AST10106

Introduction to Programming JAVA

Semester A 2015/16

Lab03
Basic Java Input / Output

Submission Details

In this lab, you are required to write **TWO** Java programs to solve the given problems shown in the section "Lab Exercises". After you have completed the implementation using **Eclipse**, submit the program file of second task (i.e. CylinderCal.java) to the assignment page of Canvas. The deadline would be **ONE week** after your lab session conducted. You are reminded to double-check your solution to verify everything in correct before submission.

Note: You are required to put your name, student ID and lab section at the beginning of your source file as the comment.

Objective

The objectives of this lab are (1) to reinforce the programming concepts (e.g. variable declarations) taught in the last lecture; (2) to introduce you the basic concept of getting user's inputs from keyboard; (3) to demonstrate the basic arithmetic operations; and (4) to show you how to display the information on screen.

Software Required

1. Java Development Kit, eg JavaSE 1.8
2. An Integrated Development Environment, e.g. Eclipse

Please start your PC with Window 7 and refer to the previous lab exercise for the installation and operation guide of Eclipse!

Introduction

In this lab, you will practice:
1. To obtain input from the user using the Scanner class and store them in variables.
2. To display information to the user using print and println.
3. To perform arithmetic calculations using arithmetic operations.

Background

1. Display information to the user:

- Java provides the following two operations (actually there is one more, but we only focus on the following two at the moment) to display on monitor's screen.

  i. System.out.print()
  ii. System.out.println()

- To use these operations, you can follow the syntax below

  Syntax:

  // Print data on screen and the cursor will stay at the end of the string printed
  System.out.print(<data> [+ <data>]);
  // Print the data on screen and move the cursor to the beginning of the next line
  System.out.println(<data> [+ <data>]);
  // Print nothing
  System.out.print("");
  // Print nothing, but move the cursor to the beginning of the next line
  System.out.println("");

- **Example**

```
public class OutputDataDemonstration
{
    public static void main(String[] args)
    {
        byte val1 = 10;
        short val2 = 412;
        int val3 = 58123;
        long val4 = 313123152;
        System.out.println("val1 & val2: " + val1 + " " + val2);
        System.out.println("val3 & val4: " + val3 + " " + val4);
        System.out.println();
        float val5 = 1.2345f;
        double val6 = 2.331256451313;
        System.out.println("val5 & val6: " + val5 + " " + val6);
        System.out.println();
        char val7 = 'D';
        System.out.println("Character in val7 is " + val7);
        System.out.println();
        boolean val8 = true;
        System.out.println("Boolean value in val8 is " + val8);
        System.out.println();
        System.out.print("Line A ");
        System.out.println("Continue to display data on line A");
```

2. *Obtain input from user and store them in variables:*

- Java provides a class named "Scanner" for getting user's input from various input device, e.g. keyboard.
- The Scanner class is part of the package java.util. A package in Java is actually a collection of classes with a certain name. In this example, java.util is a package with so many classes inside, one of them is Scanner class.

- **Syntax:**

```
// Get the class definition of the Scanner class from java.util package
import java.util.Scanner;
// Create a Scanner object with name <object name>
// System.in is the "standard" input stream, which is keyboard
Scanner <object name> = new Scanner(System.in);
// Use the Scanner class's operations (methods) to read input
<variable name> = <object name>.<method name>();
```

- A brief summary of Scanner class operations:

Method name	**Description**
boolean nextBoolean()	Returns the next input token as a boolean value
byte nextByte()	Returns the next input token as a byte value
short nextShort()	Returns the next input token as a short value
int nextInt()	Returns the next input token as a int value
long nextLong()	Returns the next input token as a long value
float nextFloat()	Returns the next input token as a float value
double nextDouble()	Returns the next input token as a double value
String next()	Returns the next input token as a String value
String nextLine()	Returns all input remaining on the current line as a String value

- **Example**

```
import java.util.Scanner;
public class InputDataDemonstration
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Please enter an integer: ");
        int intVal = sc.nextInt();
        System.out.println("The value you entered is " + intVal);
    }
}
```

3. *Perform calculation using arithmetic operations:*

- Java provides five basic operations for arithmetic calculation.
  i. Addition (The operator used is +)
  ii. Subtraction (The operator used is -)
  iii. Multiplication (The operator used is *)
  iv. Division (The operator used is /)
  v. Remainder (The operator used is %)

- **Syntax:**

```
// Add the two values stored in variable 1 and variable 2
<name of variable for the result> = <name of variable 1> + <name of variable 2>;
// Subtract the value stored in variable 2 from variable 1
<name of variable for the result> = <name of variable 1> - <name of variable 2>;
// Multiply the value stored in variable 1 by the value stored in variable 2
<name of variable for the result> = <name of variable 1> * <name of variable 2>;
// Divide the value stored in variable 1 by the value stored in variable 2
<name of variable for the result> = <name of variable 1> / <name of variable 2>;
// Find the remainder of value stored in variable 1 divided by the value stored in
// variable 2
<name of variable for the result> = <name of variable 1> % <name of variable 2>;
```

- **Example:**

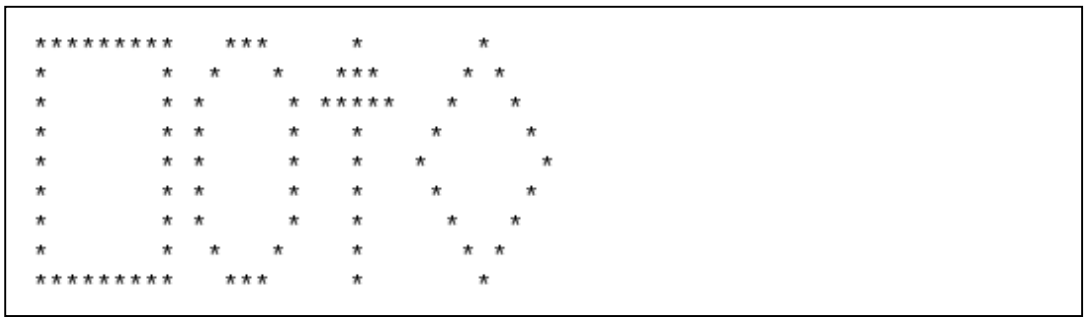
```
public class ArithmeticOperationsDemonstration
{
    public static void main(String[] args)
    {
        int val1 = 21;
        int val2 = 10;
        int sum = val1 + val2;
        int diff = val1 - val2;
        int prod = val1 * val2;
        int quot = val1 / val2;
        int rem = val1 % val2;

        System.out.println("Addition: " + sum);
        System.out.println("Subtraction: " + diff);
        System.out.println("Multiplication: " + prod);
        System.out.println("Division: " + quot);
        System.out.println("Remainder: " + rem);
    }
}
```

Lab Exercises

In this lab, you will be asked to solve **TWO** problems using Java. For these problems, write Java programs according to the following problem specifications.

1. Write a Java program that displays the shapes shown in the sample output using asterisks.



The template of the program has been given for you as follow.

```
1   public class Shapes
2   { // start class Shapes
3       public static void main(String[] args)
4       { // start main
5           /* write a series of statements that will print the
6              shapes to the command window */
7       } // end main
8   } // end class Shapes
```

2. Write a Java program **CylinderCal.java** that inputs from the user the radius and the height of a cylinder double and prints the cylinder's surface area and volume using the floating-point value 3.14159 for π. You may use the predefined constant **Math.PI** for the value of π. This constant is more precise than the value 3.14159. Class Math is defined in package java.lang in that package are imported automatically, so you do not need to import class Math to use it. Using the following formulas (r is the radius, h is the height):

Surface area = 2π rh
Volume = $\pi r^2 h$

(Note: The underlined text refers to the user input. Also, please follow closely to the output format shown below.)

Sample Output:

```
Enter radius of cylinder: 3
Enter height of cylinder: 5
Surface area is 94.24777960769379
Volume is 141.3716694115407
```

Marking Scheme

The marking of this exercise will be based on the following criteria.

Graded items	Weighting
1. Correctness of program (i.e. whether your code is implemented in a way according to the requirements as specified.)	50%
2. Presentation of the Java codes (i.e. whether the program is properly indented, how close you follow the common conventions as mentioned in class, etc.)	30%
3. Documentation (with reasonable amount of comments embedded in the code to enhance the readability)	20%
	100%

Be aware of plagiarism! DON'T copy the program file from your friends or classmates. If any identical copy is found, 5% of the coursework marks will be deducted for each of the file owner.

Program Submission Checklist

Before submitting your work, please check the following items to see you have done a decent job.

Items to be checked	☑ / ☒
1. Did I put my name, student ID and lab section at the beginning of all the source files as comment?	☐
2. Did I put reasonable amount of comments to describe my program?	☐
3. Are they all in .java extension and named according to the specification?	☐
4. Have I checked that all the submitted code are compliable and run without any errors?	☐
5. Did I zip my source files using Winzip / zip provided by Microsoft Windows? Also, did I check the zip file and see if it could be opened? ***(Only applicable if the work has to be submitted in zip format.)***	☐
6. Did I submit my lab assignment to Canvas before the deadline?	☐