Tutorial 3
CSC 201

Java Programming Concepts

مبادئ البرمجة باستخدام الجافا

Chapter 3

1. Handling Input and Output

2. Input Stream & Output Stream

3. System.in & System.out

4. A Sample IO Program

5. Integer Input

Handling Input and Output

التعامل مع إدخال وإخراج البيانات

To handle IO ,You will need to import the package java.io

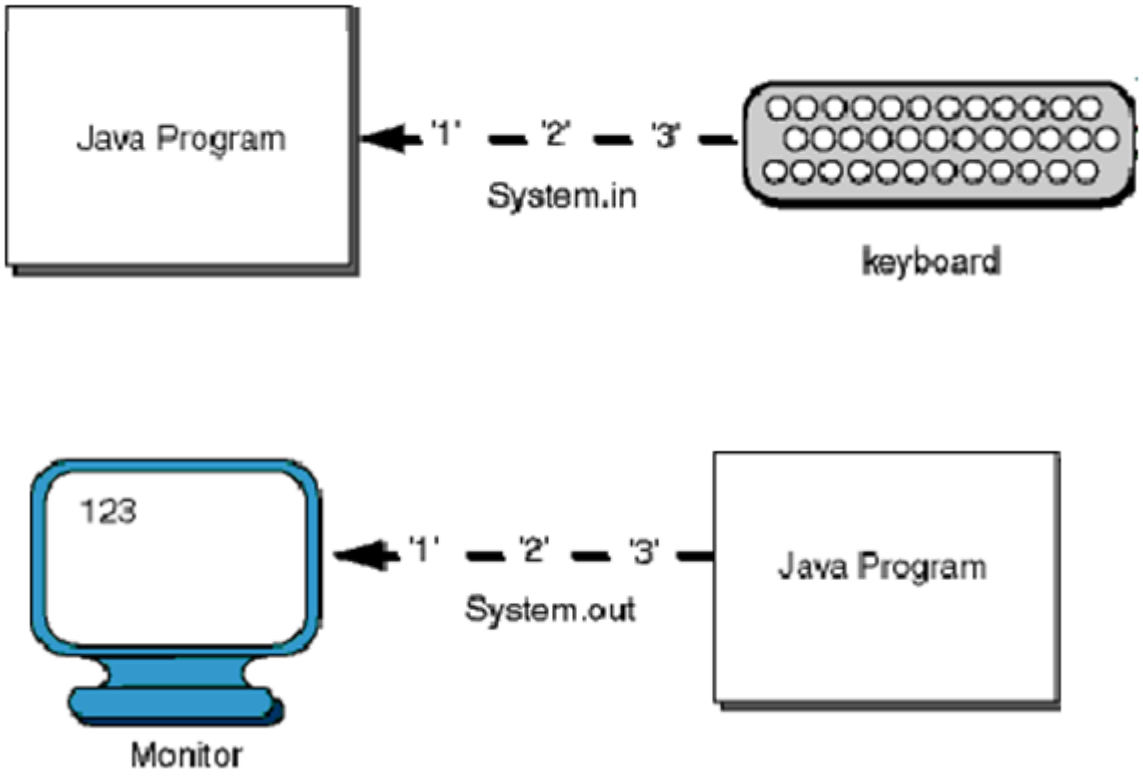
```
import java.io.*;
public class Echo
{
    public static void main (String[] args) throws IOException
  {
     String inData;
     InputStreamReader inStream = new InputStreamReader( System.in ) ;
     BufferedReader stdin = new BufferedReader( inStream );

     System.out.println("Enter the data:");
     inData= stdin.readLine();
     System.out.println("You entered:" + inData);
  }
}
```

Input stream and Output stream

" System.in - the input stream. دالة الإدخال

" System.out - the output stream for normal results. دالة الإخراج



System.in & System.out

Sample IO program

The program reads characters from the keyboard into the String called inData. Then the characters stored in that String are sent to the monitor.

import java.io.;*

Use package java.io - .* any class inside the package

throws IOException

Passing the exception on, an object contains information about input failure

InputStreamReader inStream = new InputStreamReader(System.in);

BufferedReader stdin = new BufferedReader(inStream);

Or

BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));

create a BufferedReader which is an object that does input from the keyboard. System.in gets characters from the keyboard. The InputStreamReader reads the characters from System.in and hands them to the BufferedReader. The BufferedReader objects hands the data to your program.

inData= stdin.readLine();

It gets a line of characters, and assigns them to the String variable str1.

Integer Input:

n = Integer.parseInt(stdin.readLine());

Float Input:

num = Double.parseDouble(stdin.readLine());

System.in & System.out

```
import java.io.*;
public class ForExample1
{
   public static void main(String[] args) throws IOException
    {
      String inData;
      InputStreamReader inStream = new InputStreamReader( System.in );
      BufferedReader stdin = new BufferedReader( inStream );

      System.out.print("Please Enter data: ");
      inData = stdin.readLine();

      System.out.print(“The entered data : “+inData);

    }
}
```

Chapter 4

Control statements:

أوامر التحكم في مسار تنفيذ البرنامج

In general, a Control Statement is a statement that influences the order of the execution of a program.

Java provides several control statements, which allow us to do so:

1) **Conditional Statements:**

الجمل الشرطية

- if statement
- conditional operator (?)
- switch statement

2) **Iterative Statements (Loops):**

الجمل التكرارية – الحلقات

- do loop
- while loop
- for loop

The if statement

General Syntax:

```
if (condition)
   statement;
```

A Simple Example:

```
if (score>98)
   grade='A';
```

General Syntax of the if-else statement:

```
if  (condition)
    statement1
else
    statement2;
```

General Syntax of the multiple if-then Statements

```
if (condition)
   statement;
else if(condition)
   statement;
else if(condition)
    statement;
          .
          .
          .
else
    statement;
```

Examples

A General Decision-making Situation موقف اتخاذ قرار

We can express this situation, using the if-statement in the following manner:

```
if (whether == "rainy")
  System.out.println("Wear rain coat");
else
  System.out.println("Do not wear rain coat");
```

Example1

```
if (number < 0)
   System.out.println("Number is negative");
else
   System.out.println("Number is not negative");
```

Examples

Example2

We want to print out the correct grade, base on the exam score.

```
public class IfExample1
{
   public static void main(String[] args)
   {
      int score = 78;
      char grade;
       if (score >= 90)
            grade = 'A';
       else if (score >= 80)
            grade = 'B';
       else if (score >= 70)
            grade = 'C';
       else if (score >= 60)
            grade = 'D';
       else
            grade = 'F';

       System.out.println("Grade = " + grade);
       }
 }
```

Sample Run:

Grade = C

Examples

Example 3

We want to print out whether x is greater than y, depending on their values.

```
import java.io.*;
public class IfExample2
{
 public static void main(String args[]) throws IOException
 {
  int x,y;
  InputStreamReader inStream = new InputStreamReader( System.in ) ;
  BufferedReader stdin = new BufferedReader( inStream );

  System.out.print("Enter x:");
  x = Integer.parseInt(stdin.readLine());

  System.out.print("Enter y:");
  y = Integer.parseInt(stdin.readLine());

  if(x>y)          // if x > y then the following statement will be executed
    System.out.println(" x > y");
  else             // if x <= y then the following statement is executed.
    System.out.println(" x <= y ");
   }
}
```

Sample Run:

Enter x: 20

Enter y: 11

x > y

Blocks

What is a block?

A block is a group of statements.

It can be used anywhere a single statement is allowed.

How are blocks denoted?

We place the sequence of statements between two curly braces:

```
{
    stmt1;
    stmt1;
    stmt2;
      .
      .
      .
    stmtn;
}
```

A Simple Example

Look at the following code:

```
if (num<0)
        System.out.println("The number "+num+" is negative");
else
        System.out.println("The number "+ num + " is positive");
        System.out.println("positive numbers are greater ");
        System.out.println("or equal to zero ");
System.out.println("Good-bye for now");
```

Is this program correct?

No.

To correct this code, the programmer needs to use curly braces:

```
if (num<0)
        System.out.println("The number "+num+" is negative");
else
{
        System.out.println("The number "+ num + " is positive");
        System.out.println("positive numbers are greater ");
        System.out.println("or equal to zero ");
}
System.out.println("Good-bye for now");
```

```
public class IfBlockExample
{
   public static void main(String args[])
   {
      int i = 11;
      int j = 20;

     if (i>j)      // if i > j then the following block of statements will be executed
      {
        System.out.println(" i = "+i+" j= "+j);
        System.out.println(" i > j");
      }
     else       // if i <= j then the following block of statements will be executed
      {
       System.out.println(" i = "+ i +" j = "+j);
       System.out.println(" i <= j ");
      }
   }
}
```

Sample Run:

```
i =11 j = 20
i <= j
```

Nested ifs

In **nested if** statements, we declare an **if** statement within the declaration of another **if** statement.

General Syntax

```
if (condition1)
{
      if (condition2)
      {
        statement block;
       }
      statement1;
 }
else
      statement2;
```

A Simple Example

```
if (x ! = 30)            // if x is 30 then the following block of statements will be executed
{
      if (j <40)       a = b;
      else             a = c;
}
else      a = d;        //if x is not equal to 30 then the following statement will be executed
```

Example: Nested ifs

Find out which *else* goes with which *if*

```
public class NestedIfExample
{
  public static void main(String args[])
  {
    int i = 29, j = 20;
    if(i = = 29)
    {
      if(j < 20)            System.out.println(" i = 29 and j < 20");
      if(j > 20)            System.out.println(" i = 29 and j > 20");
      else            // this else is associated with if (j>20)
      System.out.println(" i = 29 and j is not greater than 20");
    }
    else     // this else statement is outside the block. It belongs to the if (i = = 29) statement
       System.out.println("i is not equal to 29");
    System.out.println("Execution continues from here");
  }
}
```

Program Output:

i = 29 and j not greater than 20
Execution continues from here

The Conditional Operator (?) المعامل الشرطي

The ? operator is a conditional operator that is short-hand for an if-else statement:

General Syntax

condition ? expression1:expression2

The ? operator returns expression1 if condition is true or
returns expression2 if condition is false.

A Simple Example

```
x = num<0  ? -num : num
```

Here, we want to calculate the absolute value of num.
If the condition (num <0) is true, then x = -num If it is false, then x = num.

This is equivalent to writing the following if-else statement:

```
if (num<0)
   x = -num;
else
   x = num;
```

Example: Conditional operator

The following program takes a number from the user, and calculates its absolute value, using the conditional operator.

```
import java.io.*;
class AbsoluteValue
{
   public static void main(String args[]) throws IOException
  {
   InputStreamReader inStream = new InputStreamReader( System.in ) ;
   BufferedReader stdin = new BufferedReader( inStream );

   int num,abs;

   System.out.print("Enter number: ");
   num = Integer.parseInt(stdin.readLine());

   abs = num<0 ? -num : num;
   System.out.println("Absolute value of  "num + " is " + abs);
  }
}
```

Sample Run:

```
Enter number: -5
Absolute value of -5 is 5
```

The switch statement

General Syntax

```
switch(expression1)
{
   case value1:
    //statement sequence
   //break;
   case value2:
   //statement sequence
   //break;
  case value3:
  //statement sequence
  //break;
  default:
  //default statement sequence
}
```

Depending on the value of the expression, the corresponding case gets executed.

NOTE:

1- The **default** statement is executed if no case statements match.

2- The **break** takes control out of the block; if it is omitted, execution continues to the next case (see the example below).

If-else and switch-case statements:

```
If (x == 0)
    doSomething0();
else if (x == 1)
    doSomething1();
else if (x == 2)
    doSomething2();
else if (x == 3)
    doSomething3();
else if (x == 4)
    doSomething4();
else
    doSomethingElse();
```

```
switch (x)
{
 case 0:
    doSomething0(); break;
 case 1:
    doSomething1(); break;
 case 2:
    doSomething2(); break;
 case 3:
    doSomething3(); break;
 case 4:
    doSomething4(); break;
 default:
    doSomethingElse();
}
```

Example1

The code displays the name of the month, based on the value of month, using the switch statement.

```
public class SwitchExample1
{
public static void main(String[] args)
 {
   int month = 8;
   String monthName;
   switch (month)
   { case 1: monthName = "January"; break;
      case 2: monthName = "February"; break;
      case 3: monthName = "March"; break;
      case 4: monthName = "April"; break;
      case 5: monthName = "May"; break;
      case 6: monthName = "June"; break;
      case 7: monthName = "July"; break;
      case 8: monthName = "August"; break;
      case 9: monthName = "September"; break;
      case 10: monthName = "October"; break;
      case 11: monthName = "November"; break;
      case 12: monthName = "December"; break;
      default: monthName = "Invalid month";    }
    System.out.println(monthString);
  }
}
```

Example 2:

Write a C++ program to read a character and print a message “it is a vowel” if character is one of (a,e,i,o,u), “it is an operator” if character is (+,-,*,/) or “it is something else” if any other character. Use switch – case.

```
class SwitchExample2
{
  public static void main(String args[])
  {
      char ch = ‘e’;
      switch(ch)
       {
          case ‘a’: case ‘e’: case ‘i’: case ‘o’: case ‘u’:
                    System.out.println(" it is a vowel"); break;
           case ‘+’: case ‘-’: case ‘*’: case ‘/’:
                    System.out.println(" it is an operator"); break;
          default:
                    System.out.println(" It is something else");
      }
  }
}
```