Week 2 assignment

OOP and Exception handling

Assignment #1

Inheritance and polymorphism

Create classes Circle and Cylinder as shown in the UML diagram below:

```
Circle
-radius:double = 1.0
-color:String = "red"
+Circle()
+Circle(radius:double)
+Circle(radius:double,color:String)
+getRadius():double
+setRadius(radius:double):void
+getColor():String
+setColor(color:String):void
+getArea():double
+toString():String
                                           "Circle[radius=r,color=c]"
                     superclass
          extends
                     subclass
                Cylinder
-height:double = 1.0
+Cylinder()
+Cylinder(radius:double)
+Cylinder(radius:double,height:double)
+Cylinder(radius:double,height:double,
   color:String)
+getHeight():double
+setHeight(height:double):void
+getVolume():double
```

In the main() function of a Program class, create an array of Circle references with the initialization shown below:

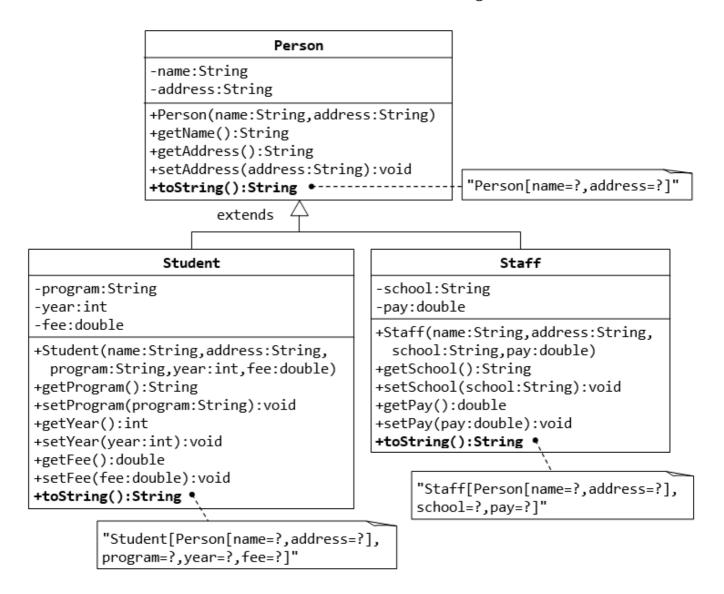
```
Circle[] circles = {
    new Cylinder(12.34),
    new Cylinder(12.34, 10.0),
    new Cylinder(12.34, 10.0, "blue")
};
```

Print the area of the circular region of each cylinder along with the volume of the same.

Assignment #2

Classes, inheritance and polymorphism

Create the classes Person, Student, and Staff as shown in the UML diagram below:



In the main() function of a Program class, create an array of Person references with the initialisation shown below:

```
Person[] people = {
    new Student("Shyam", "Bangalore, Karnataka", "Java fundamentals", 2010, 4500.0
    new Staff("Anand", "Bangalore, Karnataka", "Delhi Public school", 35000.0),
    new Staff("Umesh", "Bangalore, Karnataka", "National Public school", 42000.0),
    new Student("Suresh", "Hassan, Karnataka", "Java fundamentals", 2012, 4750.0),
    new Student("Kiran", "Vasco, Goa", "ReactJS", 2017, 12500.0)
};
```

Print the details of all Person objects (using the toString()).

Assignment #3

Classes, inheritance and polymorphism

Create the classes Shape, Circle, Rectangle, and Square as shown in the UML diagram below:

```
Shape
                -color:String = "red"
                -filled:boolean = true
                +Shape()
                +Shape(color:String, filled:boolean)
                +getColor():String
                +setColor(color:String):void
                +isFilled():boolean
                +setFilled(filled:boolean):void
                +toString():String
             Circle
                                                  Rectangle
-radius:double = 1.0
                                       -width:double = 1.0
                                       -length:double = 1.0
+Circle(radius:double)
                                       +Rectangle()
+Circle(radius:double,
                                       +Rectangle(width:double,
   color:String,filled:boolean)
                                          length:double)
+getRadius():double
                                       +Rectangle(width:double,
+setRadius(radius:double):void
                                          length:double,
+getArea():double
                                          color:String,filled:boolean)
                                       +getWidth():double
+getPerimeter():double
                                       +setWidth(width:double):void
+toString():String
                                       +getLength():double
                                       +setLength(legnth:double):void
                                       +getArea():double
                                       +getPerimeter():double
                                       +toString():String
                                                    Square
                                       +Square()
                                       +Square(side:double)
                                       +Square(side:double,
                                          color:String,filled:boolean)
                                       +getSide():double
                                       +setSide(side:double):void
                                       +setWidth(side:double):void
                                       +setLength(side:double):void
```

The toString function of the above classes should return text as given below:

+Circle()

|Classname|Sample return value from toString()| |---|---| |Shape|A Shape with color of xxx and filled/Not filled | Circle | A Circle with radius=xxx, which is a subclass of yyy (where yyy is the output of the toString() method from the superclass)| |Rectangle|A Rectangle with width=xxx and length=zzz, which is a subclass of yyy (where yyy is the output of the toString() method from the superclass)| |Square | A Square with side=xxx, which is a subclass of yyy (where yyy is the output of the toString() method from the superclass)

+toString():String

In the main() method of a Program class, create an array of 10 Shape references containing a mixture of Circle, Rectangle and Square objects of different dimensions. Using a loop, print the perimeter and area for all of them.

Assignment #4

Summarize user inputs

Write a Java application to accept integers in a loop. After each number is accepted, the user should be asked if he/she wishes to continue. If the user inputs "NO", then the loop should be stopped and following output should be displayed:

```
Number of inputs = X
Number of integer inputs = Y
Number of non-integer inputs = Z
Sum of all integer inputs = XX
The integer inputs = N1, N2, N3, ...
The non-integer inputs = ASD, SDF, DFG, ...
X, Y, Z, etc should be actual values, based on the inputs.
HINT:
Use java.util.Scanner for accepting data from the user.
Scanner s = new Scanner(System.in);
String input = s.nextLine();
int n = s.nextInt();
double d = s.nextDouble();
// ... s
```

Assignment #5

Calendar array

Write a function called "calendar" that takes a String representing year/month in YYYY-MM format and returns a two-dimensional array representing the calendar for the input month and year.

For example, if the input is "2018-03", then the output is:

```
{4, 5, 6, 7, 8, 9 10},

{11, 12, 13, 14, 15, 16, 17},

{18, 19, 20, 21, 22, 23, 24},

{25, 26, 27, 28, 0, 0, 0}

}
```

The method should throw a custom exception **InvalidDateException**, in case if the input does not represent a valid year/month combination, and **InvalidInputException** in case if the input is not in the expected YYYY-MM format.

© 2022 All rights reserved by Learn with Vinod