**Object Oriented Programming Project One**

* Name: Raman Haivaronski Department:\_\_ Software and Computer Engineering\_\_\_
* Student ID: \_\_\_\_201734474\_\_\_\_\_\_\_\_\_\_; Room\_Number:\_\_\_\_\_\_\_\_5322\_\_\_\_\_
* Due Date: \_\_\_\_\_28.11.2017\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
* **Format of File name**: ProjectOne.zip
* The zip file will contain **two** files, namely report file with file format **“ProjectOneReport** and **Source codes**.

1. **General Objective**: To write a Java program that includes the basic concepts of Object Oriented programming.
2. **Specific Objectives**
3. To write code that requests a user to enter input from key board and display output on screen
4. To write a code that declares two interfaces.

3. To write a code that declares an abstract class

4. To write a code that declares an a class which includes another class (Has-A relationship)

5. To write the code of a class that **extends** **one** super abstract class and i**mplements** one interface

(**Is-A relationship**)

6. To write the code of a class a class that **extends** **one** super abstract class and i**mplements two** interfaces

7. To write the code of a method that checks the validity of input using Exception handling concept

**Part One: Problem Description**

Suppose we want to develop a java application that computes the area and the volume of several types of geometrical shapes for a mathematical study. Assume circle, equilateral triangle, square, cone, equilateral triangular pyramid, square pyramid, cylinder, equilateral triangular prism and square prism represent the types of geometrical shapes under the mathematical study. These geometrical shapes can be **two dimensional (2D) or three dimensional (3D)**. Based on the number bases, three dimensional geometrical shapes can be pyramid or prism.

**A Pyramid** is a three dimensional geometrical shape which has one **base area** in **two dimensional space.** Cone, equilateral triangular pyramid and square pyramid belong to this group. A **Prism** is a three dimensional geometrical shape which has two areas at its **base** and **top** in two dimensional space. Cylinder, equilateral triangular prism and square prism belong to this group. The 2D geometrical shape can be represented using two or more points in **two dimensional** Euclidean **coordinate** space. The 3D shapes can be represented by their **two dimensional** base shapes and by their normal height or by their slant height. The description of the following **nine** geometrical shapes is given as follows.

* **Circle:** The center of the circle **C**(x1,y1) and a point on the b**oundary of the circle B**(x2,y2)are required to represent the circle.
* **Equilateral Triangle:** Three vertices of a triangle are required to represent an equilateral triangle: **V1**(x1,x2),**V2**(x2,y2) and ***V3***(X3,y3).
* **Square:** upper left (UL) and lower right vertices are required to represent a square: **UL**(x1,x2) and **LR**(x2,y2)
* **Cone: The** base of the cone (circle) and the **slant** height or **normal** height

of the cone are required to represent a cone.

* **Equilateral Triangular Pyramid**: The base of the pyramid (equilateral triangle) and the **slant** height or **normal** heightof the pyramid are required to represent this pyramid.
* **Square pyramid**: The base of the pyramid (square) and the **slant** height or **normal** height of the pyramid are required to represent this pyramid.
* **Cylinder**: The base of the cylinder (circle) and the **normal** height of the cylinder are required to represent a cylinder.
* **Triangular prism**: The base of the prism (equilateral triangle) and the **normal** height of the prism are required to represent this prism.
* **Square prism**: The base of the prism (square) and the **normal** height of the top of the prism are required to represent this prism.

**Description of Methods of each Geometrical Shapes**

* Two dimensional shapes should have **getDistance**() method to calculate the distance between two points in two dimensional coordinate space. The **getDistance()** method computes the radius of the circle, the length of the three sides of a triangle, the width and the length of the square.
* Two dimensional shapes should have **getArea**() method to calculate the area of 2D shapes
* Three dimensional shapes should have **getArea() method** to calculate the **total** **surface area** of three dimensional shapes.
* Three dimensional shape should have **getVolume**()method to calculate the volume of three dimensional shapes. The **getVolume**() method should **re-uses** the **getArea**() method of **2D** shapes.
* **A concrete class** is needed to represent **2D points**. This class should be used as member field for classes of **2D shapes** using the principle of **composition** (has-a relationship).

**Part Two: How to design the problem?**

We can use three different types’ designs to **re-use** the existing classes of **two** dimensional geometrical shapes to describe t**hree** dimensional geometrical shapes.

1. **Design Option One**: put all the member fields and member methods of both 3D shapes and 2D shapes in one abstract supper class.
   1. **Abstract super class**: Create an abstract super class **called Shapes2D3D**

* *Concrete method*:
* **ChooseShape (**) method. This method askes the user to choose one of the 9 geometrical shapes from the menu list. After user select one shape, the user should also requested to **enter** the necessary member fields of the selected shape type.
* *Abstract methods for 2D shapes*:
* getDistance()
* getArea()
* *Abstract methods for 3D shapes:*
* Normal height member field
* Slant height member field,
* getNHeight()
* getSHeight()
* getVolume()
  1. **Abstract Subclasses of Shapes2D3D**: Circle, equilateralTriangle and Square
* **Concrete method**:
* Implement getDistance()
* Implement getArea().
  1. **Concrete Subclasses of clases in (1.2)**:
* Cone,EquilateralTriangularPyramid,SquarePyramid
* Cylinder, EquilateralTriangularPrism, SquarePrism
* **Concrete method**:
* @Implement **getNHeight**()
* @Implement **getSHeight**()
* @ Override **getArea(**) to compute Surface area 3D shapes. Since surface area of 3D shapes is the **sum** of its **base area** in 2D space and its **lateral surface**, this method should **re-use** the **getArea**() method of 2D shapes.
* @Implement **getVoume()**.Since the volume of 3D shape is the product of its **base area** in 2D and its **height**, this method should **re-use** the **getArea**() method of 2D shapes.

1. **Design Option Two**: add the member fields and member methods of 3D shapes only in the classes where they belong.

* 1. **Abstract super class**: Create an abstract class called **Shape2D**
* ***Concrete method*:**
* **ChooseShape** () method. This method askes the user to choose one of the 9 geometrical shapes from the menu list. After user select one shape, the user should also requested to **enter** the necessary member fields of the selected shape type.
* ***Abstract methods for 2D shapes***:
* getDistance()
* getArea() .
  1. **Concrete Subclasses of Shapes2D**: Circle, equilateralTriangle and square.
* **Concrete method**:
* @Implement getDistance()
* @Implement getArea().
  1. **Concrete Subclasses of class in(2)**:
* Cone, EquilateralTriangularPyramidand SquarePyramid
* Cylinder, EquilateralTriangularPrism and SquarePrism
* **Concrete method**:
* @ Override getArea**(**):to compute Surface area 3D shapes. This method should **re-use** getArea() of 2D shapes( **like 1.3**)
* **Add**  getNHeigh**t**(): to find height of 3D shapes
* Add getSHeight(): to find height of 3D shapes
* **Add** getVoume**(**): to find volume of 3D shapes. **This** method should also re-use the **getArea**() method of 2D shapes(**like 1.3**).

1. **Design Option Three**: Using two different inheritance hierarchy. We add the member fields and member methods of 2D shapes in one inheritance tree and we add the member fields and member methods of 3D shapes in another inheritance tree.
   1. **Interface as supper class:** create **Shapes3DPyramid as super Interface**

* ***Abstract methods for 3D prism shapes***
* **Choose3DShape**() method. This method askes the user to choose one of the 6 geometrical 3D shapes and to input the necessary member fields of the chosen shape type.
* **getNHeight**(): to find height of 3D shapes
* **getArea**(): to find surface area of 3D shapes( like 1.3)
* **getVolume**() to find volume of 3D shapes(like 1.3)
  1. **Interface as supper class:** create **Shapes3DPrism as super Interface**
* **Abstract methods for 3D pyramid shapes**
* getSHeight**():** to find height of 3D shapes
  1. **Abstract class:** create **Shapes2D** as abstract superclass
* **Concrete method for this class** :
* Choose2DShape(). This method askes the user to choose one of the 3 geometrical 2D shapes and to input the necessary member fields after the user selected the type of 2D shape..
* **Abstract methods for 2D shapes**
* getDistance()
* getArea().
  1. **Concrete Subclasses Shapes2D:** Circle, EquilateralTriangle and Square.
* **Concrete method**:
* Implement getDistance() method
* Implement getArea()method .
  1. **Concrete Subclasses which extends classes in (3.4) and also implements Shapes3DPrism Interface.**
* Cylinder, EquilateralTriangularPrism and SqaurePrism
* **Concrete method**:
* @ Override **getArea(**):to compute Surface area 3D shapes(like 1.3)
* implement **getNHeight**():to find height of 3D shapes
* impement **getVoume()**: to find volume of 3D shapes(like 1.3)

* 1. **Concrete Subclasses which extends classes in (3.4) and also implements Shapes3DPrismInterface and Shapes3DPyramid Interface.**
* Cone,EquilateralTriangularPyramid,SquarePyramid
* **Concrete method**:
* @ Override **getArea():**to compute Surface area 3D shapes(like 1.3)
* implement **getNHeight**():to find height of 3D shapes
* implement **getSHeight**():to find height of 3D shapes
* implement **getVoume()**: to find volume of 3D shapes(like 1.3)

**Part 3**: **Additional Requirements.**

In addition to the above functional requirements, the application should fulfilled the following requirements.

* **Add** the proper **member fields** for each class
* Each member field should have **privat**e access modifier
* For each member field, each class should has **setter and getter** methods.
* Use **PI** value from Math class(java.lang.Math)
* **You have include exception handling codes that checks the validity of user input and some operations as follows.**
* **For** **circle**, if points c(x1,y1) and b(x2,y2) are the same, it is not circle because it a **single point.** The program should display the error message and request the user to enter valid points.
* **For triangle**: if the tree vertices V1(),V2() and V3() are collinear (lie on the same line), it is not a triangle because these points represent a straight line. The program should display the error message and request the user to enter valid vertices.
* **For** **square**, if one of the following is true, it is not a square.
* If the upper left vertex (x1,y1) and lower right vertex(x2,y2) are the same
* if x2 is less than x1,
* If y2 is less than y1.
* If the length and width are different. The program should display the error message and request the user to enter valid inputs.
* For **3D shapes**, if the **height** value is zero, the program should display the error message and request the user to enter valid inputs.
* For 3D shapes that are pyramid, if Slant height is less than normal height, the program should display the error message and request the user to enter valid inputs.
* **Common requirement for all geometrical shapes:** your code should handle the **exception** when the user enters a **character** or a **string** from key board. The program should display the error message and request the user to enter valid input.
* In each class, override the **toString**() method of the “Object” class appropriately as shown below in part 5.

**Part 4: input and output requirements of the test class**

* **For design option three:**
* Declare **ChooseShape**() method in the **test class** to display a menu for user to choose 2D shape or 3D shape.
* If the user selects 2D shape, **ChooseShape()** should invoke **choose2DShape**() method. Then, the c**hoose2DShape** () method should display a menu for the user to choose one of the **tree** two dimensional geometrical shapes. Then the program should request the user to enter the required member fields of the selected shape
* If the user selects 3D shape, **ChooseShap**e() should invoke **choose3DShape**() method. Then, the choose3DShape () method should display a menu for the user to choose one of the six three dimensional geometrical shapes. Then the program should request the user to enter the required member fields of the selected shape
* Hence, **main** () method should invoke **ChooseShape**() method
* The main () method should invoke **System.out.println**() method by using **toString()** and **getter**() methods as arguments of System.out.println() method .
* Hence, each class should override the **toString** () method of the “**Object class**” appropriately.
* **The following is the output format when a specific shape is selected.**

1. **Type of shape**: circle

* Center: (x1,y1)
* Point on boundary: (x2,y2)
* radius=value
* **CircularBaseArea** = value.

1. **Type of shape**: Equilateral Triangle

* Three vertices: (x1,y1), (x2,y2),and (x3,y3)
* Length of three sides = value.
* **TrinagularBaseArea** = s1,s2,s3.

1. **Type of shape**: Square

* Upper left vertex:(x1,y1)
* Lower right vertex: (x2,y2)
* length =value
* width = value
* **SquareBaseArea** = value**.**

1. **Type of shape**: Cone

* Center of its circular base : (x1,y1)
* Point on boundary of its circular base : (x2,y2)
* Radius of its circular base = value
* **CircularBaseArea = value**,
* **Slant Height=value and SurfaceArea = value**
* **Normal Height =cvalue and Volume = value.**

1. **Type of shape**: Equilateral triangular pyramid

* Three vertices of its triangular base : (x1,y1), (x2,y2),and (x3,y3)
* Length of three sides its triangular base = s1,s2,s3.
* **TriangularBaseArea = Value**
* **Slant Height = value and**  **SurfaceArea = value**
* **Normal height=value and Volume = Value**

1. **Type of shape**: Square pyramid

* Upper left vertex of its rectangular base :(x1,y1)
* Lower right vertex of its rectangular base :(x2,y2)
* Length of base =value
* Width of base=vaue
* **SqaureBaseArea=Value**
* **normal height=value and SurfaecArea = value**
* **Normal height=value and volume=Value**

1. **Type of shape**: Cylinder

* Center of its circular base : (x1,y1)
* Point on boundary of its circular base : (x2,y2)
* Radius of its circular base = value
* **CircularBaseArea = value**,
* **Normal height=value and SurfaecArea = value**
* **Normal height=value and volume=Value**

1. **Type of shape**: Equilateral triangular prism

* Three vertices of its triangular base : (x1,y1), (x2,y2),and (x3,y3)
* Length of three sides its triangular base = s1,s2,s3.
* **TrinagularBaseArea = value.**
* **normal height=value and SurfaecArea = value**
* **Normal height=value and volume=Value**

1. Type of shape: Square prism

* Upper left vertex of its square base :(x1,y1)
* Lower right vertex of its Square base : (x2,y2)
* Length of base=value
* width of base =vaue
* **SquareBaseArea=Value**
* **Slant height=value and SurfaecArea = value**
* **Normal height = value and volume = Value**

**Part 5: Additional Information: area of 2D shapes, Surface area and volume of 3D shapes is given as follows.**

1. **Surface area of 2D shapes**

* **Surface area of circle(AC): AC** = PI\*r2,r is radius of circle
* **Surface area of equilateral triangle**(**AT):AT** = **,**

**Where ;** s1 ,s2 and s3 are length of three sides of the triangle  **and p is** perimeter of the triangle.

* **surface area of square(AS): AS** = L2=w2 (length= width);

1. **Surface area of 3D shapes** 
   1. S**urface area of 3D pyramids: Base area +1/2\*P\*h.**

* **Surface area of cone(Acon):**
* **Acon** = **AC +PI\*r\*h**
* **AC** is surface area of the circle as the base of the cone
* **p** = perimeter of the circle as the base of pyramid
* **r** is the radius of the circle as the base
* **h**= **slant height** of the apex of the cone

Area = Pi\*r^2 + Pi\*r\*L

* **Surface area of Equilateral triangular pyramid (ATPY).**
* **ATPY**= **AT**+**1/2\*3\*s\*h**, where **p = s1+S2+S3 =3\*s**
* **AT** is surface area of the equilateral triangle as the base
* **P** = perimeter of the equilateral triangle as the base
* **S** = the side of the equilateral triangle as a base.
* **h** = the **slight height** of the apex of the triangular pyramid
* **Surface area of square pyramid, ASPY: base area+ later surface**
* ASPY= **AS**+1**/2\*P\*h**, where **p = s1+s2+s3+s4=4\*s.**
* **AS** is surface area of the ***square as the base*** of the pyramid
* **P** = perimeter of the square as a base of square pyramid
* **S** = the side of the square as a base.
* **h** = the **slight height** of the apex of the square pyramid
  1. **Surface area of 3D prisms:** 2\***Base area +P\*h.**
* **Surface area of Cylinder (ACY)**
* **Acy** = **2\*AC +P\*h**, where **p =2\*PI\*r**
* **AC** is surface area of the circle as the base
* p = perimeter of the circle as the base
* r is the radius of the circle as the base
* h= normal **height** of the cylinder( **not slant height**)
* **Surface area of triangular prism(ATPR)**
* ATPR= 2\***AT**+**P\*h**, where **p = s1+S2+S3 =3\*s**
* **AT** is surface area of the equilateral triangle as the base
* **P** = perimeter of the equilateral triangle as the base
* **S** = the side of the equilateral triangle as a base.
* **h** = the **normal height** of triangular prism (**not slant height**)
* **Surface area of sq**uare prism (ASPR)
* **ASPR**= **2\*AS+P\*h**, where **p = s1+s2+s3+s4=4\*s.**
* **AS** is surface area of the ***square as the base*** of square prism.
* **P** = perimeter of the square as a base of square pyramid
* **S** = the side of the square as a base.
* **h** = Norma height of the square prism (**not slant height**).

1. **Volume of 3D shapes :** 
   1. **Volume of 3D pyramids: 1/3\*Base Area\*height**

* **Volume of cone(Vcon)**:
* **Vcon = 1/3\*AC\*h**
* **AC** is surface area of the circle as the base of the cone
* h= **normal height** of the apex of the cone(**not slant height**)
* **Volume of equilateral triangular pyramid(VTPY)**
* **VTPY= 1/3\*AT\*h**
* **AT** is surface area of the equilateral triangle as the base
* **h** = the normal height of the pyramid(**not slant** **height)**
* **Surface area of square pyramid(VSPY)**
* **VSPY= 1/3\*AS\*h.**
* **AS** is surface area of the ***square as the base*** of the pyramid
* **h** = the normal height of the apex of the square pyramid

**3.2. Volume of 3D prisms**

* **Volume of Cylinder**: **Base area\*height**
* **Vcy = AC\*h**
* **AC** is surface area of the circle as the base of the cone
* h= **normal height** of the apex of the cone(**not height**)
* **Volume of equilateral triangular prism**:
* **VTPR= AT\*h**
* **AT** is surface area of the equilateral triangle as the base
* **h** = normal height of the triangular prism
* **Surface area of square prism:**
* **VSPR= AS\*h.**
* **AS** is surface area of the ***square as the base*** of the pyramid
* **h** = **normal** height of the square prism

**Question 1: write the advantage and disadvantage of design option 1**

**Question 2: write the advantage and disadvantage of design option 2**

**Question 3: write the advantage and disadvantage of design option 2**

**Question 4: write the code for design option 3.**

Answer 1:

The disadvantage of first design is lacking of flexibility. There is no interfaces and all methods are piled in one class that is not good when you have a really big object. The best way to do it is to split the program for each figure in its own class with its own private fields to avoid mistakes and uncertainties.

Answer 2:

The disadvantage of the second design is that there is no division to 2d and 3d subparts of the program and again no interfaces. Overall 1 and 2 ways are poorly designed and are difficult to understand by other programmers in the team.