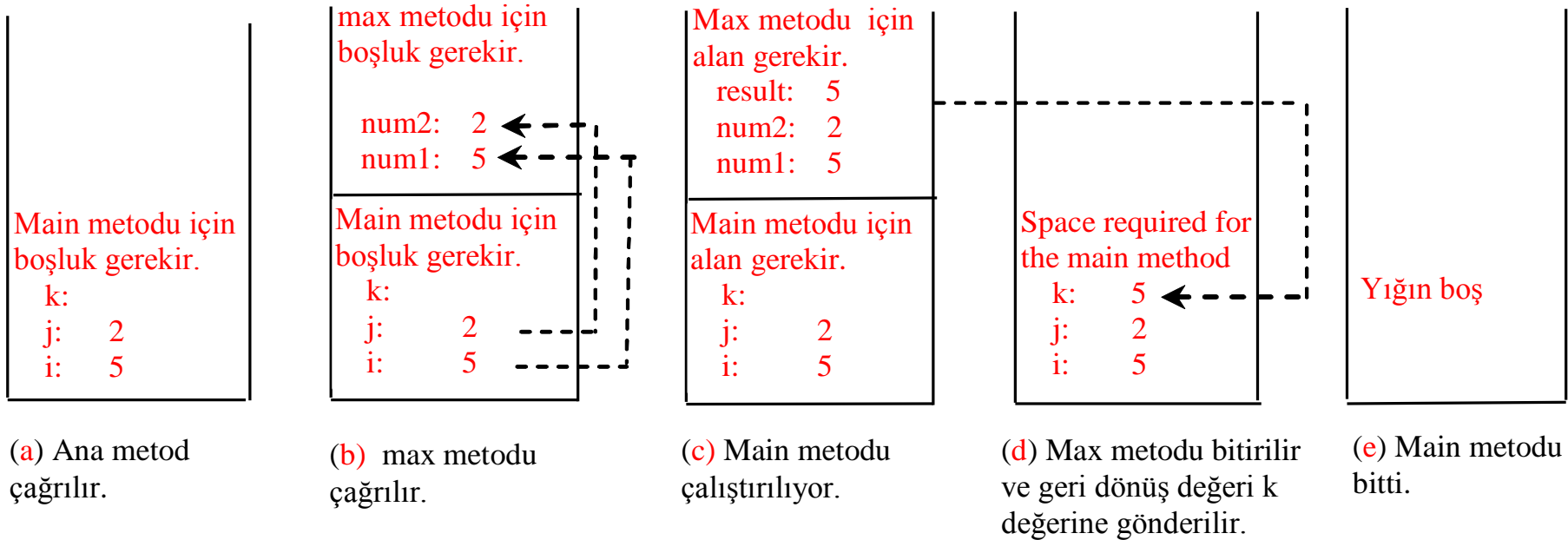
Should be

```
public static int sign(int n) {  
    if (n > 0)  
        return 1;  
    else if (n == 0)  
        return 0;  
    else  
        return -1;  
}
```

(b)

To fix this problem, delete if ($n < 0$) in (a), so that the compiler will see a return statement to be reached regardless of how the if statement is evaluated.

Yığın Çağrıları

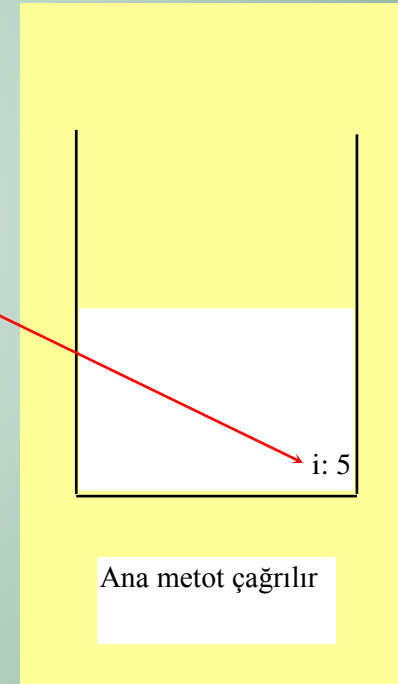


Yığın çağırılması örnek

i tanımlanır ve başlangıç değeri atanır.

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

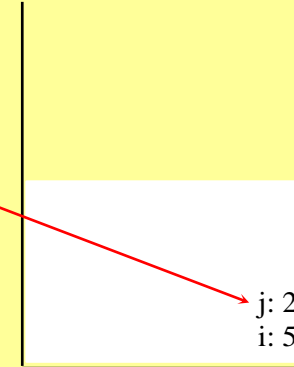


Yığın çağırılması örnek

j i tanımlanır ve değeri atanır.

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```



Main metodu
çağrılır

Yığın çağırılması örnek

k değerinin tanımlanması

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

main metodu için alan
gerekir

k:
j: 2
i: 5

main metodu
çağrılır.

Yığın çağırılması örnek

max(i, j) çağrılır

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

Main metodu için alan
gerekir.

k:
j: 2
i: 5

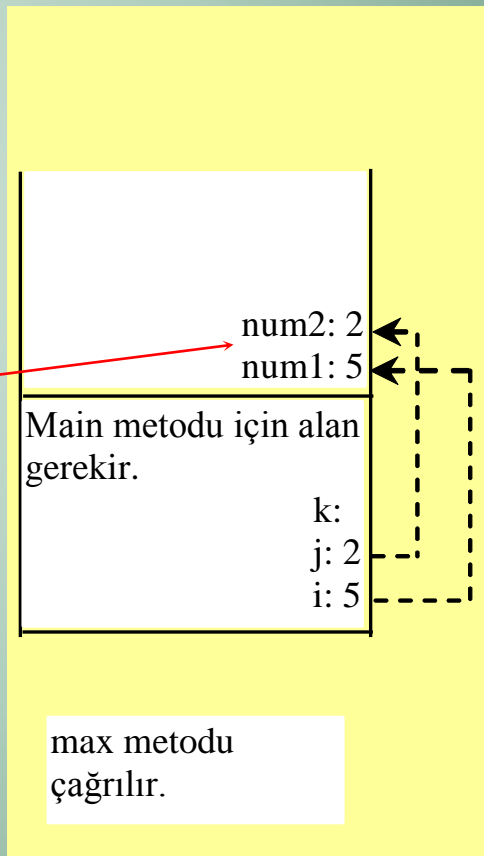
main metodu
çağrılır.

Yığın çağırılması örnek

i değerini num1'e j değerini num2'ye aktar.

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

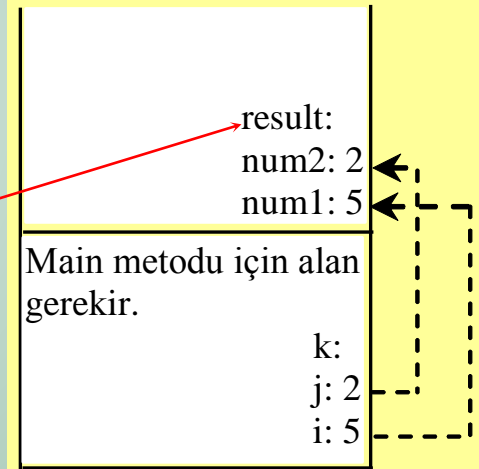


Yığın çağırılması örnek

i ve j değerlerini num1 ve num2'ye ata

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```



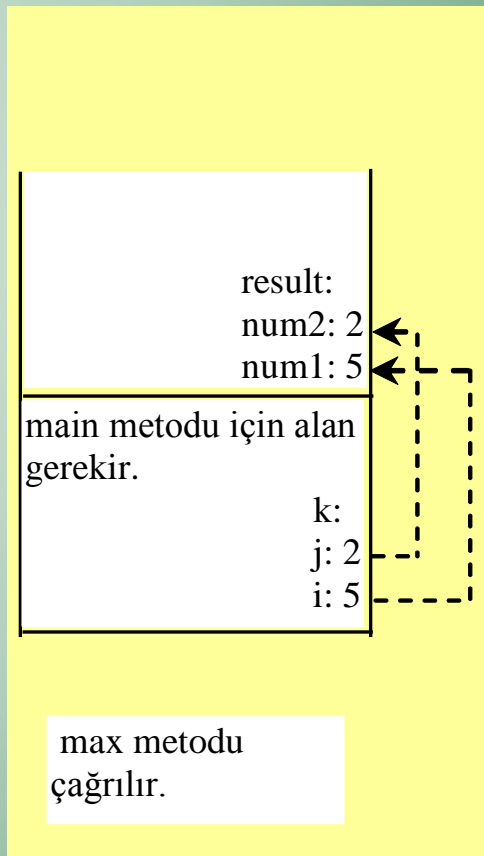
max metodu
çağrılır.

Yığın çağırılması örnek

(num1 > num2) true değeri alır

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```



Yığın çağırılması örnek

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2)  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

num1 deki değeri result
değişkenine ata

max metodu için alan
gerekir.

result: 5
num2: 2
num1: 5

Main metodu için alan
gerekir.

k:
j: 2
i: 5

max metodu
çağrılır

Yığın çağırılması örnek

result değerini geri döndür ve k'ya
ata

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2)  
{  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

max metodu için alan
gerekir.

result: 5
num2: 2
num1: 5

Main metodu için alan
gerekir.

k: 5
j: 2
i: 5

The max method is
invoked.

Yığın çağırılması örnek

print komutunu çalıştır

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

main metodu için alan
gerekir.

k:5
j: 2
i: 5

main metodu
çağrılır.

Void metodu

```
public class TestVoidMethod {  
    public static void main(String[] args)  
    {  
        System.out.print("The grade is ");  
        printGrade(78.5);  
        System.out.print("The grade is ");  
        printGrade(59.5);  
  
    }  
    public static void printGrade(double score)  
    {  
        if (score >= 90.0)    System.out.println('A');  
        else if (score >= 80.0) System.out.println('B');  
        else if (score >= 70.0) System.out.println('C');  
        else if (score >= 60.0) System.out.println('D');  
        else (System.out.println('F');  
  
    }  
}
```

Parametre Geçişi

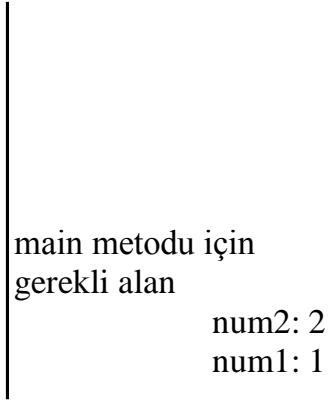
```
public static void nPrintln(String message, int n) {  
    for (int i = 0; i < n; i++)  
        System.out.println(message);  
}
```

nPrintln("Welcome to Java", 5); şeklinde metot çağrıldığında çıkış ne olur?

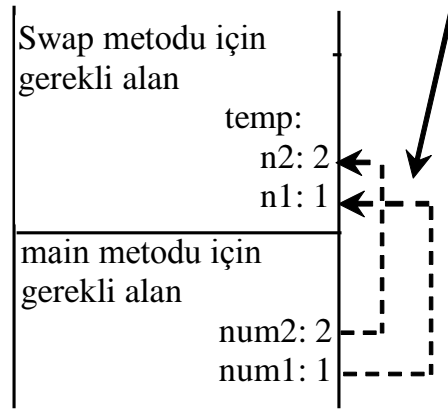
nPrintln("Computer Science", 15); şeklinde metot çağrıldığında çıkış ne olur?

Parametre Geçişi

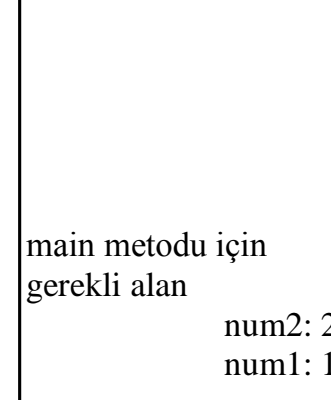
num1 ve num2 değerleri n1 ve n2'ye atılır.
Yer değiştirmenin çalıştırılması num1 ve num2'yi etkilemez.



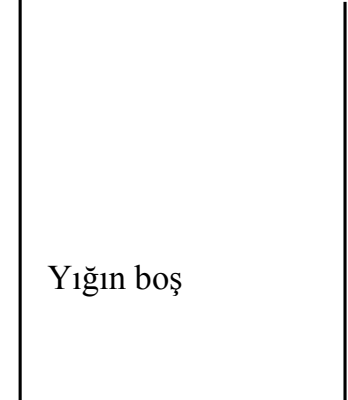
Main metodu
çağrılır.



swap metodu
çağrılır.



Swap metodu
bitirilir.



Main metodu
bitirilir.

Lokal değişkenlerin faaliyet alanı

Lokal değişken: bir metot içinde tanımlanan değişkendir.

Faaliyet alanı: değişkenin ulaşılabilirdiđi alan.

Lokal bir değişkenin faaliyet alanı tanımlandığı yerden başlar ve değişkeni içeren bloğun sonuna kadar devam eder. Lokal bir değişken kullanılmadan önce tanımlanmalıdır.

Lokal değişkenin faaliyet alanı

İç içe olmayan bloklarda bir değişkeni aynı isimle birkaç kez tanımlayabilirsiniz. Fakat içiçe bloklarda bir değişkeni aynı isimle iki kez tanımlayamazsınız.


Lokal değişkenin faaliyet alanı

Bir for döngüsü başlığının başlangıç atamasında tanımlanan bir değişken bütün döngüde bir faaliyet alanına sahiptir. Fakat bloğun herhangi bir noktasında tanımlanan bir değişken tanımlandığı yerden bloğun sonuna kadar bir faaliyet alanına sahiptir.

```
public static void method1() {  
    .  
    .  
    for (int i = 1; i < 10; i++) {  
        .  
        .  
        int j;  
        .  
        .  
        .  
    }  
}
```

The scope of i →

The scope of j →



The diagram illustrates the scope of variables in the provided code. A vertical line marks the start of the for loop body. An arrow labeled 'The scope of i' points to this line, indicating that variable i is in scope from the start of the for loop to the end of the method. Another vertical line marks the start of the block containing 'int j;'. An arrow labeled 'The scope of j' points to this line, indicating that variable j is in scope from its declaration to the end of the for loop body.

Lokal değişkenlerin faaliyet alanı

İç içe olmayan iki blok içerisinde i değişkeni tanımlanmıştır.

```
public static void method1() {  
    int x = 1;  
    int y = 1;  
    [ for (int i = 1; i < 10; i++) {  
        x += i;  
    }  
    [ for (int i = 1; i < 10; i++) {  
        y += i;  
    }  
}
```

İç içe iki blokta aynı i değişkeni Tanımlanamaz.

```
public static void method2() {  
    [ int i = 1;  
    int sum = 0;  
    [ for (int i = 1; i < 10; i++)  
        sum += i;  
    ]  
}
```


Lokal değişkenin faaliyet alanı

```
// Hata yok
public static void correctMethod() {
    int x = 1;
    int y = 1;
    // i tanımlandı
    for (int i = 1; i < 10; i++) {
        x += i;
    }
    // i tekrar tanımlandı
    for (int i = 1; i < 10; i++) {
        y += i;
    }
}
```

Lokal değişkenin faaliyet alanı

```
// hata yok
public static void incorrectMethod() {
    int x = 1;
    int y = 1;
    for (int i = 1; i < 10; i++) {
        int x = 0;
        x += i;
    }
}
```


Trigonometrik Metotlar

- ☞ `sin(double a)`
- ☞ `cos(double a)`
- ☞ `tan(double a)`
- ☞ `acos(double a)`
- ☞ `asin(double a)`
- ☞ `atan(double a)`

Radians

`toRadians(90)`

Örnekler:

```
Math.sin(0) returns 0.0
```

```
Math.sin(Math.PI / 6)  
returns 0.5
```

```
Math.sin(Math.PI / 2)  
returns 1.0
```

```
Math.cos(0) returns 1.0
```

```
Math.cos(Math.PI / 6)  
returns 0.866
```

```
Math.cos(Math.PI / 2)  
returns 0
```

Üstel metotlar

- ☞ `exp(double a)`
e^a değerini döndürür.
- ☞ `log(double a)`
a değerinin doğal logaritması.
- ☞ `log10(double a)`
a'nın 10-tabanında logaritmasını döndürür.
- ☞ `pow(double a, double b)`
a^b değerini döndürür.
- ☞ `sqrt(double a)`
a değerinin karekökünü hesaplar.

Examples:

```
Math.exp(1) returns 2.71
```

```
Math.log(2.71) returns 1.0
```

```
Math.pow(2, 3) returns 8.0
```

```
Math.pow(3, 2) returns 9.0
```

```
Math.pow(3.5, 2.5) returns  
22.91765
```

```
Math.sqrt(4) returns 2.0
```

```
Math.sqrt(10.5) returns 3.24
```

Yuvarlama Metotları

☞ `double ceil(double x)`

x en yakın büyük tam sayıya yuvarlanır. Bu tam sayı double olarak döndürülür.

☞ `double floor(double x)`

x en yakın küçük tam sayıya yuvarlanır. Bu değer double olarak döndürülür.

☞ `double rint(double x)`

x en yakın tamsayıya yuvarlanır. Eğer x iki tamsayıya yakın olarak eşit ise çift olan döndürülür.

☞ `int round(float x)`

`Return (int)Math.floor(x+0.5).`

☞ `long round(double x)`

`Return (long)Math.floor(x+0.5).`

Rounding Methods Examples

```
Math.ceil(2.1) returns 3.0
Math.ceil(2.0) returns 2.0
Math.ceil(-2.0) returns -2.0
Math.ceil(-2.1) returns -2.0
Math.floor(2.1) returns 2.0
Math.floor(2.0) returns 2.0
Math.floor(-2.0) returns -2.0
Math.floor(-2.1) returns -3.0
Math rint(2.1) returns 2.0
Math rint(2.0) returns 2.0
Math rint(-2.0) returns -2.0
Math rint(-2.1) returns -2.0
Math rint(2.5) returns 2.0
Math rint(-2.5) returns -2.0
Math.round(2.6f) returns 3
Math.round(2.0) returns 2
Math.round(-2.0f) returns -2
Math.round(-2.6) returns -3
```


min, max, and abs

☞ `max(a, b)` and `min(a, b)`

İki değerden maksimum veya minimumu döndürür.

☞ `abs(a)`

Parametrenin mutlak değerini döndürür.

☞ `random()`

`[0.0, 1.0)` aralığında double bir değer döndürür.

Örnekler:

```
Math.max(2, 3) returns 3
```

```
Math.max(2.5, 3) returns  
3.0
```

```
Math.min(2.5, 3.6)  
returns 2.5
```

```
Math.abs(-2) returns 2
```

```
Math.abs(-2.1) returns  
2.1
```


random Metodu

0.0 dan büyük eşit ve 1.0 dan küçük rastgele bir double üretir. ($0 \leq \text{Math.random()} < 1.0$).

Örnekler:

`(int) (Math.random() * 10)` → 0 ve 9 arasında rastgele bir tam sayı döndürür.

`50 + (int) (Math.random() * 50)` → 50 ile 99 arasında rastgele bir tamsayı döndürür.

Genelde,

`a + Math.random() * b` → a ile a+b arasında rastgele bir tamsayı üretir.