Vidush Somany Institute of Technology and Research, Kadi

CT405-N: Object Oriented Programming Using Java

Laboratory Assignment #1

1.1 (a) Write a Java program to find the factorial of a number. Initialize the number in the program itself. (`E.g. int n=5;`)
Solution: https://github.com/hbpatel1976/Java/blob/main/Assignment11a.java

1.1 (b) Write a Java program to find the factorial of a number. Read the number from the command line argument.

```
Example:
C:\>javac factorial.java
C:\>java factorial 5
Answer=120
```

Solution: https://github.com/hbpatel1976/Java/blob/main/Assignment11b.java

1.1 (c) Write a Java program to find factorial of a number. Read the number from the keyboard while running the program.

```
Example:
C:\>javac factorial.java
C:\>java factorial
Enter the number for which you wish to find factorial: 5
Answer=120
```

Solution: https://github.com/hbpatel1976/Java/blob/main/Assignment11c.java

1.2 Following is a sample income tax slab.
Up to Rs.2.5 lakh - No Tax (NIL)
Rs.2.5 lakh - Rs.5 lakh - 5%
Rs.5 lakh - Rs.7.5 lakh - 10%
Rs.7.5 lakh - Rs.10 lakh - 15%
Rs.10 lakh - Rs.12.5 lakh - 20%
Rs.12.5 lakh - Rs.15 lakh - 25%
Above Rs.15 lakh - 30%
Write a Java program which initialized the earning of an employee (E.g. `int income=1950000;`). Your program should display the outcome (E.g. `Income tax to be paid: 3,22,500`)
Solution: https://github.com/hbpatel1976/Java/blob/main/Assignment12.java

1.3 Design a `class Bank` with `accountNo`, `name` and `balance` as three members. The class should have methods viz. (a) `depositMoney`, (b) `withdrawMoney` and (c) `askBalanace`. Write a Java program to create an object of `Bank` class in main function and invoke all the three functions from main to test their working.
Solution: https://github.com/hbpatel1976/Java/blob/main/Assignment13.java

1.4 Create a `class stack` with `data[size]` and `stackTop` as data members and `push`, `pop` and `display` as methods. Write a Java program to create an object of stack class in main and invoke all the three functions from main to test their working.

1.5 Write a Java program to display all the prime numbers between two given numbers. (You may initialize the two numbers **or** read them from keyboard **or** input them as command line arguments)

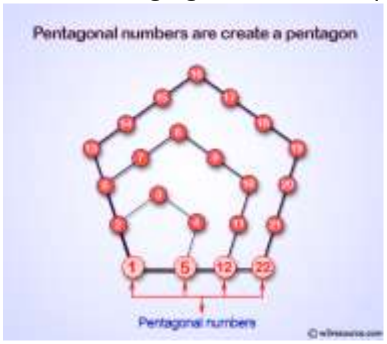
Vidush Somany Institute of Technology and Research, Kadi

CT405-N: Object Oriented Programming Using Java

Laboratory Assignment #2

2.1 The formula for calculating Body Mass Index (BMI) is weight (kg) / [height (m)]^2.
A person is "Underweight" if BMI is below 18.5, "Normal" if BMI is between 18.5 to 24.9, "Overweight" if BMI is between 25.0 to 29.9, and "Obese" if BMI is 30.0 and Above.
Write a Java program that reads "weight" (in kg) and "height" (in meter) and display the status of the person (e.g. "Underweight", "Normal" etc.)

2.2 Following figure shows the pentagonal numbers creation for a typical pentagon. (Source: w3resource.com)



For any number "i", the pentagonal number is computed using the equation [i*(3*i-1)]/2. Hence, write a Java program with *getPentagonalNumber* as method that takes a number "i" as input and returns the equivalent pentagonal number. Print first 4 (n=4) pentagonal numbers (E.g. 1, 5, 12, 22 as shown in the figure).

2.3 An employee gets paid (hours worked) × (basic pay), for each hour up to 40 hours for a week. For every hour over 40, they get overtime = (base pay) × 1.5. The base pay must not be less than the minimum wage (Rs. 150 an hour). If it is, print an error. If the number of hours is greater than 60 per, print an error message. Write a Java program that reads (1) basic pay per hour and (2) number of hours worked per week and display the wage/salary to be paid to him/her.

2.4 Write a Java program to implement Vector with following functionality.
Create a vector *student* with syntax: `Vector student = new Vector();`
Using the syntax `student.insertElementAt("Mobile",0); to` add "*Mobile*" at 0th location OR use the syntax `Student.addElement("Mobile");`
Subsequently, add "*Laptop*", "*Watch*", "*Shoes*", "*T-shirt*" and "*Jeans*".
Now, display all the items in student vector using `System.out.println(student);`
Later, the *student* decided to return "Jeans" using `student.remove("Jeans");`
After removing *Jeans*, display the items available with the *student*.
(You may try other functions related to Vectors mentioned in the lecture note)

2.5 Create a class Car with three data members [model, color, brand] and a method [setData, displayCarInfo]. Create following objects.
Car 1: Model-City, Color-White, Brand-Honda
Car 2: Model-Desire, Color-Red, Brand-Maruti
Car 3: Model-XUV500, Color-Maroon, Brand-Mahindra
You may use *setData* method to pass all these values. For example, `car1.setData(String mdl, String clr, String brnd)`. Use the displayCarInfo method to print the details of respective car (E.g. `car1.displayCarInfo();`)

Vidush Somany Institute of Technology and Research, Kadi

CT405-N: Object Oriented Programming Using Java

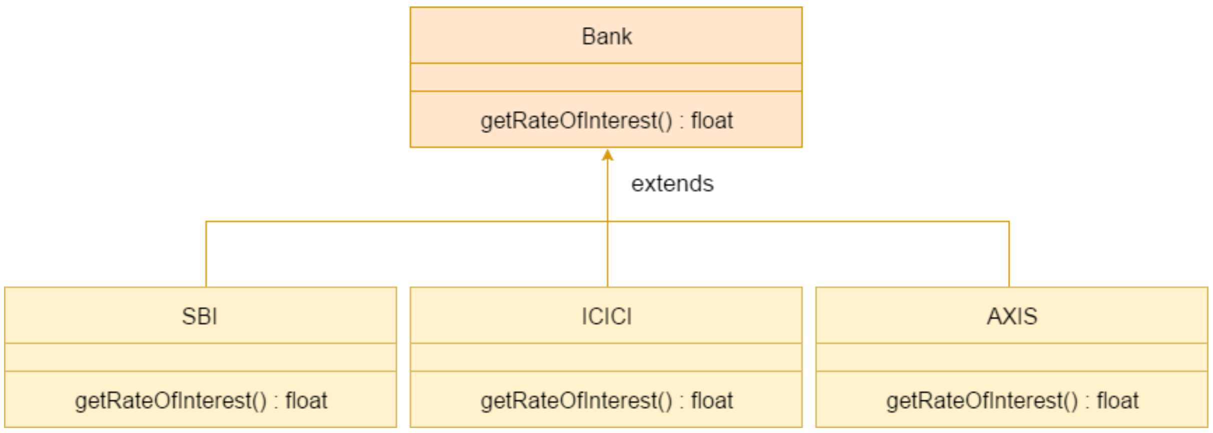
Laboratory Assignment #3

3.1 Define a class ***Doctor*** with Name as data member. Inherit a class ***Surgeon*** from the Doctor class with an extra data member Specialization and Consulting Charge per hour. Inherit another class ***Physician*** with an extra members Speciality and Fees per hour. Define objects **S1** (Specialization= Gynaecologist, Fees per hour= Rs. 2000) and **S2**(Specialization= Paediatrician, Fees per hour= Rs. 1500). Define objects **P1** (Speciality=Heart, Consulting Charge per hour= Rs. 4000) and **P2**(Speciality=Orthopaedic, Consulting Charge per hour = Rs. 2500). Write a main method to create objects as mentioned and required methods to read/display the information. NOTE: Sub-class must call super class constructor.

Solution: https://github.com/hbpatel1976/Java/blob/main/Medical.java

3.2 Create a superclass ***Animal*** that has a method called animalSound(). Subclasses of ***Animal*** could be ***Pig***, ***Cat***, and ***Dog***. And they also have their own implementation of an animal sound (e.g.: the pig oinks, and the cat meows, dog barks etc.) Now create objects of ***Pig*** and ***Dog*** objects and call the animalSound() method on both of them. [Inheritance and method overriding]

3.3 Consider a scenario where ***Bank*** is a class that provides functionality to get the rate of interest. However, the rate of interest varies according to banks. For example, ***SBI***, ***ICICI*** and ***AXIS*** banks could provide 7%, 7.5%, and 7.75%.


Bank
getRateOfInterest() : float
extends
SBI
getRateOfInterest() : float
ICICI
getRateOfInterest() : float
AXIS
getRateOfInterest() : float

Write a Java Program to demonstrate the real scenario of Java Method Overriding where three classes (***SBI***, ***ICICI*** and ***AXIS***) are overriding the method of a parent class (***Bank***)

Vidush Somany Institute of Technology and Research, Kadi

CT405-N: Object Oriented Programming Using Java

Laboratory Assignment #4

4.1 (Final method) Define a class ***Result*** with *dbms*, *os* and *java* as data members. Write a constructor with three arguments that initializes the three data members with these three arguments. Define a `final` method `calculate()` that returns the total of all the three subjects. In the main method, create an object of class ***Result*** and invoke the method `calculate()`.

Solution: https://github.com/hbpatel1976/Java/blob/main/AbstractClassBird.java

4.2 (Abstract class) Define an abstract class **Animal** with three data members viz. `vegetarian` (boolean), `runningSpeed` (int), `animalName` (String). The class **Animal** should have a constructor with three arguments which are assigned to the data members mentioned above. The class **Animal** should also have an abstract method `display()`. Now, inherit two classes **Carnivorous** and **Vegetarian** from **Animal** and write appropriate constructor and the display method. In the main method, create an object Lion of class **Carnivorous** and Cow of class **Vegetarian** with appropriate data members. The Lion and the Cow object should call the `display()` function to print appropriate messages.

Solution: https://github.com/hbpatel1976/Java/blob/main/FinalMethodAssignment1.java

Vidush Somany Institute of Technology and Research, Kadi

CT405-N: Object Oriented Programming Using Java

Laboratory Assignment #5

5.1 GENERATE Output / Error (if any) with REASONING

5.1.1. Generate Output / Error (if any) for the following code. Justify your answer.

```
class Main
{
  public static void main(String[] args)
{
    // create a final variable
    final int AGE = 32;
      AGE = 45;   // try to change the final variable
    System.out.println("Age: " + AGE);
  }
}
```

5.1.2. Generate Output / Error (if any) for the following code.

```
abstract class Shape
{
      abstract void draw();
}
class Rectangle extends Shape
{
      void draw(){System.out.println("drawing rectangle");}
}
class Circle1 extends Shape
{
      void draw(){System.out.println("drawing circle");}
}
class TestAbstraction1
{
      public static void main(String args[])
      {
            Circle1 s=new Circle1();
            s.draw();
      }
}
```

5.1.3. What will be the correct sequence of the Output / Error (if any) for the following code?

```
class Example1
{
public static void main(String args[])
      {
      try
            {
              System.out.println("First statement of try block");
              int num=45/3;
              System.out.println(num);
            }
      catch(ArrayIndexOutOfBoundsException e)
            {System.out.println("ArrayIndexOutOfBoundsException");}
      finally
            {System.out.println("finally block");}
      System.out.println("Out of try-catch-finally block");
      }
}
```

(A) finally block, First statement of try block, 15, Out of try-catch-finally block

(B) finally block, First statement of try block, Out of try-catch-finally block,15
(C) First statement of try block, 15, finally block, Out of try-catch-finally block
(D) First statement of try block, finally block, 15, Out of try-catch-finally block

5.1.4. Generate the output of the following code. Give supporting arguments/ reasons for your Output / Error (if any).

```
class Demo
{
   final int MAX_VALUE=99;
   void myMethod()
      {
           MAX_VALUE=101;
      }
   public static void main(String args[])
      {
      Demo obj=new  Demo();
      obj.myMethod();
      }
}
```

5.1.5. Generate the output of the following code. Give supporting arguments/ reasons for your Output / Error (if any).

```
public class FinalMethodExample
{
   public final void display()
      {System.out.println("We are implementing Final method example");}
   public static void main(String args[])
      {new FinalMethodExample().display();}
   class Sample extends FinalMethodExample
      {
           public void display()
      {System.out.println("hi");}
      }
}
```

5.2 [Parameterized constructor] Create a class Student with two data members viz. id and name. Create a parameterized constructor with two arguments for id and name. Create a method display() to display id and name of the two students. Create two objects s1 and s2 for the same.

5.3 [Constructor Overloading] Create a class Person with three data members name, id and age. Create a constructor with two arguments for id and name and second constructor with three arguments for id , name and age. Create two objects p1 for the first constructor and p2 for the second constructor. Create a display() method to display id, name and age of the person.

5.4 [Abstract class and abstract method]: Create MyClass as the abstract super class with an abstract method calculate(). This method does not have any body within it. Sub1, Sub2 and Sub3 are three sub classes where the abstract method is implemented as per the requirements of the objects. Sub1 calculates square root as x*x, Sub2 calculates square root using Math.sqrt(x) and Sub3 calculates cube of x. Since the same abstract method is implemented differently for different objects , they can perform different tasks.

Write a program where the abstract class MyClass has one abstract method which has got various implementations in sub classes.

Vidush Somany Institute of Technology and Research, Kadi

CT405-N: Object Oriented Programming Using Java

Laboratory Assignment #6

6.1 (Abstract method / class) Declare an abstract class **shape** having `radius`, `length` and `width` as data members and `showArea` as an abstract method. Derive a class **rectangle** from shape with a constructor accepting two integer arguments which are to be copied into `length` and `width` respectively. Override `showArea` method in **rectangle**. Also derive another class **circle** from **shape** with a constructor accepting one integer argument which is to be copied into `radius`. Override `showArea` method in **circle**. In main method, declare objects of classes **rectangle** and **circle**. Also, call `showArea` method of respective classes to display the area.

Solution: https://github.com/hbpatel1976/Java/blob/main/AbstractClassShape.java

6.2 (Multiple Inheritance) Declare an `interface` **shape** having a `final static` variable PI=3.14. Declare a method `computeArea` in shape. Now, inherit a class **rectangle** from **shape** and override the function `computeArea` to print area of a rectangle. Further, inherit a class **circle** from **shape** and override the function `computeArea` to print area of a circle. In main method, create objects of **rectangle** and **circle** and invoke the `computArea` methods of respective class.

Solution: Source: https://github.com/hbpatel1976/Java/blob/main/InterfaceTest.java

6.3 (Multiple Inheritance) Declare an `interface` **motorcycle** with `speed` as a data member and `distanceCovered` as a method. Declare another `interface` **cycle** with `time` as a data member and `runningTime` as a method. Now, derive a class **TwoWheeler** from the `interface` **motorcycle** and **cycle** and override both the methods of the interfaces here in this class to display distance travelled (speed x time) and running time, respectively. Create an object of the class **TwoWheeler** in main method and invoke the methods to display appropriate messages.

Solution: https://github.com/hbpatel1976/Java/blob/main/MultipleInheritance1.java

6.4 (Multiple Inheritance) Create a class **student** with `rollNumber` as data member and two methods viz. `getNumber` and `putNumber` to read and display the `rollNumber`, respectively. Derive a class **Test** from **student** class having two data members, `part1` and `part2`. Add two methods, `getMarks` to read/set marks of `part1` and `part2` and `putMarks` to display them. Create an interface **Sports** with `sportIndex` as data member and a method `putIndex` to display the same. Now, create a class **Results** which is derived (extended) from the class **Test** and (implemented) from the interface **Sports** with `total` as data member. The method `display` in **Results** class should display the total of `part1`, `part2` and `sportIndex`. Create object of **Results** class in main method and test the different methods by invoking them.

Solution: https://github.com/hbpatel1976/Java/blob/main/Hybrid.java