

Java Programlama ve Döngü Yapıları

8. Hafta

Dr. Öğr. Üyesi BÜŞRA ÖZDENİZCİ KÖSE

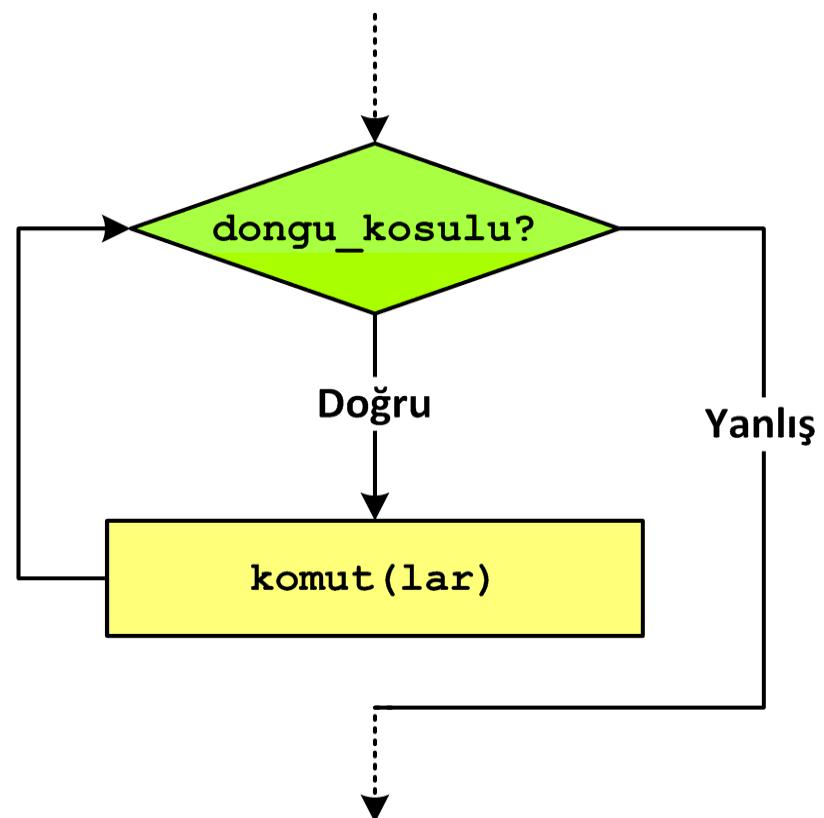
İşletme Bölümü

İşletme Fakültesi

WHILE Döngü Yapısı

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

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



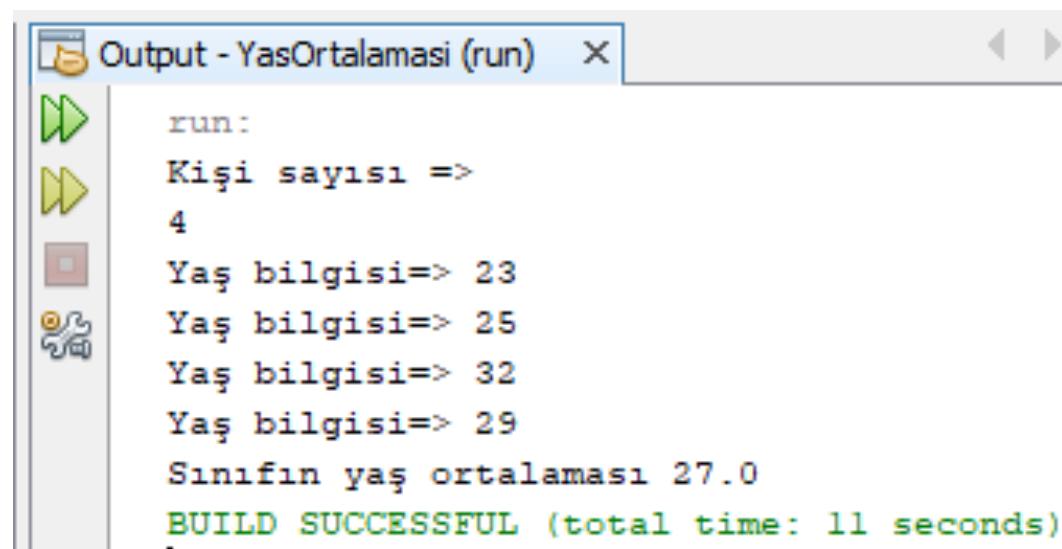
WHILE döngü yapısının genel yapısı

```
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  
}
```

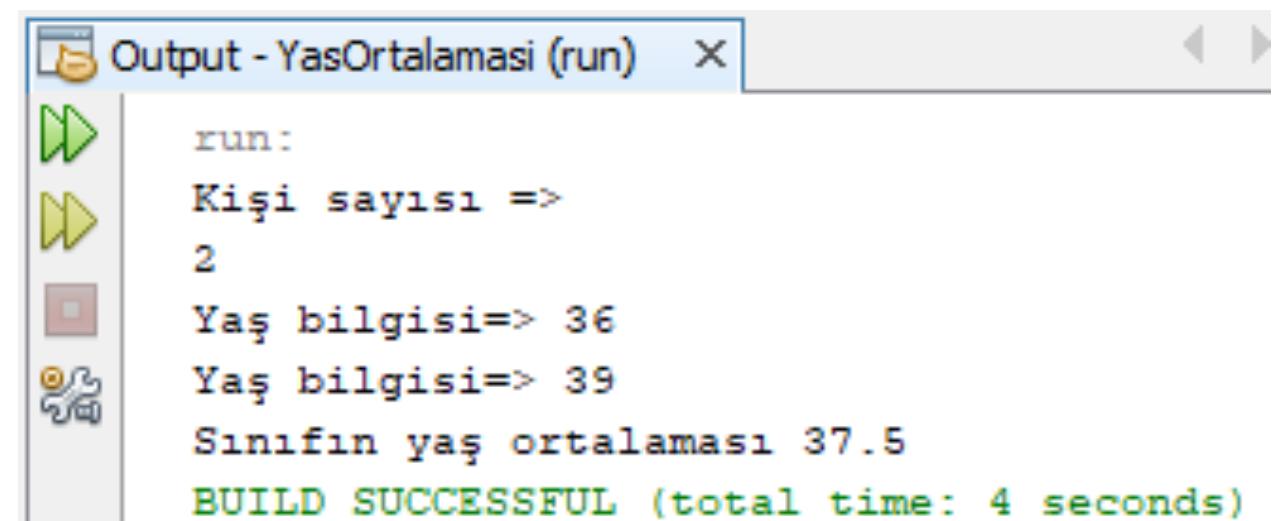
DO IT
Now!

YaşOrtalaması.java

- Kullanıcının bir sınıfa ait gireceği yaş bilgilerinin ortlamasını hesaplayan bir Java programı geliştiriniz; **while** döngüsü kullanalım.



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



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



```
YasOrtalaması.java
```

Source History

```
1 package yasortalaması;
2
3 import java.util.Scanner;
4
5 public class YasOrtalaması {
6
7     public static void main(String[] args) {
8
9         Scanner busra= new Scanner (System.in);
10        System.out.println ("Kişi sayısı =>");
11
12        int k = busra.nextInt(); //kisi sayısını aldım
13        int count=0;
14
15        double toplam=0;
16        double yas;
17
18        while (count < k) {
19            System.out.print ("Yaş bilgisi =>");  

20            yas =busra.nextInt();
21
22            toplam=toplam + yas;
23
24            count++;
25        }
26
27        double ortalama = toplam / k;
28        System.out.println ("Yaş Ortalaması " + ortalama );
29    }
30}
```

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)
```

DO IT
Now!

```
2 package ebobbulalim;
3 import java.util.Scanner;
4 public class EbobBulalim {
5     public static void main(String[] args) {
6
7         Scanner input = new Scanner(System.in);
8
9         System.out.println("Bir sayı giriniz =>");
10        int a = input.nextInt();
11
12        System.out.println("İkinci bir sayı giriniz =>");
13        int b = input.nextInt();
14
15        int ebob = 1;
16        int k = 1;
17
18        while ( (k <=a) && (k <= b) ) {
19
20            if ( (a%k ==0) && (b%k ==0) ) {
21                ebob = k;
22            }
23
24            k++;
25        }
26
27        System.out.println("En büyük ortak bölen " + ebob);
28    }
29 }
```

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  
}
```

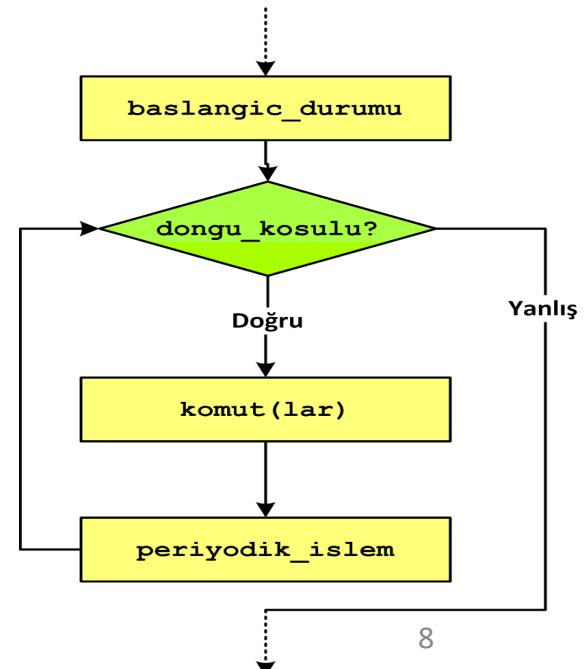


FOR döngü
formatında
yazalım !

FOR Döngü Yapısı

- Daha kolay formatta döngü yapıları yazmak için FOR döngü yapısı kullanılabilir:

```
for ( degisen = ilk_deger; sona_erme_kosulu; islem ) {  
    komut_1;  
    komut_2;  
    ...  
    komut_n;  
}
```



```
for (initial-action;  
     loop-continuation-condition;  
     action-after-each-iteration) {  
    // Loop body;  
}
```

(a)

Equivalent

```
initial-action;  
while (loop-continuation-condition) {  
    // Loop body;  
    action-after-each-iteration;  
}
```

(b)

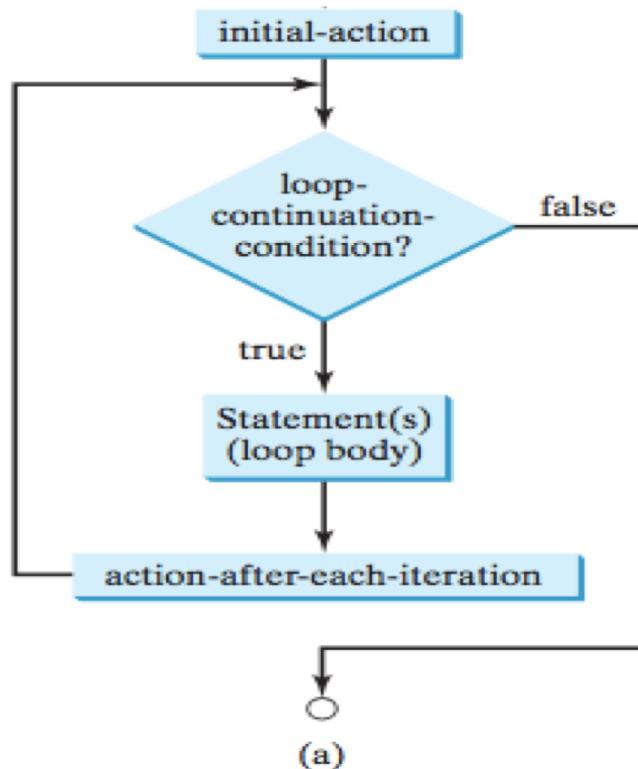
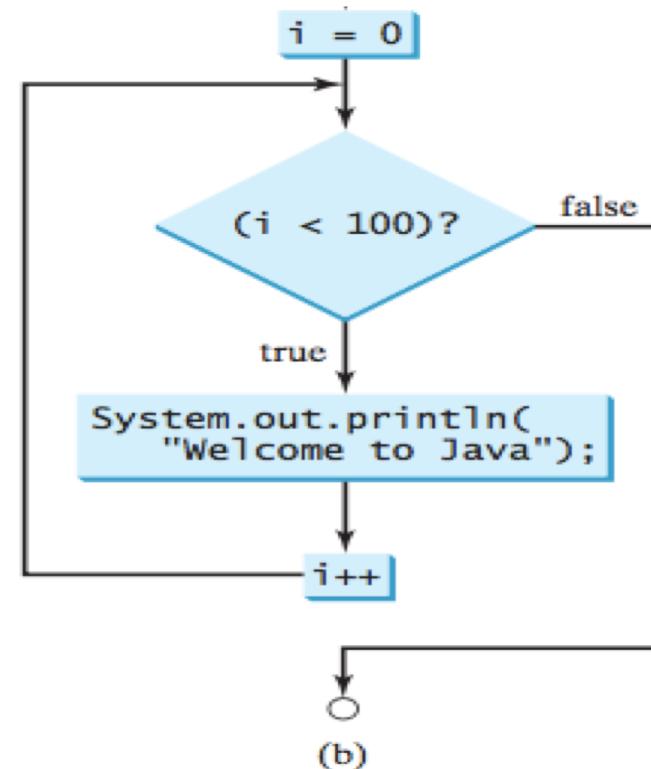


FIGURE 5.3 A **for** loop performs an initial action once, then repeatedly executes



DO IT
Now!

The screenshot shows an IDE interface with two main panes. The left pane is titled "Output - OnKezMerhaba (run)" and displays the following text:
run:
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
BUILD SUCCESSFUL (total time: 0 sec)

The right pane is titled "OnKezMerhaba.java" and contains the Java code:

```
1 package onkezmerhaba;  
2  
3 public class OnKezMerhaba {  
4  
5     public static void main(String[] args) {  
6         for ( int i=0; i<10; i=i+1) {  
7             System.out.println("Hello World!");  
8         }  
9     }  
10    }  
11}  
12  
13  
14}
```

FOR Döngü Yapısı

- Döngü değişkenini bir sayıç (counter) rolünde düşünebiliriz.
- Sayaç değişkeni ilk_deger'e atanır; her bir adımda islem ifadesi çalıştırılır, sona_erme_kosulu ifadesi değerlendirilir ve bu ifade true ise blok yapısı tekrar edilir.
- Küme parantezi içerisinde döngü komutları toplanır ve noktalı virgül ile döngü parametreleri ayrılarak belirlenir.

```
for ( degisken = ilk_deger; sona_erme_kosulu; islem; ) {  
    komut;  
}
```

DO IT
Now!

Yazdir.java

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

```
Output - OnKezMerhaba (run)
run:
Kaç kere yazdıralım =>
6
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
BUILD SUCCESSFUL (total time: 1 s)
```

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 yazdırırmak 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
24         System.out.println ("Goodbye ! ");
25
26     }
27 }
```



```
Scanner input = new Scanner (System.in);
System.out.println ("Kaç kere yazdırıralım => ");
int a= input.nextInt();

for ( int i=0; i<a; i=i+1) {
    System.out.println("Hello World!");
}
```

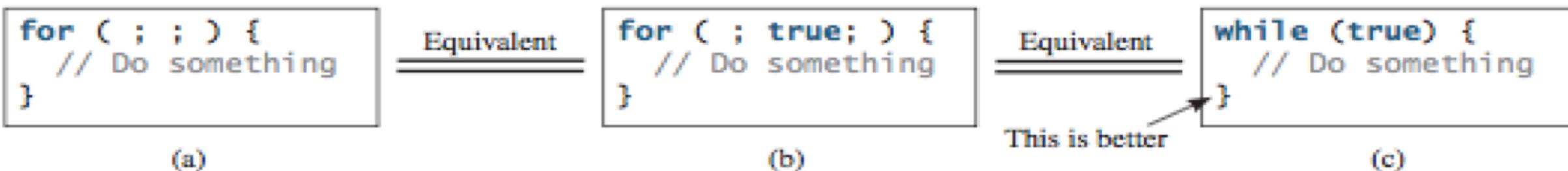
DO IT
Now!

The screenshot shows a Java development environment with two tabs: "Start Page" and "OnKezMerhaba.java". The "OnKezMerhaba.java" tab is active, displaying the following code:

```
1 package onkezmerhaba;
2
3 import java.util.Scanner;
4
5 public class OnKezMerhaba {
6
7     public static void main(String[] args) {
8
9         Scanner input = new Scanner (System.in);
10
11        System.out.println ("Kaç kere yazdırılalım => ");
12        int a= input.nextInt();
13
14        for ( int i=0; i<a; i++) {
15            System.out.println("Hello World!");
16        }
17    }
18
19
20 }
```

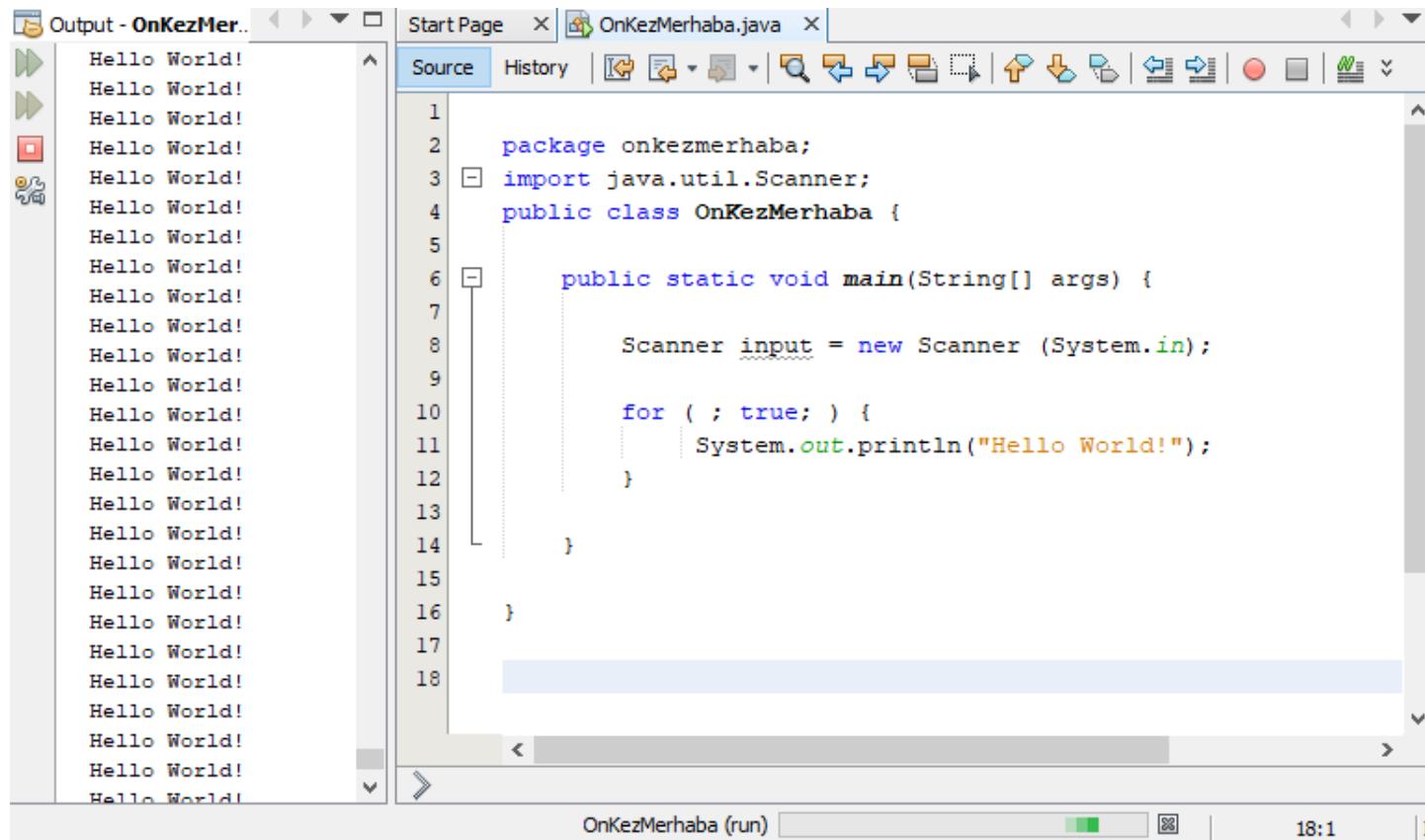
The code uses a for loop to print "Hello World!" to the console a specified number of times. The output window on the left shows the results of running the program, including the user input "3" and the three "Hello World!" outputs.

Sonsuz FOR Döngüsü



Bu ifadelerde FOR döngüsü içerisinde herhangi bir döngü değişkeni ve mantıksal ifade belirtilmemiş için FOR döngüsü sonsuza dek çalışacaktır. Blok grup içerisinde bulunan komut veya komutları uygulayacaktır. Sonsuz döngü kullanmak için (c) daha uygun bir yöntemdir.

Sonsuz FOR Döngüsü

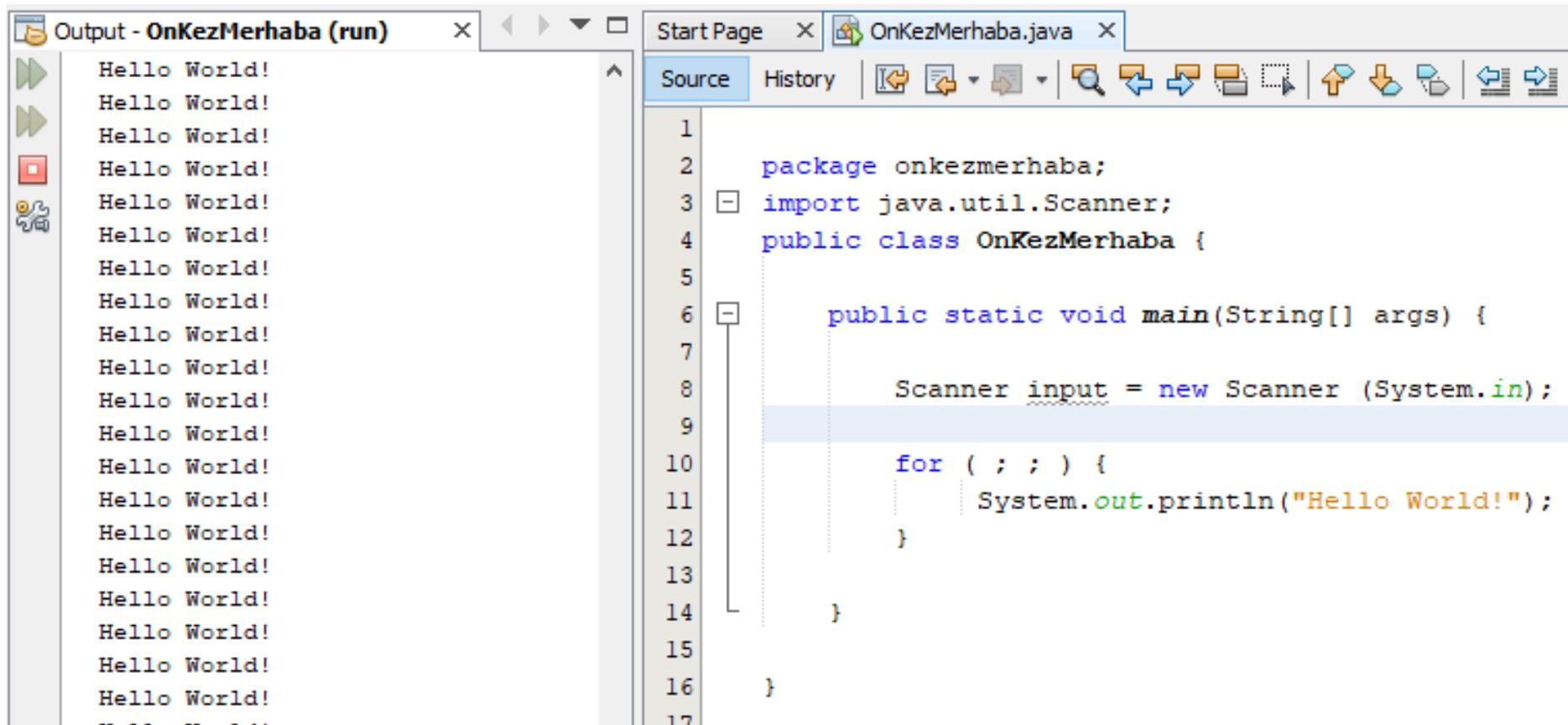


The screenshot shows a Java development environment with the following code in the main.java file:

```
1 package onkezmerhaba;
2 import java.util.Scanner;
3 public class OnKezMerhaba {
4
5     public static void main(String[] args) {
6         Scanner input = new Scanner (System.in);
7
8         for ( ; true; ) {
9             System.out.println("Hello World!");
10        }
11    }
12
13 }
14
15 }
16
17 }
18 }
```

The Output window on the left displays the repeated output "Hello World!" for each iteration of the loop.

Sonsuz FOR Döngüsü



The screenshot shows a Java development environment with two panes. The left pane, titled "Output - OnKezMerhaba (run)", displays the repeated output "Hello World!" indicating the program is stuck in a loop. The right pane, titled "OnKezMerhaba.java", shows the source code of the Java program:

```
1 package onkezmerhaba;
2 import java.util.Scanner;
3 public class OnKezMerhaba {
4
5     public static void main(String[] args) {
6         Scanner input = new Scanner (System.in);
7         for ( ; ; ) {
8             System.out.println("Hello World!");
9         }
10    }
11 }
```

The code uses a Scanner to read input from the standard input stream (System.in) and prints "Hello World!" to the standard output stream (System.out) in an infinite loop defined by the empty conditions in the for loop.

ToplamHesapla.java

**DO IT
NOW!**

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

```
Output - ToplamHesapla (run)  X

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

DO IT
Now!

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

- Output - ToplamHesapla (run) ...**: Shows the build log:
 - run:
 - Toplam : 55
 - BUILD SUCCESSFUL (total time: 0 seconds)
- Start Page**: The current file being edited.
- Source**: The active tab in the editor.
- History**: A history tab.
- Toolbar**: Includes icons for back, forward, search, and file operations.
- Code Editor Content**: The Java code for `ToplamHesapla.java`. The code calculates the sum of integers from 1 to 10 and prints the result.

```
1 package toplamhesapla;
2
3 public class ToplamHesapla {
4
5     public static void main(String[] args) {
6         int toplam = 0;
7
8         for ( int sayac=1; sayac <=10; sayac++ ) {
9
10            toplam += sayac;
11        }
12
13        System.out.println("Toplam : " + toplam);
14    }
15
16 }
17 }
```

DO IT
Now!

Yazdir.java

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

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

```
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 yazdirmak 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
24        System.out.println ("Goodbye ! ");
25
26    }
}
```

DO IT
NOW!

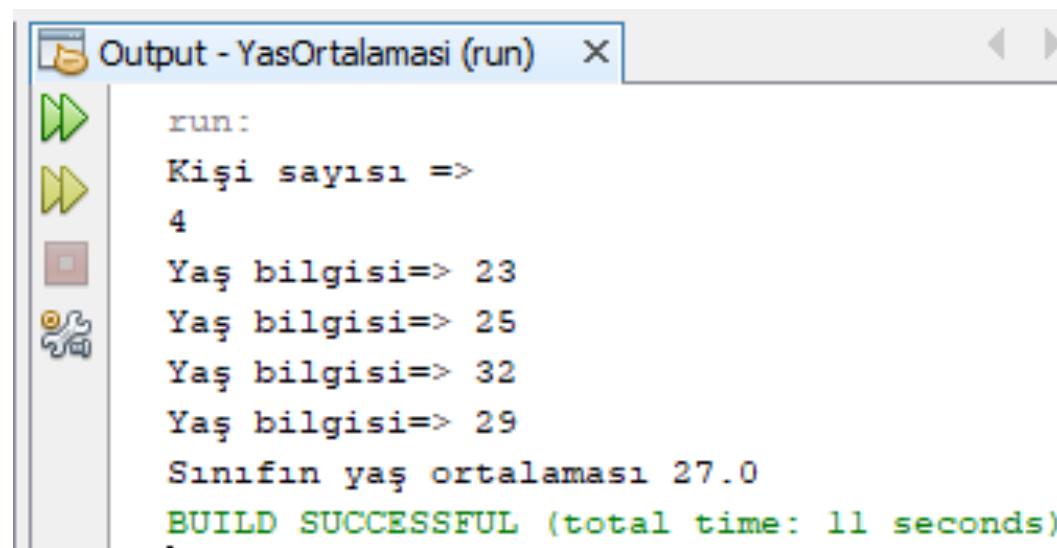
The screenshot shows a Java development environment with two main windows. The left window, titled 'Output - Yazdir (run)', displays the console output of a run. It shows the application asking for input ('Kaç kere yazdırılmak istiyorsun?'), receiving the value '4', and then printing 'Hello World :)' four times. Below this, it shows a 'BUILD SUCCESSFUL' message with a total time of 3 seconds. The right window is titled 'Start Page X Yazdir.java X' and contains the source code for the 'Yazdir' class. The code uses a Scanner to read an integer from standard input, prints a question to standard output, reads the integer, and then prints 'Hello World :)' for each iteration of a loop that runs from 0 to the input value (exclusive). The code is numbered from 1 to 21.

```
1 package yazdir;
2
3 import java.util.Scanner;
4
5
6 public class Yazdir {
7
8     public static void main(String[] args) {
9
10         Scanner busra = new Scanner (System.in);
11
12         System.out.println("Kaç kere yazdırılmak istiyorsun?");
13         int k = busra.nextInt();
14
15         for (int i=0; i < k; i++) {
16
17             System.out.println ("Hello World :)");
18         }
19     }
20 }
21 }
```

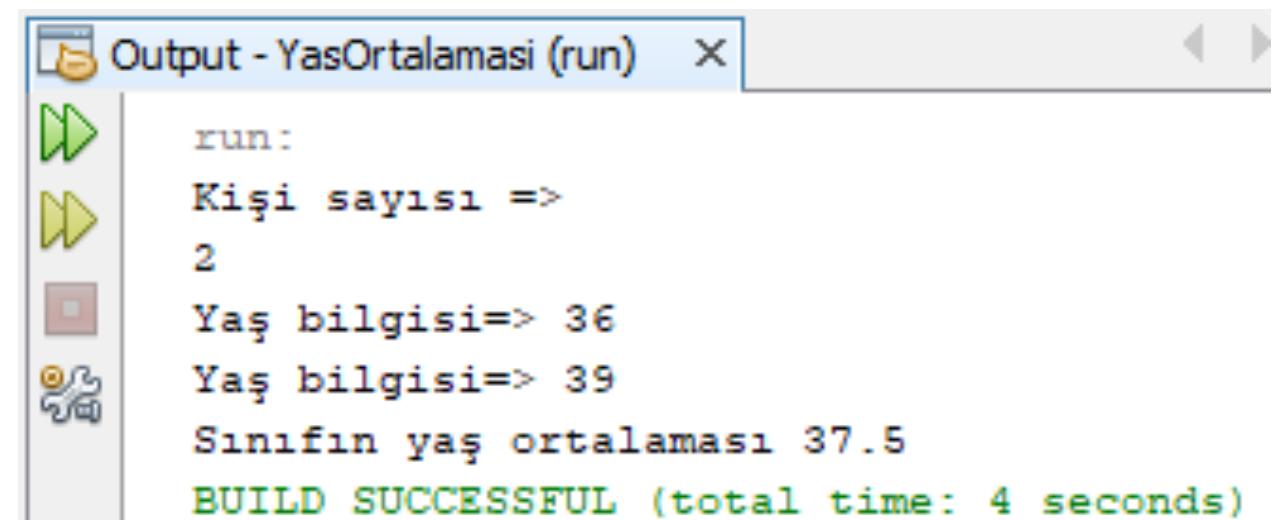
DO IT
Now!

YaşOrtalaması.java

- Kullanıcının bir sınıfına ait gireceği yaş bilgilerinin ortalamasını hesaplayan bir Java programı geliştiriniz; **for** döngüsü kullanalım.



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



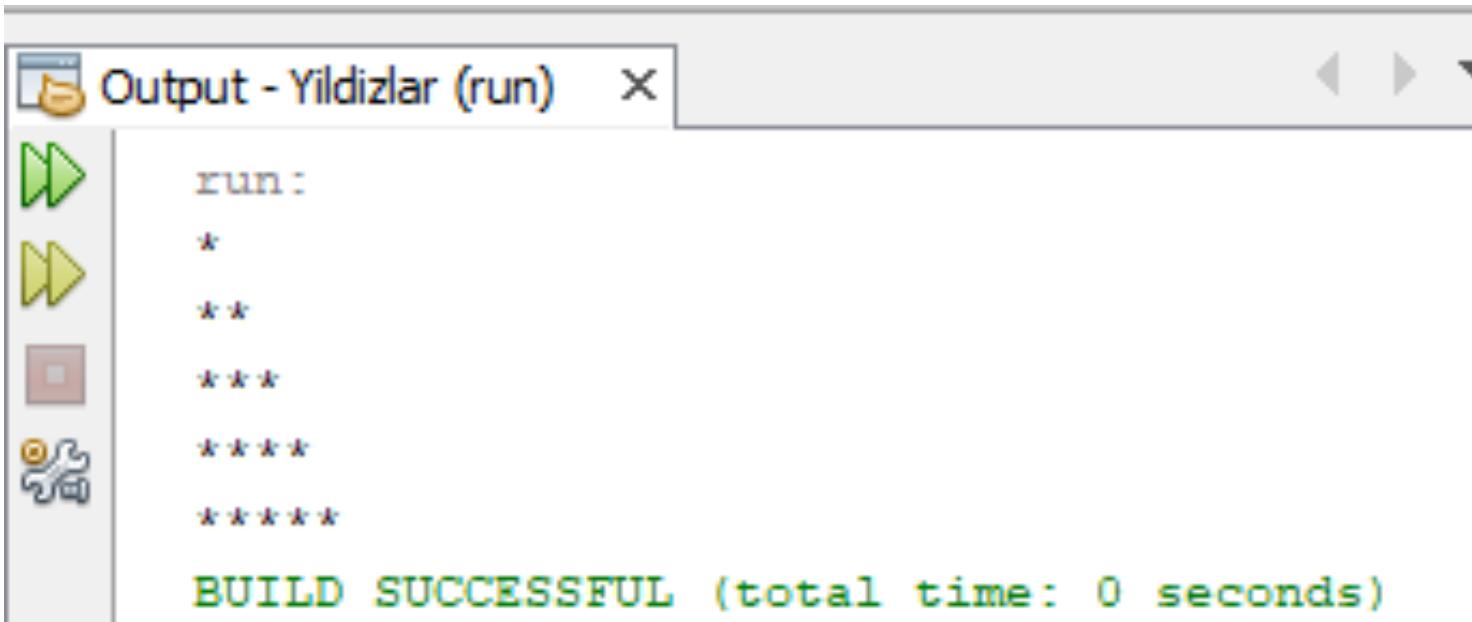
```
Output - YaşOrtalaması (run)
run:
Kişi sayısı =>
2
Yaş bilgisi=> 36
Yaş bilgisi=> 39
Sınıfin yaş ortalaması 37.5
BUILD SUCCESSFUL (total time: 4 seconds)
```

DO IT
Now!

```
1
2     package yasortalamasi;
3     import java.util.Scanner;
4     public class YasOrtalaması {
5         public static void main(String[] args) {
6
7             Scanner busra = new Scanner (System.in);
8             System.out.println("Kişi sayısı => ");
9             int n = busra.nextInt();
10
11            double toplam=0;
12            double yas;
13
14            for (int i=1; i <=n; i++) {
15                System.out.print ("Yaş bilgisi=> ");
16                yas= busra.nextDouble();
17                toplam = toplam + yas;
18            }
19
20            double ortalama = (toplam/n);
21
22            System.out.println("Sınıfın yaş ortalaması " + ortalama );
23
24        }
25    }
```

İç İçe FOR Döngüleri

- İç içe (nested) FOR yapısı kullanarak daha karmaşık, çeşitli işlemler de yapabilirsiniz. Aşağıda verilen örnekte iç içe iki tane FOR yapısı kullanılmıştır. İlk FOR döngüsü satır değişkeni için, ikinci FOR yapısı ise sutun döngü değişkeni için oluşturulmuştur. Her bir satırda, satırın değişkeninin değeri kadar * (yıldız) ekrana bastırılmaktadır.



The screenshot shows an IDE's Output window titled "Output - Yıldızlar (run)". The window displays the execution of a program named "run". The output consists of six lines of asterisks (*), where the number of asterisks increases by one in each subsequent line, forming a triangle:

```
run:  
*  
**  
***  
****  
*****
```

At the bottom of the window, the message "BUILD SUCCESSFUL (total time: 0 seconds)" is displayed in green text.

The screenshot shows a Java code editor with the file 'Yildizlar.java' open. The code prints a diamond shape of asterisks using nested for loops. The editor interface includes tabs for 'Source' and 'History', and various toolbars with icons for file operations like opening, saving, and search.

```
1 package yildizlar;
2
3 public class Yildizlar {
4
5     public static void main(String[] args) {
6
7         for ( int x = 1; x <= 5; x++ ) {
8
9             for ( int y = 1; y <= x; y++ ) {
10                System.out.print("*");
11            }
12
13            System.out.println();
14        }
15    }
16}
17}
```

```
Output - Yildizlar (run)  X  ◀ ▶ ▾

run:
*
**
***
*****
*****
BUILD SUCCESSFUL (total time: 0 seconds)
```

**DO IT
NOW!**

Yaygın Yazım Hataları ☺

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

Error

(a)

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

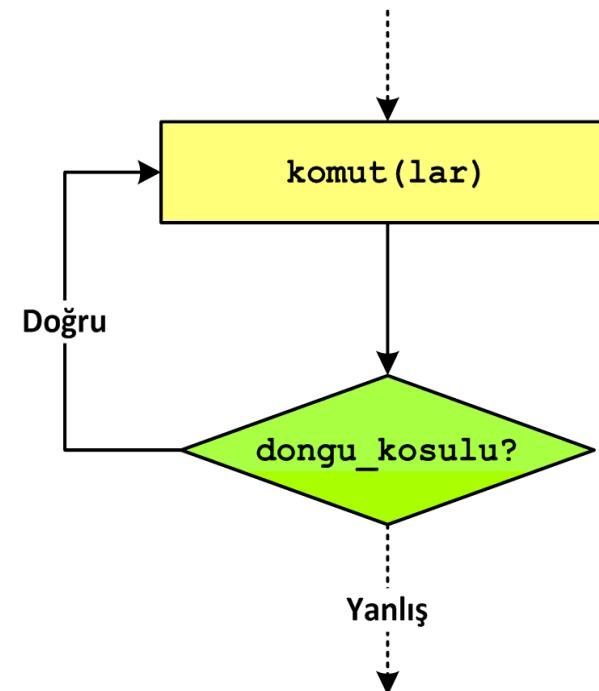
Empty body

(b)

DO-WHILE Döngü Yapısı

- do-while döngüsü while döngüsü ile benzerdir; sadece önce döngü içerisindeki komutları uygular ve sonra döngü koşulunu kontrol eder.

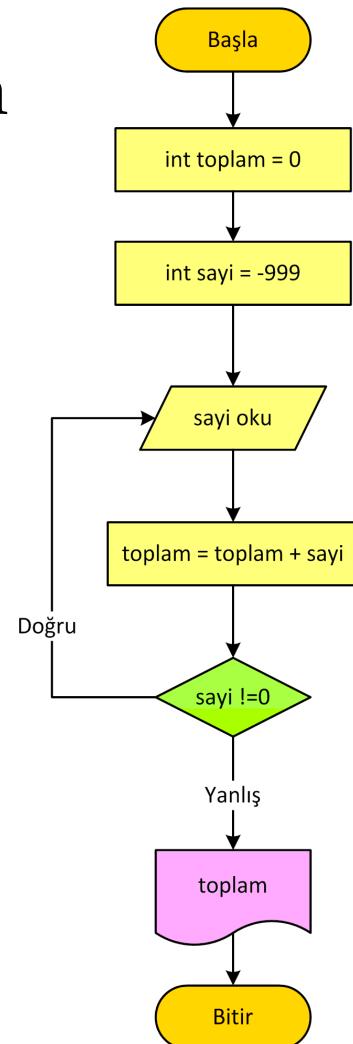
```
do {  
    komut_1;  
    komut_2;  
    ...  
    komut_n;  
} while (dongu_kosulu);
```



ToplaTest.java

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

```
Output - ToplaTest (run) X
run:
0 ile programı sonlandırabilirsiniz...
Sayı => 3
Sayı => 6
Sayı => 9
Sayı => 5
Sayı => 0
Toplam = 23
BUILD SUCCESSFUL (total time: 6 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 }
```



The screenshot shows a Java code editor window titled "ToplaTest.java". The code implements a simple addition program using a do-while loop. The code is as follows:

```
1 package topplatest;
2
3
4 import java.util.Scanner;
5
6 public class ToplaTest {
7
8     public static void main(String[] args) {
9         Scanner input = new Scanner(System.in);
10        System.out.println("0 ile programı sonlandırabilirsiniz...");
11        int toplam = 0;
12        int sayi;
13
14        do {
15            System.out.print("Sayı => ");
16            sayi = input.nextInt();
17            toplam = toplam + sayi;
18        } while (sayi != 0);
19
20        System.out.println("Toplam = " + toplam);
21    }
22
23
24 }
```

DO IT
NOW!



```
1 package dongulercalisma;
2
3
4 import java.util.Scanner;
5
6 public class DongulerCalisma {
7
8     public static void main(String[] args) {
9         int toplam=0;
10
11         for ( int count=0; count<=20; count+=2 ) {
12
13             toplam = toplam +count;
14             //toplam +=count;
15
16         }
17
18         System.out.println ("1'den 20'ye kadar çift sayıların toplamı " +toplam);
19     }
20
21 }
```

dongulercalisma.DongulerCalisma > main >

Output - DongulerCalisma (run)

run:
1'den 20'ye kadar çift sayıların toplamı 110
BUILD SUCCESSFUL (total time: 0 seconds)

DO IT
Now!

```
2 package dongulercalisma;
3
4 import java.util.Scanner;
5
6 public class DongulerCalisma {
7
8     public static void main(String[] args) {
9         int toplam=0;
10
11
12         for ( int count=1; count<=20; count+=2 ) {
13
14             toplam = toplam +count;
15             //toplam +=count;
16
17
18             System.out.println ("1'den 20'ye kadar tek sayıların toplamı " +toplam);
19
20
21     }
```

dongulercalisma.DongulerCalisma > main > for (int count = 1; count <= 20; count += 2) >

Output - DongulerCalisma (run)

```
run:
1'den 20'ye kadar tek sayıların toplamı 100
BUILD SUCCESSFUL (total time: 0 seconds)
```

DO IT
Now!

```
2 package dongulercalisma;
3
4 import java.util.Scanner;
5
6 public class DongulerCalisma {
7
8     public static void main(String[] args) {
9         int toplam=0;
10
11
12         for ( int count=1; count<=20; count++) {
13             if (count%2 == 1)
14                 toplam = toplam +count;
15             //toplam +=count;
16         }
17
18         System.out.println ("1'den 20'ye kadar tek sayıların toplamı " +toplam);
19     }
20
21 }
22
```

dongulercalisma.DongulerCalisma > main > for (int count = 1; count <= 20; count++) >

Output - DongulerCalisma (run)

```
run:
1'den 20'ye kadar tek sayıların toplamı 100
BUILD SUCCESSFUL (total time: 0 seconds)
```

Fakat bu doğru !

In the case of the **do-while** loop, the semicolon is needed to end the loop.

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

Correct

Break ve Continue

- **break** komutu, SWITCH kontrol yapısında, akış sırasını ilgili yapının sonrasına yönlendirmek için kullanılır.
- **break** komutunun döngü yapılarında da kullanılması mümkündür; döngüden çıkmak için kullanılabilir.
- **continue** komutu da döngü yapılarında döngünün sonuna gitmek için kullanılabilir, döngüden çıkmak için değildir.
- Fakat **break** ve **continue** kelimelerinin döngü yapılarında kullanılması kaliteli programlama tekniğine aykırıdır, kullanımını tavsiye etmiyoruz.

TestBreak.java

Source History

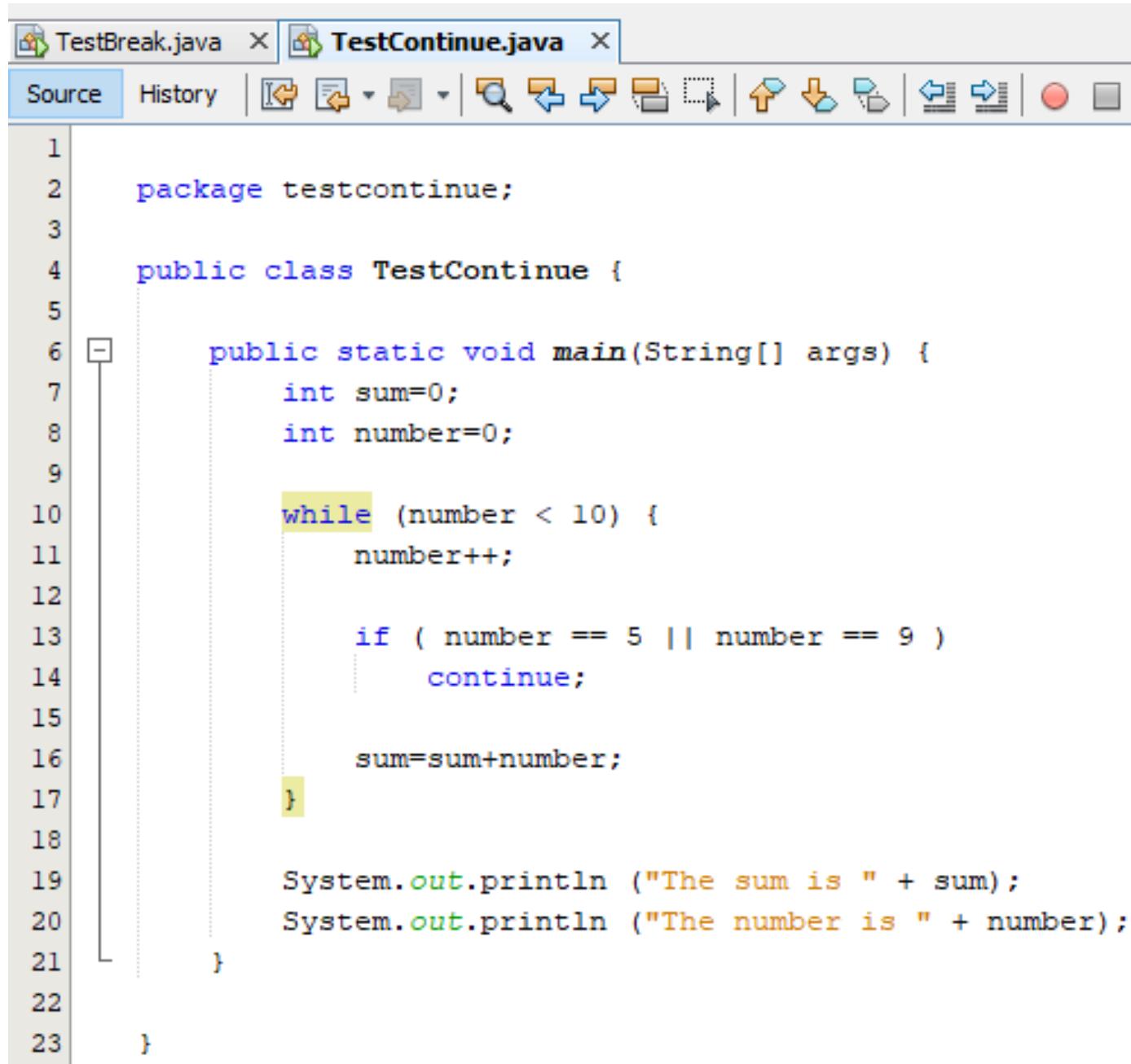


```
1
2     package testbreak;
3
4     public class TestBreak {
5         public static void main(String[] args) {
6             int sum=0;
7             int number=0;
8
9             while (number < 10) {
10                 number++;
11                 sum=sum+number;
12
13                 if ( sum > 10 )
14                     break;
15
16                 System.out.println ("The sum is " + sum);
17                 System.out.println ("The number is " + number);
18
19             }
20
21 }
```

DO IT
Now!

Output - TestBreak (run)

```
run:
The sum is 15
The number is 5
BUILD SUCCESSFUL (total time: 0 seconds)
```



DO IT
NOW!

```
Output - TestContinue (run)  X  
run:  
The sum is 31  
The number is 9  
BUILD SUCCESSFUL (total time: 0 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?