

Java Programlama ve Döngü Yapıları

7. Hafta

Dr. Öğr. Üyesi BÜŞRA ÖZDENİZCİ KÖSE

İşletme Bölümü

İşletme Fakültesi

Increment ve Decrement Operatörleri Hatırlayalım !

- Tamsayı değerinin 1 artırılması için normal olarak kullanılacak komut **sayı = sayı + 1** olacaktır.
- Bunun yerine **sayı++** yazdığımızda, neticesi aynı olan daha pratik bir kod elde etmiş oluruz.
- Komut içinde tek başına kullanıldığı taktirde **sayı++** ile **++sayı** aynı işleve sahiptir.
- Benzer şekilde **sayı --** ile **--sayı** komutları da aynı işleve sahiptir.

```
int i = 3, j = 3;  
  
int a = 5, b = 5;  
  
i++;           // i artık 4 (postincrement)  
  
++a;           // a artık 4 (preincrement)  
  
j--;           // j artık 2 (postdecrement)  
  
--b;           // b artık 4 (predecrement)
```

TABLE 2.5 Increment and Decrement Operators

<i>Operator</i>	<i>Name</i>	<i>Description</i>	<i>Example (assume i = 1)</i>
<code>++var</code>	preincrement	Increment <code>var</code> by <code>1</code> , and use the new <code>var</code> value in the statement	<code>int j = ++i;</code> <code>// j is 2, i is 2</code>
<code>var++</code>	postincrement	Increment <code>var</code> by <code>1</code> , but use the original <code>var</code> value in the statement	<code>int j = i++;</code> <code>// j is 1, i is 2</code>
<code>--var</code>	predecrement	Decrement <code>var</code> by <code>1</code> , and use the new <code>var</code> value in the statement	<code>int j = --i;</code> <code>// j is 0, i is 0</code>
<code>var--</code>	postdecrement	Decrement <code>var</code> by <code>1</code> , and use the original <code>var</code> value in the statement	<code>int j = i--;</code> <code>// j is 1, i is 0</code>

Fakat başka aritmetik işlemler içerisinde kullanırken dikkat !!!

Örnek 1

Komut	Ne Olur?	Açıklama
$x = 1$	$y = 1 + 1 = 2$	$x++$, x 'in değeri kullanıldıktan sonra bir artırılmasını gerektirir; bu nedenle x 'in değeri önce 1'dir.
$y = (x++) + 1$	$x = x + 1 = 2$	daha sonra x 'in değeri 2 olur.

Fakat başka aritmetik işlemler içerisinde kullanırken dikkat !!!

Örnek 2

Komut	Ne Olur?	Açıklama
$x = 1$	$x = 1 + 1 = 2$	$++x$, x 'in değeri kullanılmadan önce 1 artırılmasını gerektirir; bu nedenle x 'in değeri 2 olur.
$y = (++x) + 1$	$y = 2 + 1 = 3$	x 'in yenilenmiş değeri ile 1 toplandığında y 'nin değeri 3 olur.

Fakat başka aritmetik işlemler içerisinde kullanırken dikkat !!!

Örnek 3

Komut	Ne Olur?	Açıklama
$x = 1$	$x = 1 + 1 = 2$	$++x$, x 'in değeri kullanılmadan önce 1 artırılmasını gerektirir; bu nedenle x 'in değeri 2 olur.
$y = (++x) + x$	$y = 2 + 2 = 4$	x 'in yenilenmiş değeri iki kez kullanıldığında, y 'nin değeri 4 olur.

DO IT
Now!

```
1 package matematik;
2
3
4 import java.util.Scanner;
5
6 public class Matematik {
7
8     public static void main(String[] args) {
9
10        Scanner busra = new Scanner(System.in);
11
12        System.out.print ("Bir tamsayı giriniz: ");
13        int a = busra.nextInt();
14
15        System.out.println( a++ );
16
17        System.out.println( a );
18
19
20        System.out.print ("Bir tamsayı giriniz: ");
21        int b = busra.nextInt();
22
23        System.out.println( ++b );
24
25        System.out.println( b );
26    }
27
28 }
```

The screenshot shows the 'Output' window of an IDE for the 'Matematik' application. The window title is 'Output - Matematik (run)'. It displays the following text:

```
run:
Bir tamsayı giriniz: 3
3
4
Bir tamsayı giriniz: 5
6
6
BUILD SUCCESSFUL (total time: 3 seconds)
```

Döngü Yapıları

Neden Döngülere İhtiyaç Duyuyoruz?

- Bir yazıyı ekrana 100 kere basmanız gerektiği düşünün!
- Örneğin Welcome to Java!
- Aşağıdaki komutu 100 kere yazmak sizce mantıklı mı?

100 times {
System.out.println("Welcome to Java!");
System.out.println("Welcome to Java!");
...
System.out.println("Welcome to Java!");

Döngülere Giriş

- Java programındaki program satırları ardışık olarak ve birer birer icra edilebileceği gibi bazı komut satırlarının birden çok kez icra edilmesi gerekebilir:

```
int count = 0;
while (count < 100) {
    System.out.println("Welcome to Java!");
    count++;
}
```

Nasıl Çalışır?

- **count** isimli bir sayıç değişkeni tanımlanır ve değeri **0** olarak belirlenir. Döngü **count < 100** ifadesi **true** olup olmadığını kontrol eder. Eğer doğru ise, döngü içerisinde tanımlı komutları gerçekleştirir; ekrana **Welcome to Java!** yazar ve sayıç değişkenini **1** arttırır. Bu işlem **count < 100** komutu **false** olana kadar devam eder.
- Ne zaman ki **count < 100** ifadesi **false** olursa (yani **count** artık **100** olunca), döngü sonlanır.

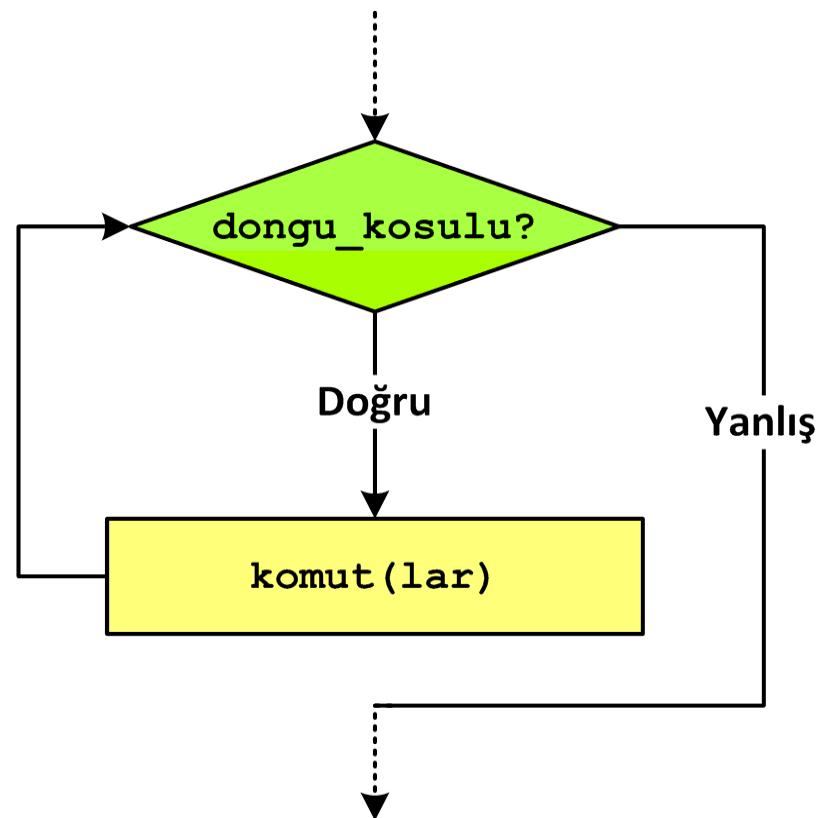
Döngü Türleri

- Döngü yapılarının Java'da önemli bir yeri vardır.
- Bir komutun ya da komut kümесinin bir koşula veya koşullara bağlı olarak tekrar edilmesini sağlayan Java döngü yapıları üç tanedir:
 - **WHILE**
 - **FOR**
 - **DO-WHILE**

WHILE Döngü Yapısı

- WHILE yapısında komut tekrarı belirli bir koşula bağlı olduğu durumda kullanılmaktadır.
- WHILE döngü yapısı:

```
while ( dongu_kosulu ) {  
    komut_1;  
    ...  
    komut_n;  
}
```



```
01 while ( dongu_kosulu )
02     komut_1;
```

Yapı 7.4. WHILE Yapısı

```
01 while ( dongu_kosulu )
02 {
03     komut_1;
04     komut_2;
05     ...
06     komut_n;
07 }
```

Yapı 7.5. WHILE Yapısında Blok Kullanımı

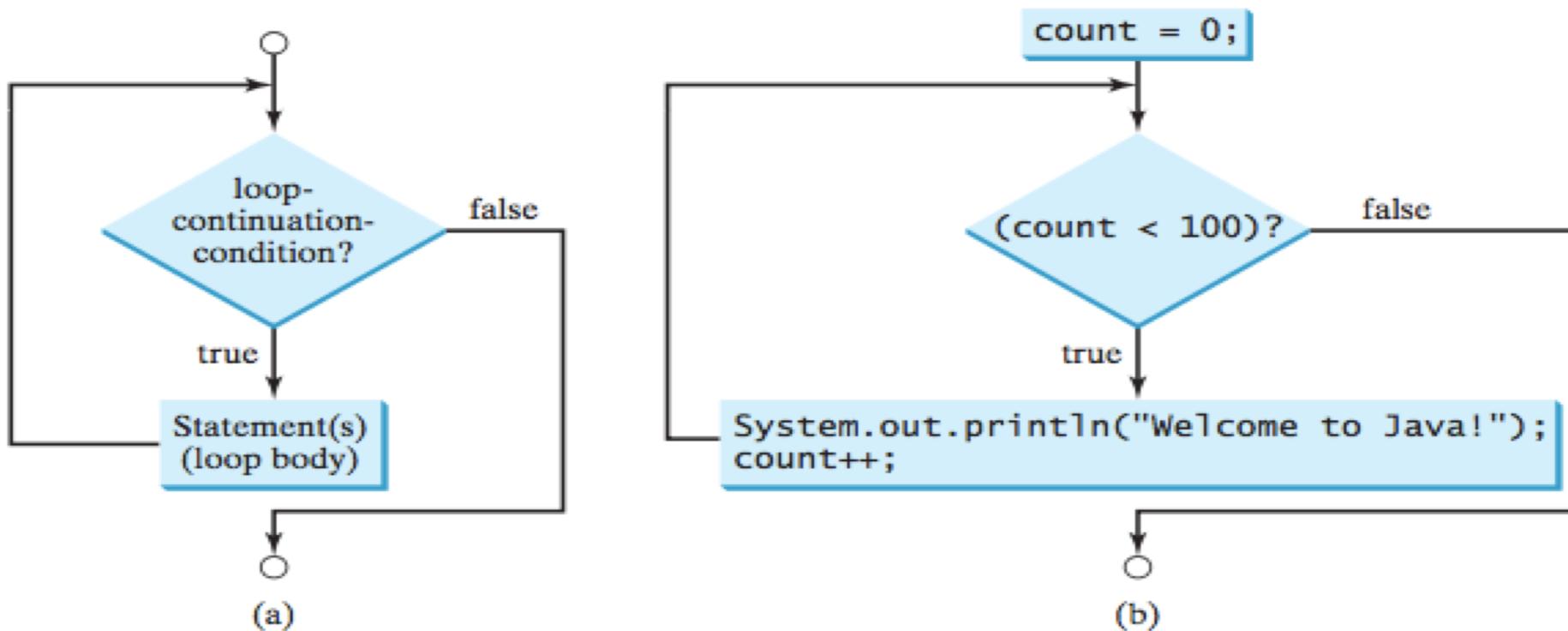


FIGURE 5.1 The `while` loop repeatedly executes the statements in the loop body when the **Loop-continuation-condition** evaluates to **true**.

```

int count = 0;
while (count < 100) {
    System.out.println("Welcome to Java!");
    count++;
}
  
```

DO IT
Now!

Yazdir.java

- 20 kere Hello World yazan bir Java programı geliştiriniz.

```
Output - Yazdir (run) X
run:
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Goodbye !
```

```
2 package yazdir;
3
4 public class Yazdir {
5
6     public static void main(String[] args) {
7
8         int sayac = 1;
9
10        while ( sayac <= 20 ) {
11
12            System.out.println ("Hello World :)");
13
14            sayac = sayac + 1;
15        }
16
17        System.out.println ("Goodbye ! ");
18    }
19}
20}
```

DO IT
Now!

Yazdir.java

- Kullanıcının istediği sayı kadar Hello World yazdırınan bir Java programı geliştiriniz.

```
Output - Yazdir (run)
run:
Kaç kere yazdırmak istiyorsun?
5
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Goodbye !
BUILD SUCCESSFUL (total time: 4 seconds)
```

```
Output - Yazdir (run)
run:
Kaç kere yazdırmak istiyorsun?
10
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Hello World :)
Goodbye !
BUILD SUCCESSFUL (total time: 3 seconds)
```

DO IT
Now!

```
2 package yazdir;
3
4 import java.util.Scanner;
5
6 public class Yazdir {
7
8     public static void main(String[] args) {
9
10         int sayac = 1;
11
12         Scanner busra = new Scanner (System.in);
13         System.out.println ("Kaç kere yazdırma istiyorsun? ");
14         int i = busra.nextInt();
15
16         while ( sayac <= i ) {
17
18             System.out.println ("Hello World :) ");
19
20             sayac = sayac + 1;
21         }
22
23         System.out.println ("Goodbye ! ");
24
25     }
26 }
```

DO IT
Now!

```
1
2     package yazdir;
3
4     [-] import java.util.Scanner;
5
6     public class Yazdir {
7
8         [-]     public static void main(String[] args) {
9
10            int sayac = 1;
11
12            Scanner busra = new Scanner (System.in);
13            System.out.println ("Kaç kere yazdırılmak istiyorsun? ");
14            int i = busra.nextInt();
15
16            while ( sayac <= i ) {
17
18                System.out.println ("Hello World :) ");
19
20                sayac++;
21            }
22
23            System.out.println ("Goodbye ! ");
24
25        }
26    }
```

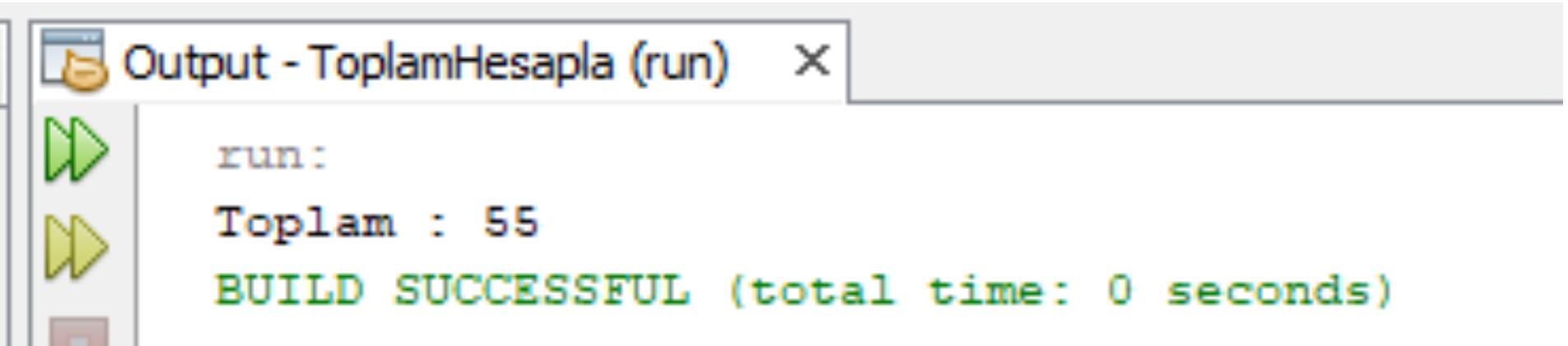
WHILE döngü yapısının genel yapısı

```
i = baslangicDeger;  
// kontrol değişkenini belirle  
  
while ( i < bitisDeger ) {  
    // true olduğu sürece çalışacak döngü için bir koşul belirle  
    // döngü komutlarını belirle  
    ...  
    i++; // kontrol değişkenini veya sayacımızı ayarla  
}
```

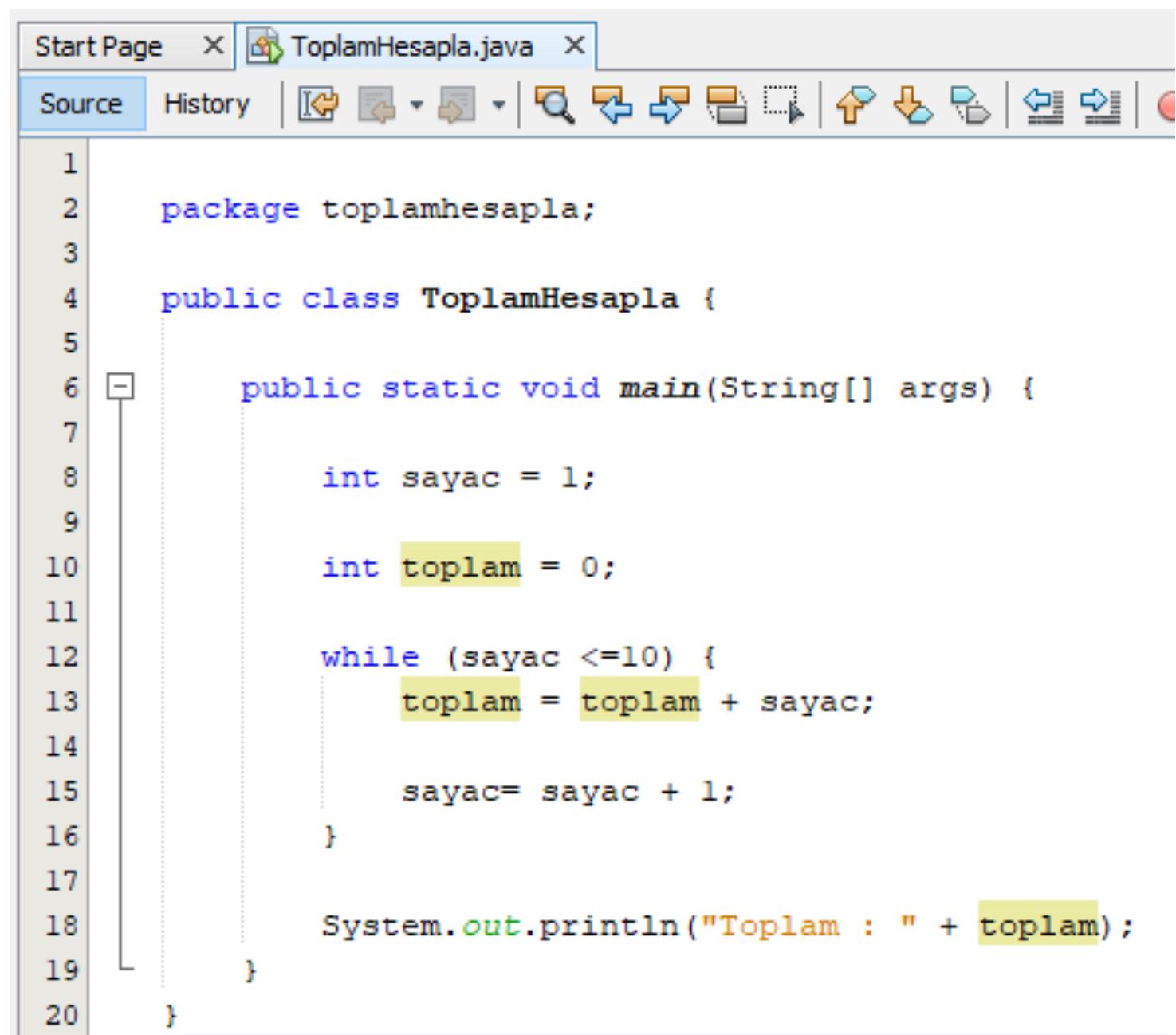
DO IT
Now!

ToplamHesapla.java

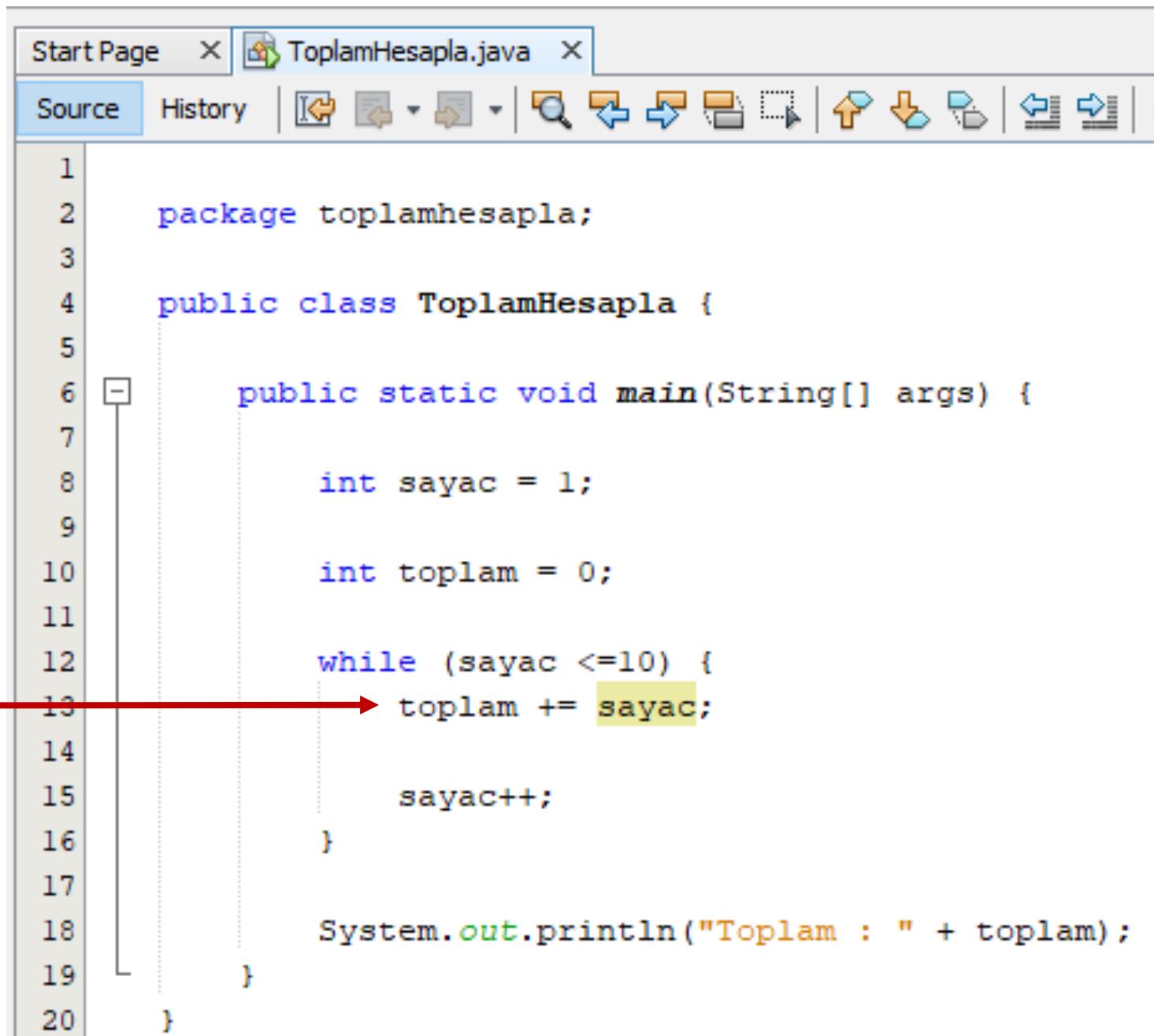
- 1'den 10'a kadar olan sayıların toplamını while döngüsü ile hesaplayan bir Java programı geliştiriniz.



```
Output - ToplamHesapla (run)
run:
Toplam : 55
BUILD SUCCESSFUL (total time: 0 seconds)
```



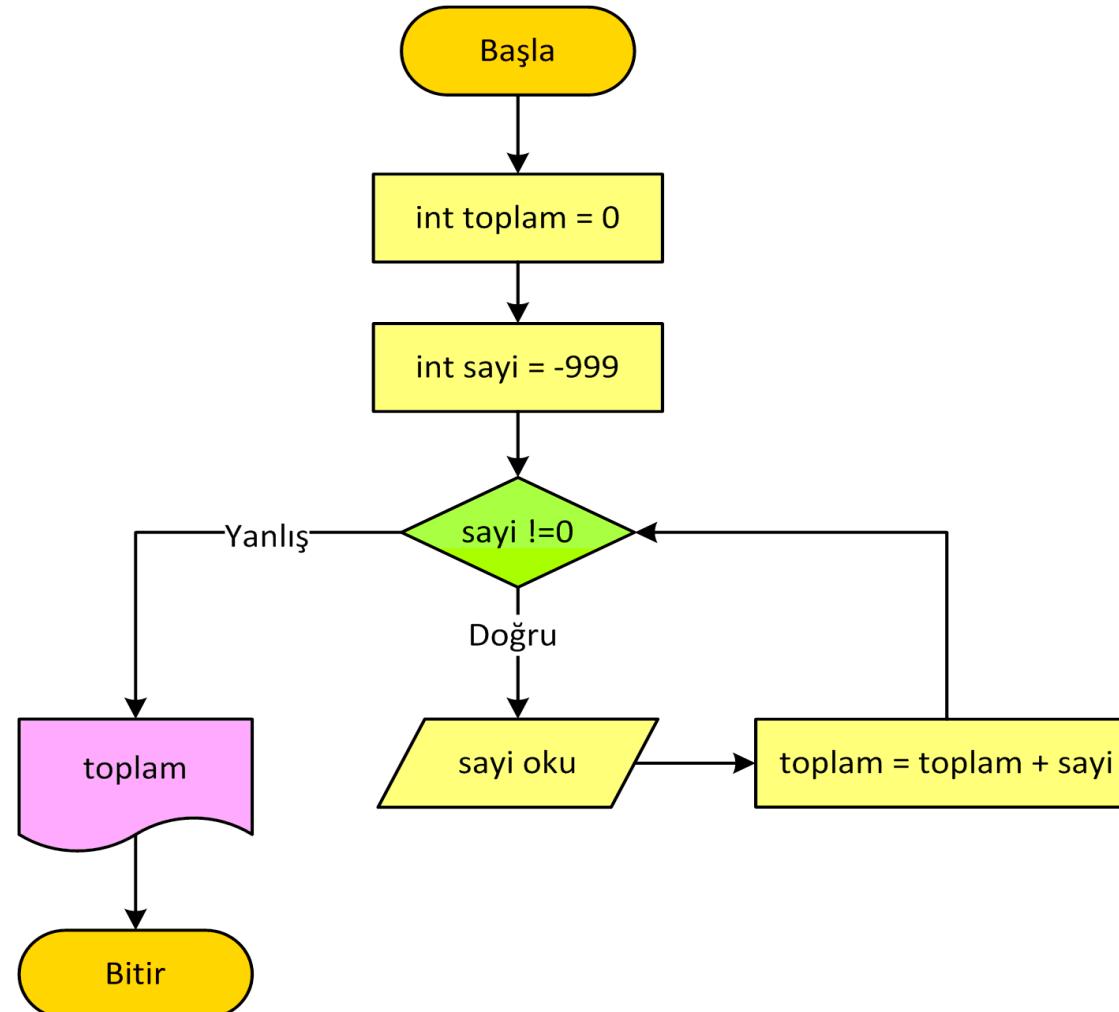
DO IT
NOW!



DO IT
NOW!

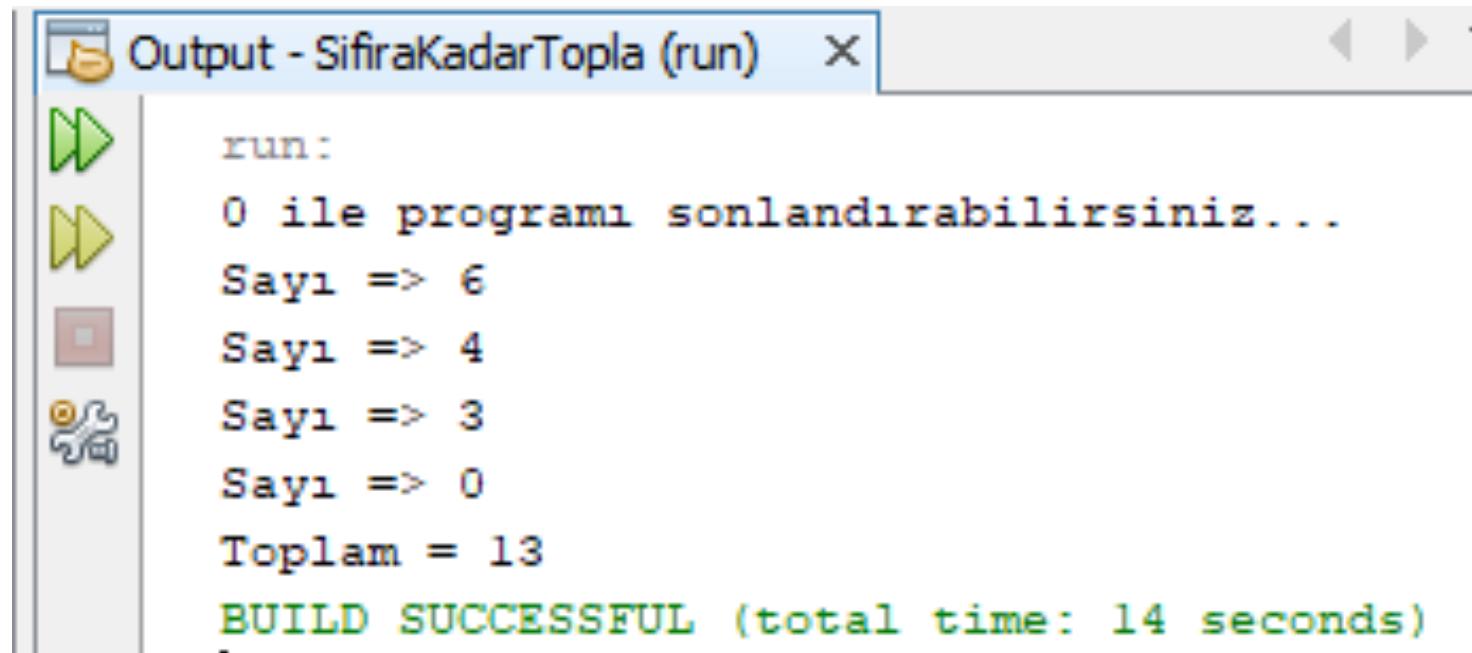
Kısayol yazım
formatlarını
kullanabilirsiniz !

Akış Diyagramını Programlayalım...



SifiraKadarTopla.java

- Kullanıcı sıfır sayısını girene kadar tüm gireceği sayıların toplamını hesaplayan bir Java programı geliştiriniz.



```
Output - SifiraKadarTopla (run) ×
run:
0 ile programı sonlandırabilirsiniz...
Sayı => 6
Sayı => 4
Sayı => 3
Sayı => 0
Toplam = 13
BUILD SUCCESSFUL (total time: 14 seconds)
```

Start Page X SifiraKadarTopla.java X

Source History | Back Forward | Search | Find | Replace | Open | Save | Print | Home | Back | Forward | Stop | Refresh | Help | Options | View | Window | Exit

```
1
2 package sifirakadartopla;
3 import java.util.Scanner;
4 public class SifiraKadarTopla {
5     public static void main(String[] args) {
6
7         Scanner input = new Scanner(System.in);
8         System.out.println( "0 ile programı sonlandırabilirsiniz... " );
9
10        int toplam = 0;
11        int sayi = -999; // 0 hariç istediğiniz değer ile döngüye girebilirsin
12
13        while ( sayi != 0 ) {
14            System.out.print( "Sayı => " );
15            sayi = input.nextInt();
16
17            toplam = toplam + sayi;
18        }
19        System.out.println( "Toplam = " + toplam );
20
21    }
22
23 }
```



```
1
2     package sifirakadartopla;
3     [-] import java.util.Scanner;
4     public class SifiraKadarTopla {
5         [-]     public static void main(String[] args) {
6
7             Scanner input = new Scanner(System.in);
8             System.out.println( "0 ile programı sonlandırabilirsiniz... " );
9
10            int toplam = 0;
11            int sayi = 1;
12
13            while ( sayi != 0 ) {
14                System.out.print( "Sayı => " );
15                sayi = input.nextInt();
16
17                toplam = toplam + sayi;
18            }
19            System.out.println( "Toplam = " + toplam );
20
21        }
22    }
23 }
```



Sonsuz WHILE Döngüsü

The screenshot shows an IDE interface with two tabs: "Start Page" and "Yazdir.java". The "Yazdir.java" tab is active, displaying the following code:

```
1 package yazdir;
2
3 public class Yazdir {
4
5     public static void main(String[] args) {
6         int sayac = 1;
7
8         while ( sayac < 20 ) {
9             System.out.println ("Hello World :)");
10            sayac++;
11        }
12    }
13}
```

The output window on the left shows the repeated output "Hello World :)" 20 times.

```
01 while( true )
02 {
03     komut_1;
04     komut_2;
05     ...
06     komut_n;
07 }
```

Yapı 7.6. WHILE Yapısında Sonsuz Döngü

Sonsuz WHILE Döngüsü

The screenshot shows a Java development environment with the following details:

- Output - Yazdir (run) window:** Displays the repeated output "Hello World :)" in the console.
- Project Explorer:** Shows files like "Start Page", "SifraKadarTopla.java", and "Yazdir.java".
- Toolbars:** Standard Java IDE toolbars for file operations, search, and navigation.
- Code Editor:** Displays the Java code for the "Yazdir" class:

```
1 package yazdir;
2
3 public class Yazdir {
4
5     public static void main(String[] args) {
6         while ( true ) {
7             System.out.println ("Hello World :)");
8         }
9     }
10}
```

The code is annotated with line numbers (1-18) and syntax highlighting.
- Status Bar:** Shows the current run configuration "Yazdir (run)" and the timestamp "18:1".

Math.random() Metodu

- **Math.random()** rastgele sayılar seçmek için kullanılan ve Math sınıfında bulunan önemli bir metotdur; 0.0 ile 1.0 arasında -1.0 hariç- rastgele double bir değer üretir.
- Örneğin, **(int)(Math.random()*10)** metodu 0 ile 9 arasında rastgele bir tamsayı değer döndürür.
- Bir başka örnek olarak **(int)(Math.random()*6+1)** metodu 1 ile 6 arasında rastgele bir tamsayı değer döndürür.
- **(int)(Math.random()*12+1)** metodu 1 ile 12 arasında rastgele bir tamsayı değer döndürür.

DO IT
Now!

Sayı Tahmin Oyunu

- Program rastgele 0 ile 20 arasında bir sayı üretecektir ve kullanıcının bu sayıyı yönlendirmeler (büyük veya küçük sayı girdiniz şeklinde) ile tahmin etmesi sağlanacaktır.
- Math.random() fonksiyonunu kullanalım...

```
Output - SayiTahminOyunu (run)
run:
Sayı Tahmin Oyununa Hoşgeldiniz...
Bir tahmin yapınız =>
6
Küçük bir değer girdiniz
Bir tahmin yapınız =>
18
Büyük bir değer girdiniz
Bir tahmin yapınız =>
15
Büyük bir değer girdiniz
Bir tahmin yapınız =>
10
Büyük bir değer girdiniz
Bir tahmin yapınız =>
9
Evet doğru ! TEBRİKLER !
BUILD SUCCESSFUL (total time: 11 seconds)
```

```
2 package sayitahminoyunu;
3 import java.util.Scanner;
4 public class SayiTahminOyunu {
5     public static void main(String[] args) {
6
7         Scanner input = new Scanner(System.in);
8
9         System.out.println ("Sayı Tahmin Oyununa Hoşgeldiniz... ");
10
11        int rastgele = (int) (Math.random () * 21); // 0-20 arasında rastgele integer üretildi
12
13        int tahmin = 21; // 0-20 arası haricinde istediğiniz değeri kullan, döngüye girelim
14
15        while (rastgele != tahmin ) {
16
17            System.out.println("Bir tahmin yapınız =>");
18            tahmin = input.nextInt();
19
20            if (tahmin > rastgele)
21                System.out.println ("Büyük bir değer girdiniz");
22            else if (tahmin < rastgele)
23                System.out.println ("Küçük bir değer girdiniz");
24            else
25                System.out.println ("Evet doğru ! TEBRİKLER ! ");
26        }
27    }
28 }
```



Yaygın Yazım Hataları ☺

```
int i = 0;
while (i < 10);
{
    System.out.println("i is " + i);
    i++;
}
```

(c)

```
int i = 0;
while (i < 10) { };
{
    System.out.println("i is " + i);
    i++;
}
```

(d)

DO IT
Now!

YaşOrtalaması.java

- Kullanıcının bir sınıfa ait gireceği yaş bilgilerinin ortlamasını hesaplayan bir Java programı geliştiriniz.

```
Output - YaşOrtalaması (run) X
run:
Kişi sayısı =>
4
Yaş bilgisi=> 23
Yaş bilgisi=> 25
Yaş bilgisi=> 32
Yaş bilgisi=> 29
Sınıfın yaş ortalaması 27.0
BUILD SUCCESSFUL (total time: 11 seconds)
```

```
Output - YaşOrtalaması (run) X
run:
Kişi sayısı =>
2
Yaş bilgisi=> 36
Yaş bilgisi=> 39
Sınıfın yaş ortalaması 37.5
BUILD SUCCESSFUL (total time: 4 seconds)
```

DO IT
Now!

En Büyük Ortak Bölen

- Kullanıcıdan iki tamsayı alarak en büyük ortak böleni (greatest common divisor) bulan bir Java Programı geliştiriniz !

```
Output - EbobBulalim (run) ×
run:
Bir sayı giriniz =>
15
İkinci bir sayı giriniz =>
30
En büyük ortak bölen 15
BUILD SUCCESSFUL (total time: 7 seconds)
```

```
Output - EbobBulalim (run) ×
run:
Bir sayı giriniz =>
32
İkinci bir sayı giriniz =>
48
En büyük ortak bölen 16
BUILD SUCCESSFUL (total time: 8 seconds)
```

```
Output - EbobBulalim (run) ×
run:
Bir sayı giriniz =>
125
İkinci bir sayı giriniz =>
2525
En büyük ortak bölen 25
BUILD SUCCESSFUL (total time: 5 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?