CS 200
Homework Assignment 2
Given on September 23, 2022
Due in Canvas by Sunday, October 9, 2022 @ 11:59 PM
70 points available

This assignment involves the following Java topics:

1. User-defined methods
2. Branches
3. Objects and Classes
4. Input and Output
5. Inheritance
6. Exceptions

For this assignment, you will:

Create an abstract Java class called Shape. Shape has a single field member called weight. It has two constructors.

- Constructor #1 - has no parameters and initializes the Shape's weight to 0.0
- Constructor #2 - has one double parameter that initializes the Shape’s weight

It also has two methods

- setWeight() which accepts an double for the Shape object’s weight and returns nothing
- getWeight() which accepts no arguments and returns the Shape object’s weight value

Measurement of weight is in pounds.

Next, create a Java class named Box. It will be a derived class from Shape with the following characteristics:

The class has three private double instance variables to store the length, width, and height of the Box. Measurement is in feet.

This Box class has two constructors:

- Constructor #1 - has no parameters and initializes the Box's instance variables to 0.0
- Constructor #2 - has four double parameters that initializes the Box object's length, width, height, and weight

This Box class has the following 6 accessor methods

- setLength() which accepts a double for the Box object’s length and returns nothing
- getLength() which accepts no arguments and returns the Box object’s length value as a double
- setWidth() which accepts a double for the Box object’s width and returns nothing
- getWidth() which accepts no arguments and returns the Box object’s width value as a double
- setHeight() which accepts a double for the Box object’s height and returns nothing
- getHeight() which accepts no arguments and returns the Box object’s height value as a double

and the following 5 additional methods:

- isACube() which takes no arguments and returns a Boolean (true if a cube, meaning length, width and height are the same; false otherwise). It needs to call the appropriate accessor methods to get the input values.

- hasASquare() which takes no arguments and returns a Boolean (true if it contains a square, meaning *at least* two of the three: length, width and height contains the same value; false otherwise). It needs to call the appropriate accessor methods to get the input values.
- calculateVolume() which takes no arguments and returns the volume (formula: length*width*height) of the Box object as a double. It will need to call the appropriate accessor methods to make the calculation.
- calculateDensity() which takes no arguments and returns the density of the Box (formula: weight/volume) as a double. Check for divide by 0. It will need to call calculateVolume() to calculate the volume. It will also need to call an accessor method to get the weight. Create an exception to test for a divide by 0 condition, and print a message to the user if this is the case. If it is not a divide by 0, perform the calculation and return the calculated density.
- isHeavy() which takes no arguments and returns a Boolean true value if the density is greater than or equal to 10.00 lbs./cu ft.; false otherwise. It needs to call calculateDensity() to obtain the density.
- isLightweight() which takes no arguments and returns a Boolean true value if the density is less than or equal to 0.10 lbs./cu ft.; false otherwise. It also needs to call calculateDensity() to obtain the density.

Although Box does not have its own getWeight() and setWeight() methods, it should make use of the functionality that comes from the methods in its superclass, Shape. In other words, model Box as a derived class, not as an independent class. Weight must be declared as a protected or private variable (attribute) within Shape.

The Box class contains a main() method that prompts the user for double values of length, width, height, and weight and creates an instance of the Box object. It computes the Box's volume and density and then outputs the results to the user's screen (for density, output to hundredths, e.g. 15.33). It also prints out a determination of whether one of the sides contains a square and if it is a cube. It should also output if it is a heavy box.

Main() should not perform any calculations – it should only contain comments, method calls, print statements, prompts to the user (and reading in responses), and conditional statements based on the values returned from the methods.

All outputs should use the printf() method and output to hundredths (e.g. 15.33)

An example:

```
Enter the length in feet: 6.0

Enter the width in feet: 3.0

Enter the height in feet: 2.0

Enter the weight in pounds: 484.0

The volume of the box is 36.00 cubic ft.

The density of the box is 13.44 lbs./cubic ft.

No sides of the box are square.

The box is not a cube.

The box is heavy.

The box is not lightweight.
```

Another example:

```
Enter the length in feet: 4.4

Enter the width in feet: 4.4

Enter the height in feet: 4.4

Enter the box weight: 188.6

The volume of the box is 85.18 cubic ft.

The density of the box is 2.21 lbs./cubic ft.

At least one side of the box is square.

The box is a cube.

The box is not heavy.

The box is not lightweight.
```

A third example:

```
Enter the box length: 0.0

Enter the box width: 5.0

Enter the box height: 5.0

Enter the box weight: 20.03

Division by zero attempted.

The volume of the box is 0.00 cubic feet.

The density of the box is 0.00 lbs./cubic ft.

At least one side of the box is square.

The box is not a cube.

The box is not heavy.

The box is lightweight.
```

What to submit:

- Two java files (or a BlueJ project containing these two class files) – one for the Box class, one for the Shape class. The files should be able to run in BlueJ without any modifications on my part, so keep this in mind if you use a different IDE.
- A short video (or link to the video on YouTube or other content-sharing service) showing your code running. In your video include the examples given above in your demo. You can use a free tool like screencast-o-matic (https://screencast-o-matic.com) to capture the screen video.

Grading:

You should make appropriate comments in your code – they do not need to be excessive but should include a clear, concise understanding of the code you have written. Also, include a header similar to what was done in Assignment 1. There will be a deduction of up to **5 points each** if these are not included.

Use appropriate variable names that clearly illustrate their purpose. For counting variables, you can use single letters, but all others should be descriptive. A deduction of up to **5 points** will be made for using inappropriate variable names.

Coding should follow the instructions above. While the rubric for this assignment does not specifically allocate points for each method, the breakdown is *roughly* as follows:

- Creating the shape class correctly: **10 points**
- Creating the box object correctly: **20 points**
- Use of inheritance between objects and of a derived class (no duplicated methods or variables): **10 points**
- Implementing the appropriate methods, making use of method calls and return values as appropriate: **15 points**
- Input and output given as described in the examples in the assignment: **5 points**
- Video clearly illustrates the tasks and examples given above: **5 points**

Pay attention to the deadline for this assignment. Penalties do apply for late submissions.