

Java Programlama ve Metotlar

10. Hafta

Dr. Öğr. Üyesi BÜŞRA ÖZDENİZCİ KÖSE

İşletme Bölümü

İşletme Fakültesi

Metotlar (Fonksiyonlar)

- 1'den 10'a, 20'den 37'ye, 35'den 49'a kadar olan sayıların toplamını hesaplayan bir program yazalım.
- Nasıl bir yöntemle yapardınız?
- Farklı değerlerle 3 kere döngü yapısı kullanarak mı ?

```
int sum = 0;  
for (int i = 1; i <= 10; i++)  
    sum += i;  
System.out.println("Sum from 1 to 10 is " + sum);
```

```
sum = 0;  
for (int i = 20; i <= 37; i++)  
    sum += i;  
System.out.println("Sum from 20 to 37 is " + sum);
```

```
sum = 0;  
for (int i = 35; i <= 49; i++)  
    sum += i;  
System.out.println("Sum from 35 to 49 is " + sum);
```

Bir kere benzer kodu
veya işlemi
gerçekleştirebilen
bir yapı kullanıksak
nasıl olurdu?

Toplama işlemi yapan bir metot oluşturursak

DO IT
NOW!

The screenshot shows a Java code editor with the following code:

```
1 package toplayalim;
2 public class Toplayalim {
3
4     public static int toplama (int a, int b) {
5         int toplam=0;
6         for (int i=a; i<b; i++) {
7             toplam=toplam+i;
8         }
9
10        return toplam;
11    }
12
13
14    public static void main(String[] args) {
15
16        int sonucl= toplama (1, 10);
17        int sonuc2= toplama (20, 37);
18        int sonuc3= toplama (35, 49);
19
20        System.out.println ( " 1 ile 10 arasindaki sayilarin toplami " + sonucl);
21        System.out.println ( " 20 ile 37 arasindaki sayilarin toplami " + sonuc2);
22        System.out.println ( " 35 ile 49 arasindaki sayilarin toplami " + sonuc3);
23    }
24
25 }
```

The code defines a class named `Toplayalim` with a static method `toplama` that takes two integers as parameters and returns their sum. The `main` method demonstrates this by calculating the sum of integer pairs (1, 10), (20, 37), and (35, 49) and printing the results.



The screenshot shows a Java code editor with the following code:

```
1 package toplayalim;
2 public class Toplayalim {
3
4     public static void main(String[] args) {
5
6         System.out.println ( " 1 ile 10 arasındaki sayıların toplamı " + toplama (1, 10) );
7         System.out.println ( " 20 ile 37 arasındaki sayıların toplamı " + toplama (20, 37) );
8         System.out.println ( " 35 ile 49 arasındaki sayıların toplamı " + toplama (35, 49) );
9     }
10
11
12     public static int toplama (int a, int b) {
13         int toplam=0;
14         for (int i=a; i<b; i++) {
15             toplam=toplam+i;
16         }
17         return toplam;
18     }
19
20 }
```

Annotations on the right side of the code:

- An arrow points from the word **toplama** in the **Metot Tanımlama** section to the **Metot Çağırma** section.
- An annotation labeled **Metot Çağırma** points to the three `System.out.println` statements where the `toplama` method is being called.
- An annotation labeled **Metot Tanımlama** points to the `public static int toplama (int a, int b)` method definition.

DO IT NOW

Metot Çağırma

Metot Tanımlama

Akış Kesme (Interrupt)

```
public static void main( String[] args )
{
    byte toplam;

    toplam = topla( 1, 5 ); ← (1, 5)

    System.out.println( "Toplam=" + toplam );
}
```

(15)

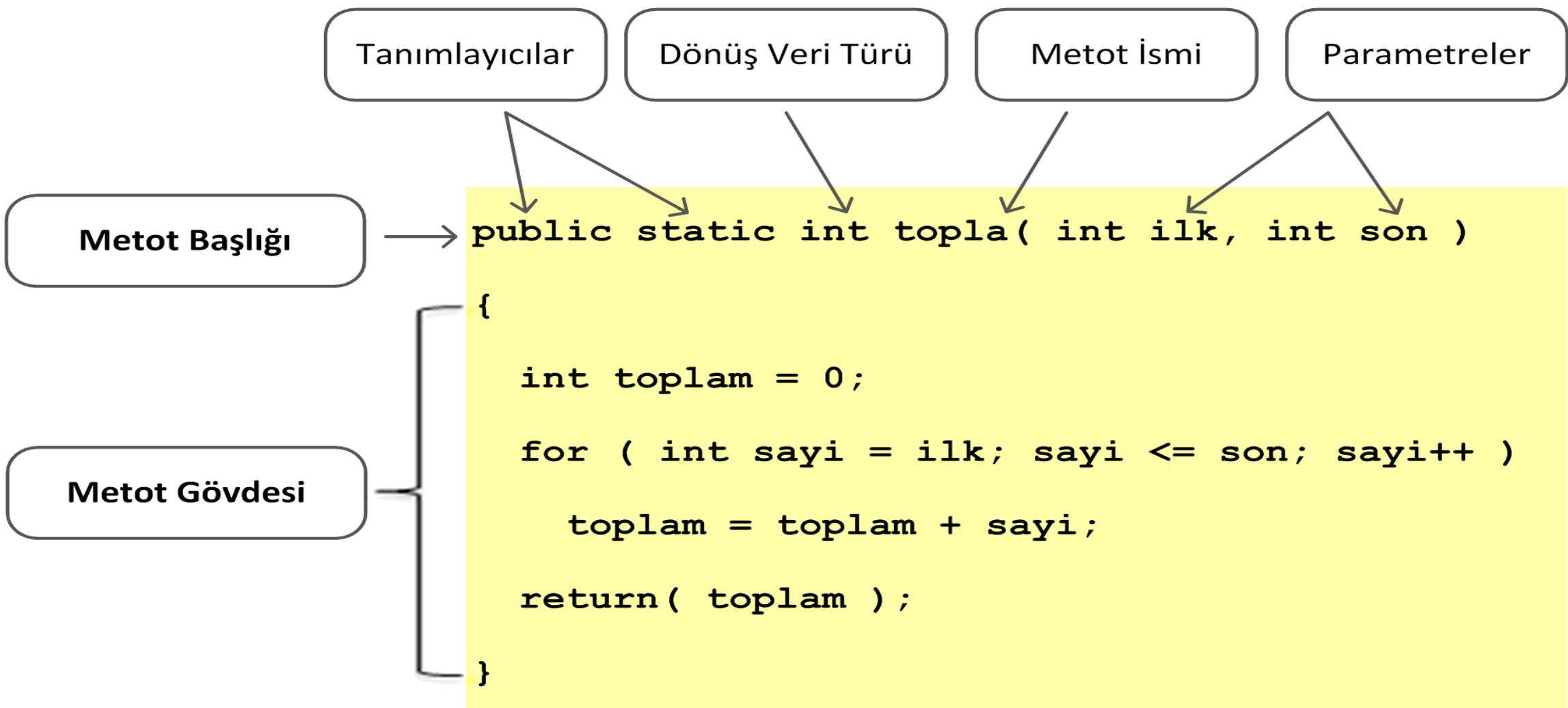
```
byte topla( byte ilk, byte son ) ← (15)
{
    byte toplam = 0;

    for ( int sayi = ilk; sayi <= son; sayi++ )
    {
        toplam = toplam + sayi;
    }

    return( toplam );
}
```

Metot Yapısı

```
tanimlayici(lar)  donus_turu  metot_ismi  ( parametre{ler} )  
{  
    metot_gövdesi  
}  
}
```



Metot Dönüş Türleri

- **return** komutu ile bir veri tipi (int, double, float, long, byte vb) döndürebilir;

veya

- Bir veri döndürmez ve programı kendi sonlandırır; **void** bir metot olur; **main()** metoduna benzer şekilde !

Output - ToplamHesaplayalim (run) X ToplamHesaplayalim.java X

Source History | Back Forward Find Save Open Project Build Run Stop Break | Minimize Maximize Close

```
1
2 package toplamhesaplayalim;
3 public class ToplamHesaplayalim {
4
5     public static void main(String[] args) {
6
7         toplama (1, 10) ;
8         toplama (20, 37) ;
9         toplama (35, 49) ;
10    }
11
12    public static void toplama (int a, int b) {
13        int toplam=0;
14        for (int i=a; i<b; i++) {
15            toplam=toplam+i;
16        }
17        System.out.println ( a + " ile " + b + " arasindaki sayilarin toplami "+ toplam);
18    }
19
20 }
```

Output - ToplamHesaplayalim (run) X ToplamHesaplayalim.java X

run:

```
1 ile 10 arasindaki sayilarin toplami 45
20 ile 37 arasindaki sayilarin toplami 476
35 ile 49 arasindaki sayilarin toplami 581
BUILD SUCCESSFUL (total time: 0 seconds)
```

Metot İsimleri Nasıl Olmalı?

Örnek

hesapla
isimBul

Değerlendirme

Doğru

Hesapla
Topla

İlk harf küçük olmalı

çarp

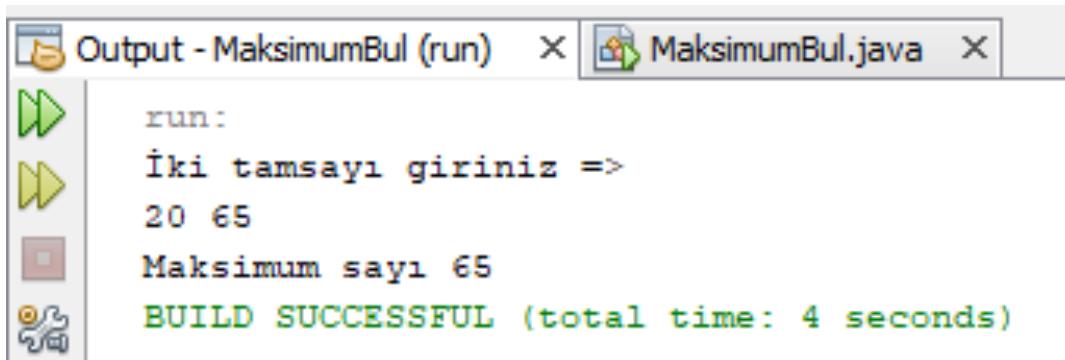
Türkçe karakter kullanılamaz

İsimBul

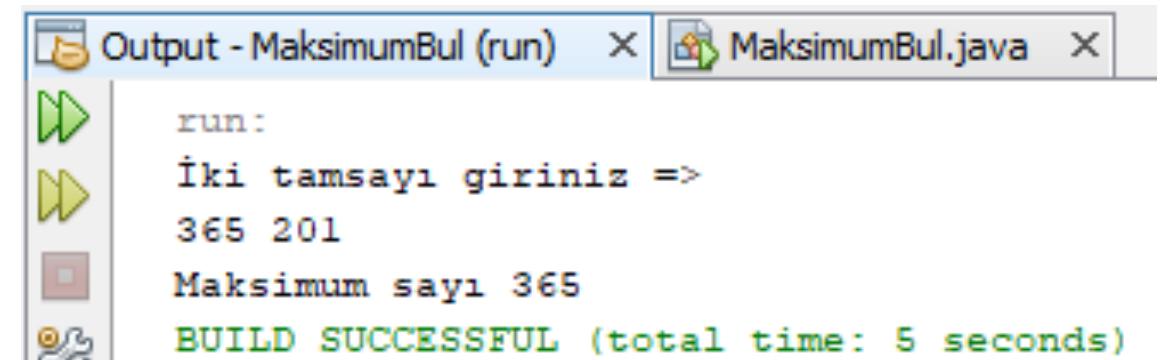
İlk harf küçük olmalı

MaksimumBul.java

- İki tamsayıdan büyük olanı hesaplayan ve sonucu geri döndüren bir **maksimumHesapla()** metodu oluşturunuz.
- Kullanıcıdan iki tamsayı okuyan ve **maksimumHesapla()** metodunu çağrıran **main()** test programını da geliştiriniz.



```
Output - MaksimumBul (run) X MaksimumBul.java X
run:
İki tamsayı giriniz =>
20 65
Maksimum sayı 65
BUILD SUCCESSFUL (total time: 4 seconds)
```



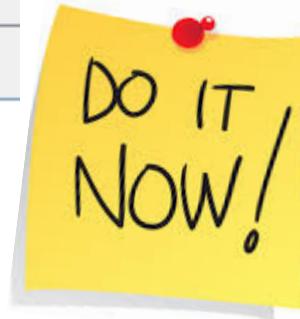
```
Output - MaksimumBul (run) X MaksimumBul.java X
run:
İki tamsayı giriniz =>
365 201
Maksimum sayı 365
BUILD SUCCESSFUL (total time: 5 seconds)
```

The screenshot shows a Java IDE interface with the following details:

- Title Bar:** Output - MaksimumBul (run) > MaksimumBul.java
- Toolbar:** Includes icons for Source, History, and various file operations like Open, Save, and Print.
- Code Editor:** Displays the Java code for 'MaksimumBul.java'. The code defines a class 'MaksimumBul' with a main method that reads two integers from the user and prints the maximum value. It also contains a helper method 'maksimumHesapla' that returns the maximum of two integers.

```
1 package maksimumbul;
2 import java.util.Scanner;
3 public class MaksimumBul {
4
5     public static void main(String[] args) {
6
7         Scanner busra=new Scanner (System.in);
8         System.out.println ("İki tamsayı giriniz => ");
9         int a= busra.nextInt();
10        int b= busra.nextInt();
11
12        int sonuc = maksimumHesapla (a, b); //metot çağrırdım ve sonucu alıyorum
13
14        System.out.println ("Maksimum sayı " + sonuc);
15
16    }
17
18
19    public static int maksimumHesapla (int num1, int num2) {
20
21        int buyuk;
22
23        if (num1 > num2)
24            buyuk = num1;
25        else
26            buyuk=num2;
27
28        return buyuk;
29
30    }
31 }
```

**DO IT
NOW!**



The screenshot shows a Java code editor with the following code:

```
1 package maksimumbul;
2 import java.util.Scanner;
3 public class MaksimumBul {
4
5     public static void main(String[] args) {
6
7         Scanner busra=new Scanner (System.in);
8         System.out.println ("İki tamsayı giriniz => ");
9         int a= busra.nextInt();
10        int b= busra.nextInt();
11
12        int sonuc = maksimumHesapla (a, b); //metot çağrırdım ve sonucu alıyorum
13
14        System.out.println ("Maksimum sayı " + sonuc);
15
16    }
17
18
19    public static int maksimumHesapla (int num1, int num2) {
20
21        if (num1 > num2)
22            return num1;
23        else
24            return num2;
25    }
26}
```

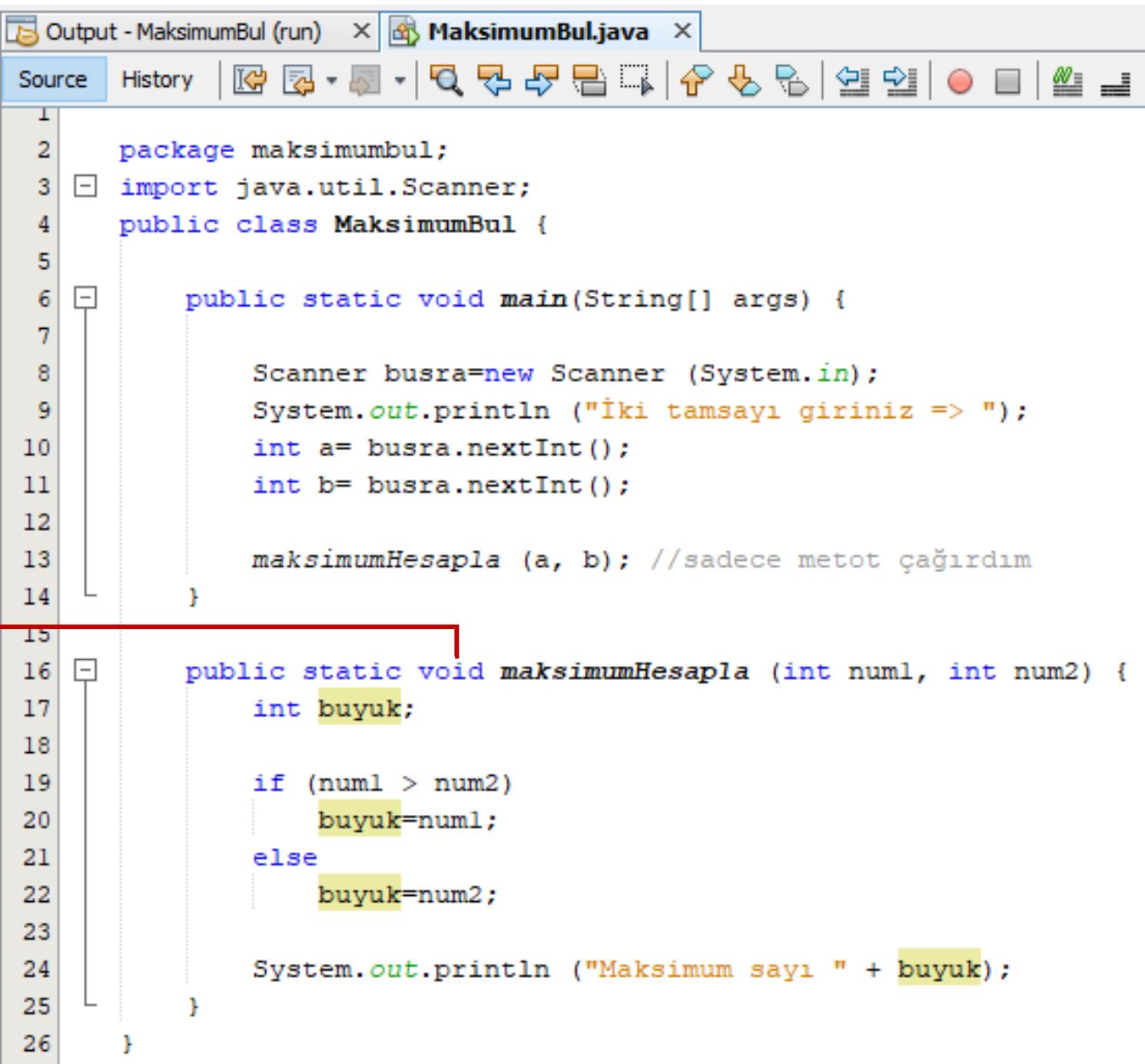
DO IT
Now!

MaksimumBul.java

- İki tamsayıdan büyük olanı hesaplayan ve sonucu ekrana yazdırın bir **maksimumHesapla()** metodunu oluşturunuz.
- Kullanıcıdan iki tamsayı okuyan ve **maksimumHesapla()** metodunu çağrıran **main()** test programını da geliştiriniz.

The screenshot shows an IDE interface with two tabs: "Output - MaksimumBul (run)" and "MaksimumBul.java". The "Output" tab displays the following text:
run:
İki tamsayı giriniz =>
20 65
Maksimum sayı 65
BUILD SUCCESSFUL (total time: 4 seconds)

The screenshot shows an IDE interface with two tabs: "Output - MaksimumBul (run)" and "MaksimumBul.java". The "Output" tab displays the following text:
run:
İki tamsayı giriniz =>
365 201
Maksimum sayı 365
BUILD SUCCESSFUL (total time: 5 seconds)



*void bir metot
olunca, bir sonuç
döndürmüyor
demektir; return
komutu
olmayacaktır !*

**DO IT
NOW!**

Return Komutu ile Döndürdiğiniz Veri Türü Önemli !

Metot

Değerlendirme

```
short topla( ... ) {  
    short x;  
    ...  
    return ( x );  
}
```

• ***Doğru***

```
boolean topla( ... ) {  
    boolean y;  
    ...  
    return ( y );  
}
```

• ***Doğru***

```
byte topla( ... ) {  
    short z;  
    ...  
    return ( z );  
}
```

• ***Error !!!***

Metot Komutlarına Örnekler

Metot	Açıklama
<code>x = y + topla(a, b);</code>	<i>Döndürülen değer ile y değişkeninin değer toplamı x değişkenine atanır.</i>
<code>int c = hangisiSecildi(k, l, m);</code>	<i>Döndürülen değer c değişkenine atanır.</i>
<code>System.out.print(ortalama(x, y, z, w));</code>	<i>Döndürülen değer ekrana yazılır.</i>
<code>boolean sonuc = dogrumuYanlismi(a, b);</code>	<i>Döndürülen değer, daha önceden tanımlanmış olduğu anlaşılan sonuc değişkenine atanır.</i>

Değişkenlerin Erişim Menzili

- Metot değişkenleri metodun içerisinde { ile } simgeleri arasında tanımlanmış değişkenleridir.
- Metot değişkenlerine o metot içinde tanımlanmış olduğu satırdan başlayarak metot sonuna kadar değer atama ve değer okuma işlemleri gerçekleştirilebilir.
- Aynı program içinde varsa başka metodlar içinden ya da başka programlar içinden o değişkenlere ulaşmak mümkün değildir.
- Metot içinde bir blok içinde tanımlanmış olan değişkenlere ise tanımlanmış olduğu satırdan başlayarak o blok sonuna kadar yer alan komutlar tarafından erişilebilir

— It is fine to declare `i` in two nonnested blocks.

```
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;  
    }  
}
```

It is wrong to declare `i` in two nested blocks.

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

FIGURE 6.6 A variable can be declared multiple times in nonnested blocks, but only once in nested blocks.

- Bir değişkene erişilebilen bölgeye, o değişkenin erişim menzili denilir.
- Değişkenin tanımlandığı yer, erişim menzilini de belirler.
- Erişim menzili, derleme esnasında dikkate alınır ve bir değişkene erişim menzili dışından erişilmeye çalışıldığında derleyici hata verecektir.



HarfNotu.java

- Midterm (%40) ve Final (%60) sınav notlarının sonucuna bağlı olarak harf notunu hesaplayan ve sonucu ekrana yazdırın bir **harfNotuHesapla()** metodu oluşturunuz.
 - 90-100: A, 80-89: B, 70-79: C, 60-69: D, 0-59: F
- Kullanıcının midterm ve final notlarını okuyan ve **harfNotuHesapla()** metodunu çağrıran **main()** test programını da geliştiriniz.

```
1 package harfnotu;
2 import java.util.Scanner;
3 public class HarfNotu {
4
5     public static void main(String[] args) {
6
7         Scanner busra= new Scanner (System.in);
8         double a, b;
9         System.out.println ("Midterm ve Final notunu giriniz => ");
10        a=busra.nextInt();
11        b=busra.nextInt();
12        harfNotuHesapla (a, b);
13    }
14
15
16    public static void harfNotuHesapla (double m, double f) {
17
18        double sonuc = (m*40/100) + (f*60/100);
19
20        if (sonuc >= 90 )
21            System.out.println ("A");
22
23        else if (sonuc >= 80 )
24            System.out.println ("B");
25
26        else if (sonuc >= 70 )
27            System.out.println ("C");
28
29        else if (sonuc >= 60 )
30            System.out.println ("D");
31
32        else
33            System.out.println ("F");
34    }
35 }
```

**DO IT
NOW!**

The screenshot shows a Java development environment with two tabs: "Output - HarfNotu (run)" and "HarfNotu.java". The "HarfNotu.java" tab displays the following code:

```
1 package harfnotu;
2 public class HarfNotu {
3
4     public static void main(String[] args) {
5
6         harfNotuHesapla (60.5, 73.8);
7         harfNotuHesapla (90.0, 68.2);
8         harfNotuHesapla (23.8, 65.0);
9     }
10
11
12     public static void harfNotuHesapla (double m, double f) {
13
14         double sonuc = (m*40/100) + (f*60/100);
15
16         System.out.print (m + " ve " + f + " notlarina sahip ogrencinin harf notu ");
17         if (sonuc >= 90 )
18             System.out.println ("A");
19
20         else if (sonuc >= 80 )
21             System.out.println ("B");
22
23         else if (sonuc >= 70 )
24             System.out.println ("C");
25
26         else if (sonuc >= 60 )
27             System.out.println ("D");
28
29         else
30             System.out.println ("F");
31     }
32 }
```

The "Output - HarfNotu (run)" tab shows the execution results:

- run:
 - 60.5 ve 73.8 notlari A
 - 90.0 ve 68.2 notlari B
 - 23.8 ve 65.0 notlari C
- BUILD SUCCESSFUL

**DO IT
NOW!**

```
Output - HarfNotu (run)  X  HarfNotu.java  X
run:
60.5 ve 73.8 notlarına sahip öğrencinin harf notu D
90.0 ve 68.2 notlarına sahip öğrencinin harf notu C
23.8 ve 65.0 notlarına sahip öğrencinin harf notu F
BUILD SUCCESSFUL (total time: 0 seconds)
```

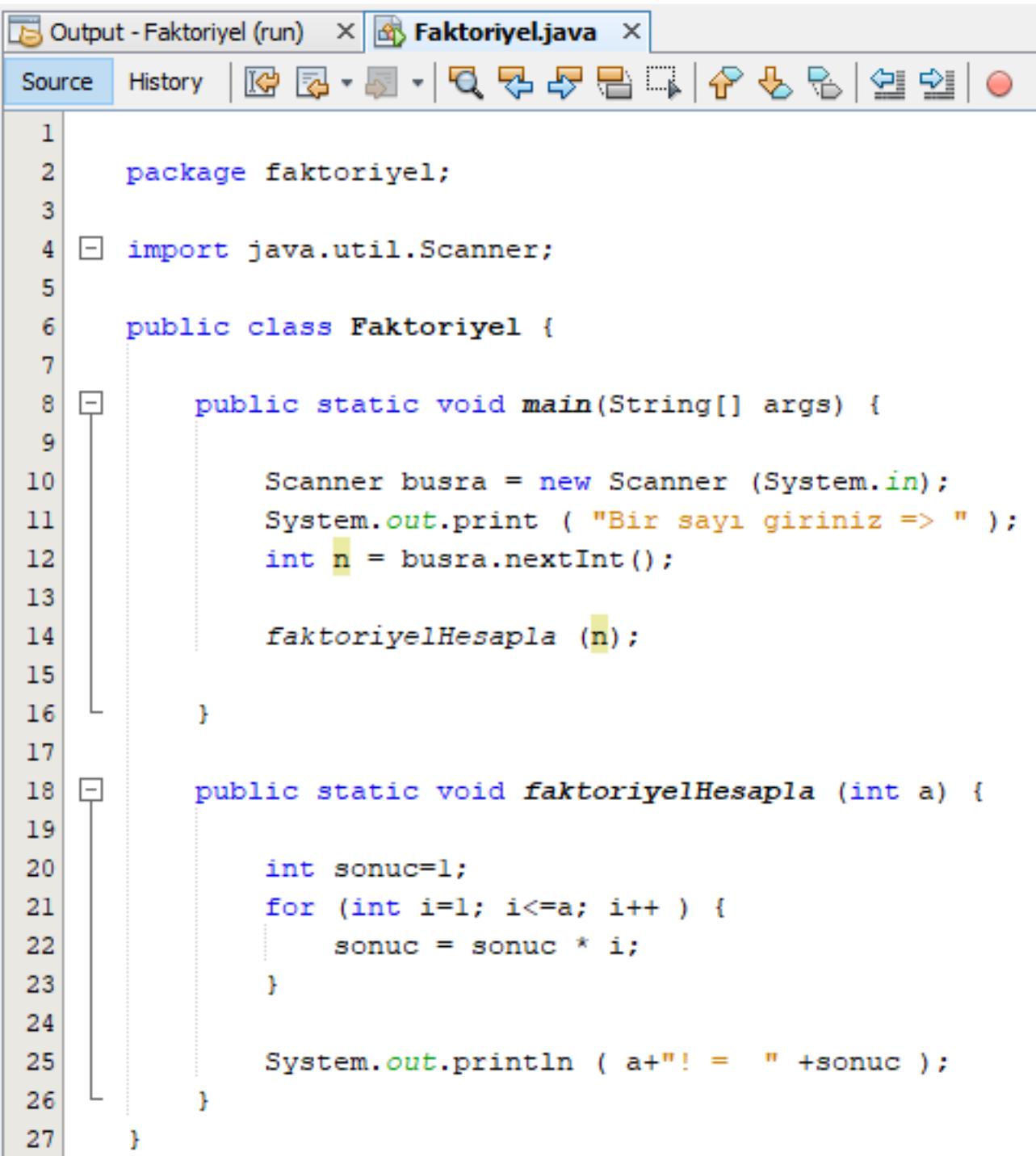
DO IT
Now!

Faktoriyel.java

- Bir sayının faktöriyel değerini hesaplayan ve sonucu ekrana yazdırın **faktoriyelHesapla()** metodunu oluşturunuz.
- Kullanıcıdan bir sayı okuyan ve **faktoriyelHesapla()** metodunu çağrıran **main()** test programını da geliştiriniz.

$$F(n) = n * (n-1) * (n-2) * \dots * 1$$

```
Output - Faktoriyel (run)
run:
Bir sayı giriniz => 6
6! = 720
BUILD SUCCESSFUL (total time: 2 seconds)
```



DO IT
NOW!



Hipotenus.java

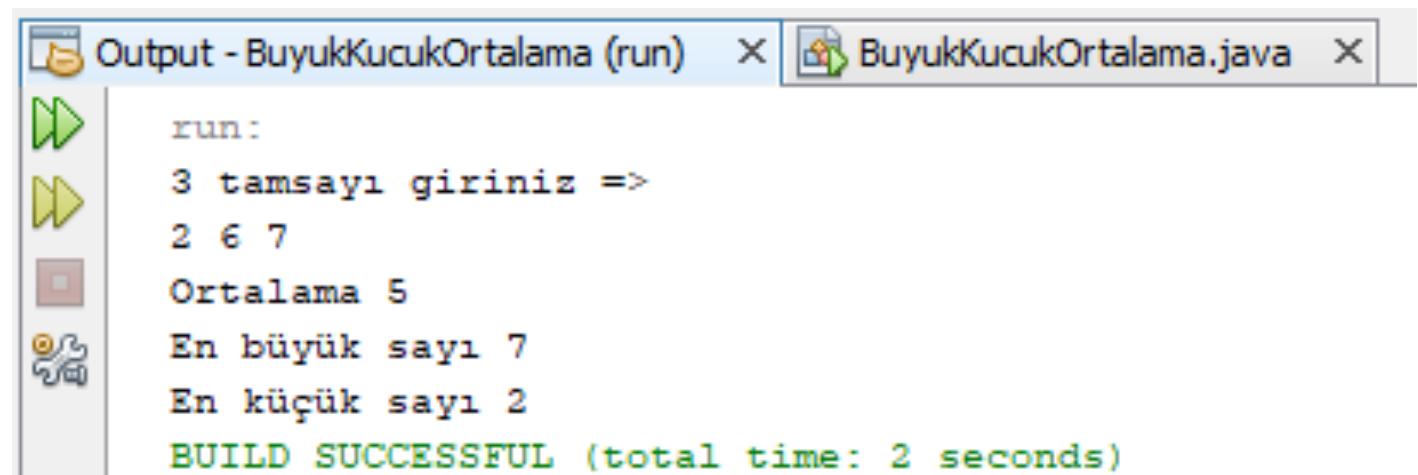
- Bir dik üçgenin hipotenüsünü hesaplayan ve sonucu döndüren **hipotenusHesapla()** metodunu oluşturunuz.
- Karekök almak için **Math.sqrt()** metodunu kullanabilirsiniz.
- Kullanıcıdan bir dik üçgenin iki kısa kenar uzunluğunu okuyan ve **hipotenusHesapla()** metodunu çağrıran **main()** test programını da geliştiriniz.

$$\text{hipotenus}^2 = \text{kenar1}^2 + \text{kenar2}^2$$

DO IT
Now!

BuyukKucukOrtalama.java

- Üç tamsayıdan büyük olanı hesaplayan ve sonucu döndüren **buyukSayi()** metodunu; küçük olanı hesaplayan ve sonucu döndüren **kucukSayi()** metodunu; üç tamsayının ortalamasını hesaplayan ve sonucu ekrana yazdırınan **ortalamaHesapla()** metodunu geliştiriniz.
- Kullanıcıdan üç tamsayı alarak ilgili fonksiyonları çağrıran **main()** test programını da geliştiriniz



```
Output - BuyukKucukOrtalama (run) X BuyukKucukOrtalama.java X
run:
3 tamsayı giriniz =>
2 6 7
Ortalama 5
En büyük sayı 7
En küçük sayı 2
BUILD SUCCESSFUL (total time: 2 seconds)
```

Java'da Özel Anlamlı Kelimeler

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Any Questions?