

Lab 02

Java Fundamentals-I

Objective:

This lab is designed to give you practice with some of the fundamentals concepts in Java.

Activity Outcomes:

On completion of this lab student will be able

- Identify basic elements in java program (comments, reserve words, identifiers etc.)
- Use variables and constants in program development
- Communicate with the user by using the Scanner class
- Create a program from scratch by translating a pseudocode algorithm

Instructor Note:

As a pre-lab activity, read Chapter 02 from the text book “Java How to Program, Deitel, P. & Deitel, H., Prentice Hall, 2019”.

1) Useful Concepts

To write meaningful programs, you must learn the programming language's special symbols, words, and syntax rules. The syntax rules tell you which statements (instructions) are legal, or accepted by the programming language, and which are not. You must also learn the semantic rules, which determine the meaning of the instructions. The programming language's rules, symbols, special words, and their meanings enable you to write programs to solve problems.

Comments

Single-line comments begin with `//` and can be placed anywhere in the line. Multiline comments

```
System.out.println("7 + 8 = " + (7 + 8)); //prints: 7 + 8 = 15
```

Multiple-line comments are enclosed between `/*` and `*/`.

```
/*  
You can include comments that can  
occupy several lines.  
*/
```

Special Symbols

The following are some of the special symbols:

```
+      -      *      /  
.      ;      ?      ,  
<=     !=     ==     >=
```

Reserved Words (Keywords)

Reserved words are also called keywords used by programming language

```
int, float, double , char, void, public, static, throws , return
```

Identifiers

Identifiers are names of things, such as variables, constants, and methods, that appear in programs. Some identifiers are predefined; others are defined by the user. A Java identifier consists of letters, digits, the underscore character (`_`), and the dollar sign (`$`) and must begin with a letter, underscore, or the dollar sign

User defined identifiers

```
First, conversion,  
payRate, counter1,  
$Amount
```

Pre-defined identifiers

```
print , println , and  
printf , nextInt,  
nextDouble , next , and  
nextLine
```

Data Types

A set of values together with a set of operations on those values.

Data Type	Values	Storage (in bytes)
<code>char</code>	0 to 65535 ($= 2^{16} - 1$)	2 (16 bits)
<code>byte</code>	-128 ($= -2^7$) to 127 ($= 2^7 - 1$)	1 (8 bits)
<code>short</code>	-32768 ($= -2^{15}$) to 32767 ($= 2^{15} - 1$)	2 (16 bits)
<code>int</code>	-2147483648 ($= -2^{31}$) to 2147483647 ($= 2^{31} - 1$)	4 (32 bits)
<code>long</code>	-9223372036854775808 ($= -2^{63}$) to 9223372036854775807 ($= 2^{63} - 1$)	8 (64 bits)

Variables: A memory location whose content may change during program execution.

Syntax

```
dataType identifier1, identifier2, ..., identifierN;
```

Named constant: A memory location whose content is not allowed to change during program execution.

Syntax

```
static final dataType IDENTIFIER = value;
```

Putting Data into Variables

```
variable = expression;
```

Declaring and Initializing Variables

In order to use variables in program, you need to declare and initialize it properly.

```
int first;  
int second;  
char ch;  
double x;  
first = 13;  
second = 10;  
ch = ' ';  
x = 12.6;
```

You can also declare and initialize at the same line

```
int first = 13;  
int second = 10;  
char ch = ' ';  
double x = 12.6;  
double y = 123.456;
```

Declaring String variable

```
String name;
```

Reading Data using the Scanner class

To put data into variables from the standard input device, Java provides the class Scanner. Using this class, we first create an input stream object and associate it with the standard input device. The following statement accomplishes this:

```
Scanner console = new Scanner(System.in);
```

To use the scanner class first you need to import following package

```
Syntax: import packageName.*;
```

```
import java.util.Scanner;
```

nextByte()	Reads an integer of the byte type.
nextShort()	Reads an integer of the short type
nextInt()	Reads an integer of the int type.
nextLong()	Reads an integer of the long type.
nextFloat()	Reads a number of the float type.
nextDouble()	Reads a number of the double type.
next().charAt(0)	Read a character
next()	Read a string
nextLine()	Read a line

Displaying Output

In Java, output on the standard output device is accomplished by using the standard output object **System.out**. The object **System.out** has access to two methods, **print** and **println**, to output a string on the standard output device

Syntax

```
System.out.print(expression);
```

```
System.out.println(expression);
```

```
System.out.println();
```

Displaying Multiple items with the + operator

String concatenation operator appends one string to another

```
System.out.println("This is " + "one string.");
```

This statement will print:

```
This is one string
```

```
number = 5;
```

```
System.out.println("The value is " + number);
```

```
The value is 5
```

Escape Sequences

	Escape Sequence	Description
<code>\n</code>	Newline	Cursor moves to the beginning of the next line
<code>\t</code>	Tab	Cursor moves to the next tab stop
<code>\b</code>	Backspace	Cursor moves one space to the left
<code>\r</code>	Return	Cursor moves to the beginning of the current line (not the next line)
<code>\\</code>	Backslash	Backslash is printed
<code>\'</code>	Single quotation	Single quotation mark is printed
<code>\"</code>	Double quotation	Double quotation mark is printed

2) Solved Lab Activities

<i>Sr.No</i>	<i>Allocated Time</i>	<i>Level of Complexity</i>	<i>CLO Mapping</i>
<i>Activity 1</i>	<i>15 mins</i>	<i>Low</i>	<i>CLO-5</i>
<i>Activity 2</i>	<i>15 mins</i>	<i>Low</i>	<i>CLO-5</i>
<i>Activity 3</i>	<i>15 mins</i>	<i>Low</i>	<i>CLO-5</i>
<i>Activity 4</i>	<i>15 mins</i>	<i>Low</i>	<i>CLO-5</i>
<i>Activity 5</i>	<i>15 mins</i>	<i>Low</i>	<i>CLO-5</i>

Activity 1:

This program illustrates how data in the variables are manipulated. Translate the instruction as given below

Solution:

```
public class Activity1{  
    public static void main(String [] args){  
        // Declare an variable num1 of type int  
        // Declare an variable num2 of type int  
        // Declare an variable sale of type double  
        // Declare an variable first of type char  
        // Assign 4 value to num1  
        // Display the value in num1 variable on output screen  
        // Store the result of 4 * 5 - 11 expression in num2 variable  
        // Display the value in num2 variable on output screen  
        // Store the result of 0.02 * 1000 expression in sale variable  
        // Display the value in sale variable on output screen  
        // Assign 'D' value to a variable first  
        // Display the value in first variable on output screen  
    }  
}
```

```

public class Activity1{

    public static void main(String [] args){

        int num1;

        int num2;

        double sale;

        char first;

        num1 = 4;

        System.out.println("num1 = " + num1);

        num2 = 4 * 5 - 11;

        System.out.println("num2 = " + num2);

        sale = 0.02 * 1000;

        System.out.println("sale = " + sale);

        first = 'D';

        System.out.println("first = " + first);

    }

}

```

Output

```

num1 = 4
num2 = 9
sale = 20.0
first = D

```

Activity 2:

This program illustrates how input statements work.

Solution:

```

import java.util.Scanner;

public class Activity2{

    public static void main(String [] args){

        Scanner console = new Scanner(System.in);

        int feet;

        int inches;

        System.out.print("Enter two integers separated by spaces.");

        feet = console.nextInt();
    }

}

```

```

        inches = console.nextInt();
        System.out.print("feet = " + feet);
        System.out.print(" inches = " + inches);
    }
}

```

Output

```

Enter two integers separated by spaces. 23 7
feet = 23 inches = 7

```

Activity 3:

This program illustrates how to read strings and numeric data

Solution:

```

import java.util.*;
public class Activity3{
    static Scanner console = new Scanner(System.in);
    public static void main(String [] args){
        String firstName;
        String lastName;
        int age;
        double weight;
        System.out.println("Enter first name, last name, " + "age, and
weight separated " + "by spaces.");
        firstName = console.next();
        lastName = console.next();
        age = console.nextInt();
        weight = console.nextDouble();
        System.out.print("Name: " + firstName+ " " + lastName);
        System.out.print(" Age: " + age);
        System.out.print(" Weight: " + weight);
    }
}

```

Output

```

Enter first name, last name, age, and weight separated by spaces.
Sheila Mann 23 120.5
Name: Sheila Mann Age: 23 Weight: 120.5

```

Activity 4:

This program demonstrates the close relationship between characters and integers

Solution:

```
public class Activity4{
    public static void main(String[] args){
        char letter;
        letter = 65;
        System.out.println(letter);
        letter = 66;
        System.out.println(letter);
    }
}
```

Output

```
A
B
```

Activity 5:

This program illustrate the concept of constants used in java

Solution

```
import java.util.Scanner;
public class ComputeAreaWithConstant {
    public static void main(String[] args) {

        final double PI = 3.14159; // Declare a constant

        Scanner input = new Scanner(System.in);

        System.out.print("Enter a number for radius: ");
        double radius = input.nextDouble();

        double area = radius * radius * PI;
        System.out.println("The area for the circle of radius " +
radius + " is " + area);
    }
}
```

Output

```
Enter a number for radius: 1
The area for the circle of radius is 3.14159
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Consider the following program segment

```
//import classes
public class Activity1
{
    public static void main(String [] args)
    {
        //variable declaration
        //executable statements
    }
}
```

- a. Write Java statements that declare the following variables: **num1** , **num2** , and **num3** , and **average** of type **int** .
- b. Write Java statements that store **125** into **num1** , **28** into **num2** , and **-25** into **num3** .
- c. Write a Java statement that stores the average of **num1** , **num2** , and **num3** into **average**.
- d. Write Java statements that output the values of **num1** , **num2** , **num3** , and **average** .
- e. Compile and run your program

Lab Task 2

Consider the following Java program in which the statements are in the incorrect order. Rearrange and format the statements so that it prompts the user to input the length and width of a rectangle and output the area and perimeter of the rectangle.

```
public class Activity2
{
    Scanner console = new Scanner(System.in);
    import java.util.*;
    {
        public static void main(String[] args)
        int width;
        System.out.print("Enter the length: ");
        width = console.nextInt();
        System.out.println();
        int length;
        System.out.print("Enter the width: ");
        length = console.nextInt();
        System.out.println();
        area = length * width;
```

```

System.out.println("Area = " + area);
System.out.println("Perimeter = " + perimeter);
perimeter = 2 * (length + width);
int area;
int perimeter;
}}

```

Lab Task 3

Consider the following program segment:

```

//import classes
public class LabTask3{
    public static void main(String [] args){
        //variable declaration
        //executable statements
    }
}

```

- a) Write a Java statement that imports the class Scanner.
- b) Write a Java statement that declares **console** to be a Scanner object for inputting data from the standard input device.
- c) Write Java statements that declare and initialize the following named constants: **SECRET** of type **int** initialized to 11; **RATE** of type **double** initialized to 12.50.
- d) Write Java statements that declare the following variables: **num1** , **num2** , and **newNum** of type **int** ; name of type **String**; **hoursWorked** and **wages** of type **double**.
- e) Write Java statements that prompt the user to input two integers and store the first number into **num1** and the second number into **num2**.
- f) Write a Java statement(s) that outputs the value of **num1** and **num2** , indicating which is **num1** and which is **num2**. For example, if **num1** is 8 and **num2** is 5 , then the output is:

The value of num1 = 8 and the value of num2 = 5.
- g) Write a Java statement that multiplies that value of **num1** by 2 , adds the value of **num2** to it, and then stores the result in **newNum** . Then write a Java statement that outputs the value of **newNum**
- h) Write a Java statement that updates the value of **newNum** by adding the value of the named constant **SECRET** . Then, write a Java statement that outputs the value of **newNum** with an appropriate message.
- i) Write Java statements that prompt the user to enter a person's last name and then store the last name into the variable **name**.
- j) Write Java statements that prompt the user to enter a decimal number between 0 and 70 and then store the number entered into **hoursWorked**.
- k) Write a Java statement that multiplies that value of the named constant **RATE** with the value of **hoursWorked** and stores the result into the variable **wages** .
- l) Write Java statements that produce the following output:

```

Name: //output the value of the variable name
Pay Rate: $ //output the value of the named constant RATE

```

```
Hours Worked: //output the value of the variable hoursWorked
Salary: $ //output the value of the variable wages
```

For example, if the value of name is "Rainbow" and hoursWorked is 45.50 , then the output is:

```
Name: Rainbow
Pay Rate: $12.50
Hours Worked: 45.50
Salary: $568.75
```

- m) Write a Java program that tests each of the Java statements in parts (a)—(l). Place the statements at the appropriate place in the preceding Java program segment. Test run your program (twice) on the following input data:

- a. num1 = 13, num2 = 28; name = "Mustafa"; hoursWorked = 48.30
- b. num1 = 32, num 2 = 15; name = "Shakeel"; hoursWorked = 58.45