Object Oriented Programming with Java

Practical List

Last Updated on 09 / 08 / 2017

Lab #	Aim	#Hrs
1	**Programs to demonstrate printing to console and taking input as command line arguments** 1. Write a Java program to display "Hello World" on the console. 2. Write a Java program to take username (e.g. Ravi) as command line argument and display "**Welcome Ravi!**" on the console. 3. Write a Java program to find sum of digits of a number taken as a command line argument. 4. Write a Java program to add two numbers entered as command line argument. [Extend this program to develop simple calculator.]	2
2	**Programs to understand concept of class and methods, use of array and Scanner class.** 1. Write a Java program to find minimum and maximum value from the given array. 2. Write a Java program to demonstrate use of Scanner class. a. Take two whole numbers (suppose x and y) from user and compute xy and display the result. b. Scan three statements (suppose s1, s2, s3). Display the output as s1@@s2##ss3. [s1 is"Good Morning", s2 is "Good Afternoon" and s3 is "Good Night". Output should be Good Morning@@Good Afternoon## Good Night. 3. Do as directed for a given class Student as below **class Student** **{** **int roll_no;** **String name;** **int marks[] = new int[5];** **}** a. Store details of one student by creating one object of Student class and display them. b. Add **double findAverage()** method in the Student class. c. Store details of 3 students by creating array of object of Student class and display the details of the student who has highest average amongst the three students.	2
3	**Programs to demonstrate recursion, constructor overloading, this keyword and passing objects as parameters.**	2

	1. Write a java program to find factorial using recursion. 2. Write a java program as follows: **<u>Box class contains following :</u>** **Properties:** width, height, depth **Constructors:** • **Box() :** which initializes all properties to -1 • **Box(double width, double height, double depth) :** which sets parameter values to corresponding properties. (Hint : use **this** keyword) • **Box(double length) :** which sets length value to all properties in case Box is a cube. • **Box(Box box) :** which sets values of ***box*** properties to corresponding properties of current object. (i.e. copy constructor) **Methods:** • **double findVolume() :** which returns volume of the box calculated using properties of the box. **<u>BoxDemo class :</u>** Create four box objects using each of the constructors and display volume of each box.	
4	**Programs to demonstrate inheritance, method overriding, abstract class and final keyword.** 1. Write a java program containing following classes with mentioned features. • **Person class :** • An instance variable fullName of type String. • A constructor with parameter to initialize fullname. • An abstract method getDescription() with return type String. • A concrete method getName() which returns fullName of that person. • **Employee class:** • An instance variable salary. • A constructor with two parameters full name and salary. • A method getSalary() which returns salary of that employee. • An implementation of getDescription() which returns description of type of that person (i.e. employee) with salary value. • **Student class:** • An instance variable branch. • A constructor with two parameters full name and branch. • A method getBranch() which returns branch in which that student is studying. • An implementation of getDescription() which returns description	2

	of type of that person (i.e. student) with his/her branch name.
	• **DemoAbstract class:**
	  • In main method, create two variables person1 and person2 of Person type.
	  • Initialize person1 with an object of Employee type with necessary values.
	  • Initialize person2 with an object of Student type with necessary values.
	  • Print description of each person.
	• **Sample output :**
	  • Manoj Pandey, an employee with a salary of Rs. 50,000.
	  • Rohini Dave, a student studying in Computer Science.
	
	2. Write a java program with classes Person and Employee as in above problem. Also write classes Manager (subclass of Employee) and DemoFinal as per the following description.
	
	• **Manager class :**
	  • A String instance variable type which stores value of manager type (i.e. HR, General, Finance..).
	  • A constructor with three parameters full name, salary and type.
	  • An instance method increaseSalary(), which takes an amount by which manger's salary will be incremented as a parameter and new increased amount is set as a new value to variable salary.
	  • An overridden method getDescription() which also prints the manager type of that manager.
	
	• **DemoFinal class :**
	  • In main method, create an instance manager1 of type Manager with suitable initial values.
	  • Print description of manager1.
	  • Call increaseSalary() on manager1 by passing some appropriate value.
	  • Print description of manager1 again.
	
	• **Execute a program with different cases :**
	  • Case A: Make an Employee class final.
	  • Case B: Make a method getDescription() in Employee class as a final method.
	  • Case C: Make an instance variable salary in Employee class as a final variable.	

5	**Programs to demonstrate use of interface and multidimensional array**
	
	1. Write a java program as per the given description to demonstrate use of interface.
	  • Define an interface RelationInterface.
	  • Write three abstract methods: isGreater, isLess and isEqual. All methods have
		2

return type of boolean and take an argument of type

- Line with which the caller object will be compared.
- Write Line class implements RelationInterface interface. It has 4 double variables for the x and y co-ordinates of the line.
- Write a constructor in Line class that initializes these 4 variables.
- Write a method getLength() that computes length of the line.
  [double length = Math.sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1))].
- Implement the methods in interface.
- In class CompareLines.java, create two objects of Line class, call the three methods to compare the lengths of the lines.

2. Write a java program as per the given description to demonstrate use of interface for multiple inheritances.
   - Define two interfaces named Terrestrial and Aquatic.
   - In Terrestrial interface, write two methods eats() and walks(). Both methods have return type void and take no arguments.
   - In Aquatic interface, write two methods eats() and swims(). Both methods have return type void and take no arguments.
   - Write a class Amphibian that implements both the interfaces. Write all methods.
   - In DemoMultipleInterface class, create Frog object of Amphibian and call all the methods.
3. Write a program that inputs two 3 X 3 matrices and performs matrix multiplication on them. Display properly formatted output.