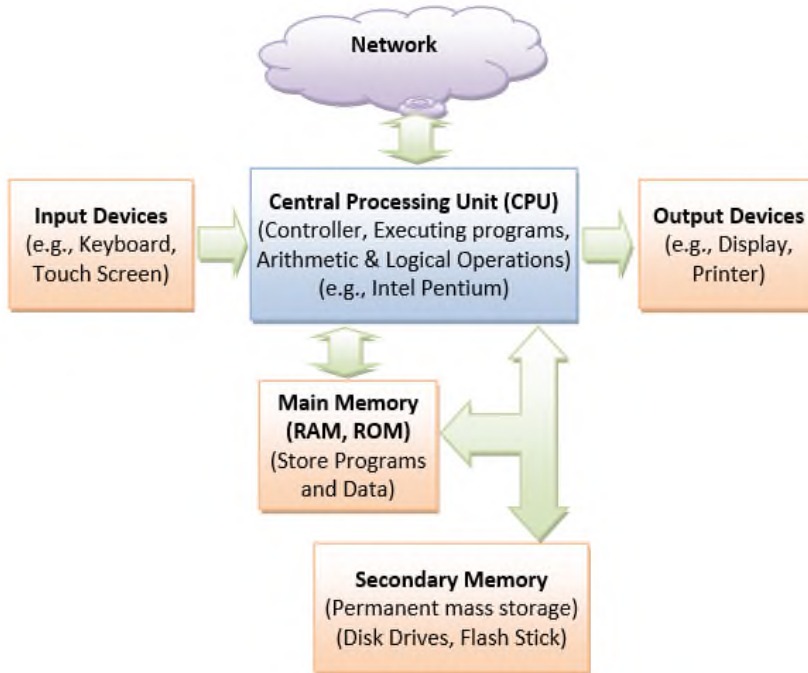


İleri Programlama

Java'ya Giriş

Hüseyin Ahmetođlu

Bilgisayar Mimarisi



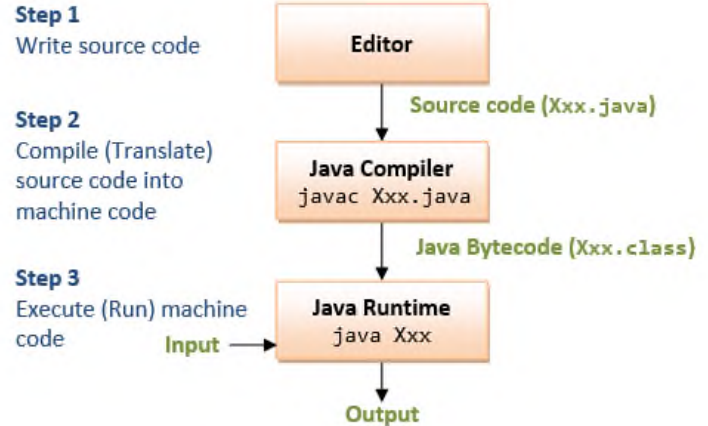
İlk Java Programı

```
1  /*
2   * First Java program, which says hello.
3   */
4  public class Hello { // Save as "Hello.java"
5      public static void main(String[] args) { // Program entry point
6          System.out.println("Hello, world!"); // Print text message
7      }
8  }
```

Nasıl çalışıyor?

Java Programlama Adımları

- ▶ Adım 1: "Xxx.java" kaynak kodunu yazın.
- ▶ Adım 2: "Xxx.java" kaynak kodunu, "javac Xxx.java" komutunu vererek JDK'nın Java derleyicisini kullanarak Java taşınabilir bayt kodu (veya makine kodu) "Xxx.class" olarak derleyin.
- ▶ Adım 3: "java Xxx" komutunu vererek JDK Java Runtime kullanarak derlenmiş "Xxx.class" bayt kodunu çalıştırın.



Java Program Şablonu

```
1  /*
2   * Comment to state the purpose of the program
3   */
4  public class Classname { // Choose a meaningful Classname. Save as "Classname.java"
5      public static void main(String[] args) { // Entry point of the program
6          // Your programming statements here!!!
7      }
8  }
```

System.out.print () ile çıktı almak

- ▶ `System.out.println (aString)` (print-line) bir String yazdırır ve imleci sonraki satırın başına ilerletir.
- ▶ `System.out.print (aString)` aString yazdırır, ancak imleci yazdırılan dizeden sonra yerleştirir.
- ▶ Parametresiz `System.out.println ()` bir satırsonu yazdırır.

```

1  /*
2  * Test System.out.println() (print-line) and System.out.print().
3  */
4  public class PrintTest { // Save as "PrintTest.java"
5      public static void main(String[] args) {
6          System.out.println("Hello world!"); // Advance the cursor to the beginning of next line after printing
7          System.out.println("Hello world again!"); // Advance the cursor to the beginning of next line after printing
8          System.out.println(); // Print an empty line
9          System.out.print("Hello world!"); // Cursor stayed after the printed string
10         System.out.print("Hello world again!"); // Cursor stayed after the printed string
11         System.out.println(); // Print an empty line
12         System.out.print("Hello,");
13         System.out.print(" "); // Print a space
14         System.out.println("world!");
15         System.out.println("Hello, world!");
16     }
17 }

```

```

Hello world!
Hello world again!

```

```

Hello world!Hello world again!
Hello, world!
Hello, world!

```

Egzersizler

- ▶ Aşağıdaki çıktıları konsol ekranında verecek kodları System.out.print() ve System.out.println() kullanarak kodlayınız.

```
* * * * *      * * * * *      * * * * *      *
* * * * *      *       *      *       *      * * * * *
* * * * *      *       *      * *      * *      * *
* * * * *      *       *      * *      * *      * *
* * * * *      * * * * *      *       *       * *      *

(a)           (b)           (c)           (d)
```

```
      . _ .
      (oo)
 /=====\/
 / || @@ ||
 * ||----||
  w      w
  ..     ..
```

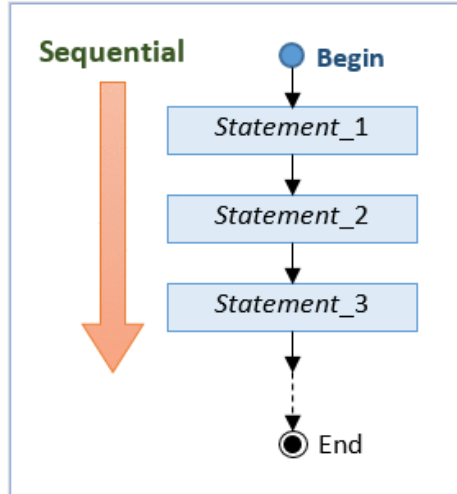

Örnek

```
1  /*
2   * Add five integers and display their sum.
3   */
4  public class FiveIntegerSum { // Save as "FiveIntegerSum.java"
5      public static void main(String[] args) {
6          int number1 = 11; // Declare 5 integer variables and assign a value
7          int number2 = 22;
8          int number3 = 33;
9          int number4 = 44;
10         int number5 = 55;
11         int sum; // Declare an integer variable called sum to hold the sum
12         sum = number1 + number2 + number3 + number4 + number5; // Compute sum
13         System.out.print("The sum is "); // Print a descriptive string
14                                     // Cursor stays after the printed string
15         System.out.println(sum); // Print the value stored in variable sum
16                                 // Cursor advances to the beginning of next line
17     }
18 }
```

Program nedir?

- Bir program, öngörülebilir bir şekilde birbiri ardına yürütülen bir talimatlar dizisidir.

Sıralı akış, programlama deyimlerinin yazıldıkları sırayla yukarıdan aşağıya sıralı bir şekilde yürütülür.



```

1  /*
2  * Print the area and circumference of a circle, given its radius.
3  */
4  public class CircleComputation { // Save as "CircleComputation.java"
5      public static void main(String[] args) {
6          // Declare 3 double variables to hold radius, area and circumference.
7          // A "double" holds floating-point number with an optional fractional part.
8          double radius, area, circumference;
9          // Declare a double to hold PI.
10         // Declare as "final" to specify that its value cannot be changed (i.e. constant).
11         final double PI = 3.14159265;
12
13         // Assign a value to radius. (We shall read in the value from the keyboard later.)
14         radius = 1.2;
15         // Compute area and circumference
16         area = radius * radius * PI;
17         circumference = 2.0 * radius * PI;
18
19         // Print results
20         System.out.print("The radius is "); // Print description
21         System.out.println(radius); // Print the value stored in the variable
22         System.out.print("The area is ");
23         System.out.println(area);
24         System.out.print("The circumference is ");
25         System.out.println(circumference);
26     }
27 }

```

```

The radius is 1.2
The area is 4.523893416
The circumference is 7.5398223600000005

```

Egzersizler

- Önceki slaytta yer alan programı dikdörtgen ve silindir için ayrı ayrı kodlayınız.

Değişken nedir?

- ▶ Bir bilgisayar programı, verileri işler. Değişken, işlenmek üzere bir veri parçasını depolayan bir depolama yeridir (bir ev, bir güvercin deliği, bir mektup kutusu gibi). Değişken olarak adlandırılır çünkü içinde depolanan değeri değiştirebilirsiniz.
- ▶ Daha kesin olarak, bir değişken, belirli bir veri türünün değerini depolayan adlandırılmış bir depolama konumudur. Başka bir deyişle, bir değişkenin bir **adı**, bir **türü** vardır ve bu belirli türde bir **değer** depolar.
- ▶ Bir değişkenin adı (tanımlayıcı olarak da bilinir), örneğin yarıçap, alan, yaş, yükseklik vb. Ad, değişkene bir değer atamak (örneğin yarıçap = 1,2) ve depolanan değeri (ör. Yarıçap * yarıçap * 3.14159265) almak için her değişkeni benzersiz şekilde tanımlamak için gereklidir.
- ▶ Bir değişkenin bir türü vardır. Tür örnekleri:
 - int: sıfır, pozitif ve negatif tam sayılar dahil olmak üzere tam sayılar (veya tam sayılar veya sabit noktalı sayılar) anlamına gelir; 123, -456 ve 0 gibi;
 - double: isteğe bağlı bir ondalık basamağa ve kesirli bölüme sahip 3.1416, -55.66 gibi kayan noktalı sayılar veya gerçek sayılar içindir.
 - Dize: "Merhaba", "Günaydın!" Gibi metinler içindir. Dizeler iki çift tırnak içine alınacaktır.
- ▶ Bir değişken, bildirilen türün bir değerini depolayabilir. Çoğu programlama dilinde bir değişkenin bir türle ilişkilendirildiğini ve yalnızca o belirli türdeki değeri depolayabileceğini not etmek önemlidir. Örneğin, bir int değişkeni 123 gibi bir tamsayı değeri depolayabilir, ancak 12.34 gibi gerçek bir sayı veya "Merhaba" gibi bir metin depolayamaz. Tip kavramı, verilerin yorumlanmasını basitleştirmek için erken programlama dillerinde yer almıştır.

Değişken nedir?

| TYPE | NAME | VALUE | |
|--------|-------------|-----------|-----------------------------------|
| int | number → | 1 | Stored only Integer |
| int | sum → | 500500 | Stored only Integer |
| double | radius → | 5.5 | Stored only floating-point number |
| double | area → | 95.0334 | Stored only floating-point number |
| String | greeting → | Hello | Stored only texts |
| String | statusMsg → | Game Over | Stored only texts |

*A variable has a **name**, stores a **value** of the declared **type**.*

Değişkeni tanımlamak ve kullanmak

- Bir değişkeni kullanmak için, önce adını ve türünü ve isteğe bağlı bir başlangıç değerini aşağıdaki sözdizimlerinden birinde bildirmeniz gerekir:

```
type varName;           // Declare a variable of a type
type varName1, varName2,...; // Declare multiple variables of the SAME type
type varName = initialValue; // Declare a variable of a type, and assign an initial value
type varName1 = initialValue1, varName2 = initialValue2,... ; // Declare variables with initial values
```

```
int sum;                // Declare a variable named "sum" of the type "int" for storing an integer.
                        // Terminate the statement with a semi-colon.
double average;        // Declare a variable named "average" of the type "double" for storing a real number.
int number1, number2; // Declare 2 "int" variables named "number1" and "number2", separated by a comma.
int height = 20;       // Declare an "int" variable, and assign an initial value.
String msg = "Hello"; // Declare a "String" variable, and assign an initial value.
```

Değişkeni tanımlamak ve kullanmak

- ▶ Her değişken bildirim ifadesi bir türle başlar ve yalnızca bu tür için geçerlidir. Yani, bir değişken bildirim ifadesinde 2 türü karıştıramazsınız.
- ▶ Her ifade noktalı virgülle (;) sonlandırılır.
- ▶ Çok değişkenli bildirimde, değişken adları virgülle (,) ayrılır.
- ▶ Atama operatörü olarak bilinen '=' sembolü, bir bildirim ifadesinde bir değişkene bir başlangıç değeri atamak için kullanılabilir.

Değişkeni tanımlamak ve kullanmak

```
int number;           // Declare a variable named "number" of the type "int" (integer).
number = 99;         // Assign an integer value of 99 to the variable "number".
number = 88;         // Re-assign a value of 88 to "number".
number = number + 1; // Evaluate "number + 1", and assign the result back to "number".
int sum = 0;         // Declare an int variable named "sum" and assign an initial value of 0.
sum = sum + number; // Evaluate "sum + number", and assign the result back to "sum", i.e. add number into sum.
int num1 = 5, num2 = 6; // Declare and initialize 2 int variables in one statement, separated by a comma.
double radius = 1.5; // Declare a variable named "radius", and initialize to 1.5.
String msg;         // Declare a variable named msg of the type "String".
msg = "Hello";     // Assign a double-quoted text string to the String variable.
int number;         // ERROR: The variable named "number" has already been declared.
sum = 55.66;       // ERROR: The variable "sum" is an int. It cannot be assigned a double.
sum = "Hello";     // ERROR: The variable "sum" is an int. It cannot be assigned a string.
```

Dikkat!

- ▶ Her değişken yalnızca bir kez bildirilebilir. (Aynı adrese sahip iki eviniz olamaz.)
- ▶ Java'da, kullanılmadan önce bildirildiği sürece, programınızın herhangi bir yerinde bir değişken bildirebilirsiniz. (Bazı eski diller, programın başında tüm değişkenleri bildirmenizi gerektirir.)
- ▶ Bir değişken bildirildikten sonra, atama operatörü '=' aracılığıyla bu değişkene bir değer atayabilir ve sonra yeniden yeni bir değer atayabilirsiniz.
- ▶ Bir değişkenin türü bildirildikten sonra, yalnızca o türden bir değeri saklayabilir. Örneğin, bir int değişkeni yalnızca 123 gibi bir tamsayı tutabilir ve -2.17 gibi kayan noktalı sayı veya "Merhaba" gibi metin dizesi bu değişkende saklanamaz.
- ▶ Bir kez bildirildikten sonra, bir değişkenin türü DEĞİŞTİRİLEMEZ.

Değişkene değer atama $x = x + 1$?

- ▶ Programlamada atama ('=' olarak gösterilir) Matematikteki eşitlikten farklıdır ('=' olarak da belirtilir). Örneğin, " $x = x + 1$ " Matematikte geçersizdir. Bununla birlikte, programlamada, x artı 1 değerini hesaplamak ve sonucu x değişkenine geri atamak anlamına gelir.
- ▶ " $x + y = 1$ " Matematikte geçerlidir ancak programlamada geçersizdir.
- ▶ Programlamada, '=' RHS (Right-Hand Side) bir değer olarak değerlendirilmelidir; LHS (Left-Hand Side) ise değişken olacaktır. Yani, önce RHS'yi değerlendirin, ardından sonucu LHS'ye atayın.

Bazı diller, eşitlikle karışıklığı önlemek için atama operatörü olarak: = veya -> kullanır.

```
int x = 79;  
x = x + 1;
```

x 79


```
int x = 79;  
x = x + 1;
```

x 79 

```
int x = 79;  
x = x + 1;
```

x 79 $79 + 1$

```
int x = 79;  
x = x + 1;
```

x 79 

```
int x = 79;  
x = x + 1;
```

END x 80

Temel Aritmetiksel Operatörler

| Operator | Mode | Usage | Meaning | Example x=5; y=2 |
|----------|-------------------------------|-------------|---------------------|--|
| + | Binary Unary | x + y +x | Addition | x + y returns 7 |
| - | Binary Unary | x - y -x | Subtraction | x - y returns 3 |
| * | Binary | x * y | Multiplication | x * y returns 10 |
| / | Binary | x / y | Division | x / y returns 2 |
| % | Binary | x % y | Modulus (Remainder) | x % y returns 1 |
| ++ | Unary Prefix Unary Postfix | ++x x++ | Increment by 1 | ++x or x++ (x is 6) Same as x = x + 1 |
| -- | Unary Prefix Unary Postfix | --x x-- | Decrement by 1 | --y or y-- (y is 1) Same as y = y - 1 |

```

1  /*
2  * Test Arithmetic Operations
3  */
4  public class ArithmeticTest {    // Save as "ArithmeticTest.java"
5      public static void main(String[] args) {
6          int number1 = 98; // Declare an int variable number1 and initialize it to 98
7          int number2 = 5;  // Declare an int variable number2 and initialize it to 5
8          int sum, difference, product, quotient, remainder; // Declare 5 int variables to hold results
9
10         // Perform arithmetic Operations
11         sum = number1 + number2;
12         difference = number1 - number2;
13         product = number1 * number2;
14         quotient = number1 / number2;
15         remainder = number1 % number2;
16
17         // Print results
18         System.out.print("The sum, difference, product, quotient and remainder of "); // Print description
19         System.out.print(number1);           // Print the value of the variable
20         System.out.print(" and ");
21         System.out.print(number2);
22         System.out.print(" are ");
23         System.out.print(sum);
24         System.out.print(", ");
25         System.out.print(difference);
26         System.out.print(", ");
27         System.out.print(product);
28         System.out.print(", ");
29         System.out.print(quotient);
30         System.out.print(", and ");
31         System.out.println(remainder);
32
33         ++number1; // Increment the value stored in the variable "number1" by 1
34                 // Same as "number1 = number1 + 1"
35         --number2; // Decrement the value stored in the variable "number2" by 1
36                 // Same as "number2 = number2 - 1"
37         System.out.println("number1 after increment is " + number1); // Print description and variable
38         System.out.println("number2 after decrement is " + number2);
39         quotient = number1 / number2;
40         System.out.println("The new quotient of " + number1 + " and " + number2
41             + " is " + quotient);
42     }
43 }

```

```

The sum, difference, product, quotient and remainder of 98 and 5 are 103, 93, 490, 19, and 3
number1 after increment is 99
number2 after decrement is 4
The new quotient of 99 and 4 is 24

```

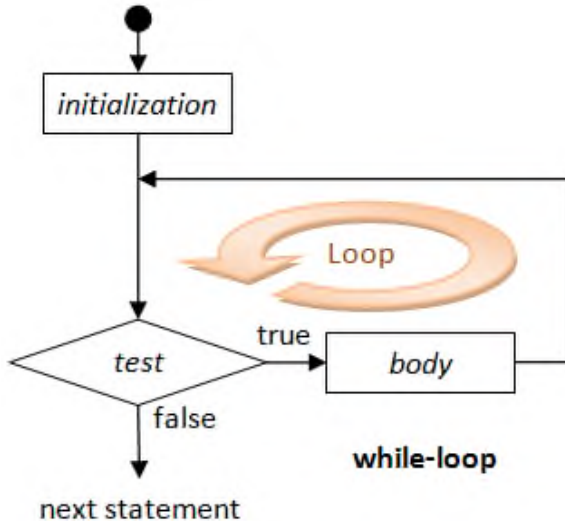
Binlerce sayı ile işlem yapmanız gerekirse!

- 1'den 1000'e kadar olan tüm tam sayıları eklemek istediğinizi varsayalım. Önceki örneği izlerseniz, bin satırlık bir programa ihtiyacınız olacaktır! Bunun yerine, tekrar eden bir görevi gerçekleştirmek için programınızda döngü denen bir şey kullanabilirsiniz.

```
1  /*
2   * Sum from a lowerbound to an upperbound using a while-loop
3   */
4  public class RunningNumberSum { // Save as "RunningNumberSum.java"
5      public static void main(String[] args) {
6          int lowerbound = 1; // Store the lowerbound
7          int upperbound = 1000; // Store the upperbound
8          int sum = 0; // Declare an int variable "sum" to accumulate the numbers
9              // Set the initial sum to 0
10         // Use a while-loop to repeatedly sum from the lowerbound to the upperbound
11         int number = lowerbound;
12         while (number <= upperbound) {
13             // number = lowerbound, lowerbound+1, lowerbound+2, ..., upperbound for each iteration
14             sum = sum + number; // Accumulate number into sum
15             ++number; // increment number
16         }
17         // Print the result
18         System.out.println("The sum from " + lowerbound + " to " + upperbound + " is " + sum);
19     }
20 }
```

The sum from 1 to 1000 is 500500

Döngüler nasıl çalışır?



```
initialization-statement;  
while (test) {  
    Loop-body;  
}  
next-statement;
```

Koşullar ve Kararlar

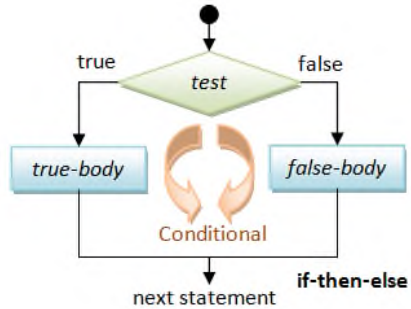
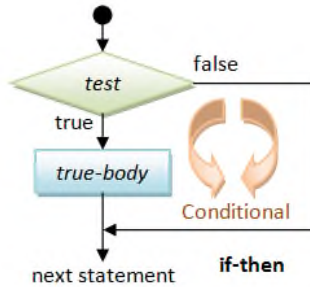
- Ya tüm tek sayıları ve ayrıca 1 ile 1000 arasındaki tüm çift sayıları toplamak isterseniz?

```
1  /*
2  * Sum the odd numbers and the even numbers from a lowerbound to an upperbound
3  */
4  public class OddEvenSum { // Save as "OddEvenSum.java"
5      public static void main(String[] args) {
6          int lowerbound = 1, upperbound = 1000; // lowerbound and upperbound
7          int sumOdd = 0; // For accumulating odd numbers, init to 0
8          int sumEven = 0; // For accumulating even numbers, init to 0
9          int number = lowerbound;
10         while (number <= upperbound) {
11             // number = lowerbound, lowerbound+1, lowerbound+2, ..., upperbound for each iteration
12             if (number % 2 == 0) { // Even
13                 sumEven += number; // Same as sumEven = sumEven + number
14             } else { // Odd
15                 sumOdd += number; // Same as sumOdd = sumOdd + number
16             }
17             ++number; // Next number
18         }
19         // Print the result
20         System.out.println("The sum of odd numbers from " + lowerbound + " to " + upperbound + " is " + sumOdd);
21         System.out.println("The sum of even numbers from " + lowerbound + " to " + upperbound + " is " + sumEven);
22         System.out.println("The difference between the two sums is " + (sumOdd - sumEven));
23     }
24 }
25 }
```

```
The sum of odd numbers from 1 to 1000 is 250000
The sum of even numbers from 1 to 1000 is 250500
The difference between the two sums is -500
```


Koşullar ve Kararlar

- Ya tüm tek sayıları ve ayrıca 1 ile 1000 arasındaki tüm çift sayıları toplamak isterseniz?



```
// if-then
if ( test ) {
    true-body;
}

// if-then-else
if ( test ) {
    true-body;
} else {
    false-body;
}
```

Karşılaştırma ve Karşılaştırma Operatörlerini Birleştirme Operatörleri

| Operator | Mode | Usage | Meaning |
|----------|--------|--------|--------------------------|
| == | Binary | x == y | Equal to |
| != | Binary | x != y | Not equal to |
| > | Binary | x > y | Greater than |
| >= | Binary | x >= y | Greater than or equal to |
| < | Binary | x < y | Less than |
| <= | Binary | x <= y | Less than or equal to |

| Operator | Mode | Usage | Meaning | Example |
|----------|--------------|--------|-------------|------------------------|
| && | Binary | x && y | Logical AND | (x >= 1) && (x <= 100) |
| | Binary | x y | Logical OR | (x < 1) (x > 100) |
| ! | Unary Prefix | !x | Logical NOT | !(x == 8) |

KAYNAKLAR

- ▶ Programming Notes. (2021, March 11). Retrieved from <https://www3.ntu.edu.sg/home/ehchua/programming/index.html>