CS 200 Homework Assignment 3
Given: Monday, 17 October 2022
Due: Sunday, 6 November 2022 @ 11:59 PM
100 points (+10 bonus points) available

This assignment involves the following Java topics:

1. User-defined methods
2. Objects and Classes
3. Inheritance
4. Exceptions

For this assignment you will make some modifications to your Assignment 2, so it may be useful to use that as a starting point. We will look at several objects and how much waste is involved with providing a container for that shape. See https://web.nmsu.edu/~pbaggett/Lessons/ballbox1/ballbox.html for more background information.

Shape will become an *interface* instead of an *abstract class*. (I am being intentionally vague about what is necessary in this step, but the objects will implement Shape instead of extending it while retaining the same functionality.)

Create several objects that implement Shape. When constructed, they will output their values.

- **Box** constructor takes four parameters length, width, and height plus weight. In the constructor, output the values for each dimension.
- **Sphere** constructor takes a single parameter (diameter, a double, which is the maximum value of {length, width, and height}) plus weight. In the constructor, output the values for each dimension.
- **Cylinder** constructor takes two parameters, height, which is the maximum value of {length, width, and height}) and diameter, which is the second largest value of {length, width, and height}) plus weight. In the constructor, output the values for each dimension.
- **Cone** constructor takes two parameters, height, which is the maximum value of {length, width, and height}) and diameter, which is the second largest value of {length, width, and height}) plus weight. In the constructor, output the values for each dimension.

Implement an appropriate constructor for each class. Each of the 4 implementations of shapes (Box, Sphere, Cylinder, and Cone) should have a minimum of one constructor.

Implement the appropriate accessor methods (getters and setters) for each variable for each object type.

Each shape should have a weight, in pounds. Weight is of type double.

The methods from HW2 to evaluate if an item has a square or is a cube or is lightweight or is heavy are not needed for this assignment.

The following methods will need to be created:

- **calculateVolume()** which takes up to three arguments and/or arrays and returns the volume of an object (Box, Cone, Cylinder, or Sphere)
    - For **Box**, it will use the height, length, and width and calculate the volume
    - For **Cone**, it will use the *maximum* of: height, length, and width as the height, and the *second greatest value* of the three as a diameter. It will then calculate the volume. V=π * radius^2 * height/3, where radius = diameter/2. Use Math.pi for π.
    - For **Cylinder**, it will use the *maximum* of: height, length, and width as the height, and the *second greatest value* of the three as a diameter. It will then calculate the volume. V=π * radius^2 * height, where radius = diameter/2. Use Math.pi for π.

    - For **Sphere**, it will use the *maximum* measurement of height, length, and width. It will consider this maximum value as the diameter. It will then calculate the volume of the sphere. V = 4/3 * π * radius^3 where radius = diameter/2. Use Math.pi for π.
- **calculateDensity()** which takes up to four arguments and/or arrays and returns the density of an object (Box, Cone, Cylinder, or Sphere) as a double.
    - Formula: weight/volume
    - It will need to call calculateVolume() to calculate the volume for each object
    - It will calculate the densities for each shape and output the volume and density of each shape (or pass back to main() to print from there) as a comparison.
- **calculateBestFit()** which will indicate the minimum volume dimensions for a container to hold the object.
    - for a **Box**, it is length * width * height
    - for a **Cone**, it is diameter * diameter * height
    - for a **Cylinder**, it is diameter * diameter * height
    - for a **Sphere**, it is diameter * diameter * diameter
    - It will either print the values to the screen or pass them back to main() to print from there.
- **calculateWaste(),** which indicates the amount of wasted space for a container for each object and output it as a percentage.
    - The formula is (containerVolume – volume) / containerVolume.
    - This will be output as a double rounded to two decimal places (e.g., 16.76%) and printed as a percent.
    - It will either print the values to the screen or pass them back to main() to print from there.
- Create a **separate class** that contains a **main( )** method which prompts the user for height, width, length, and weight. It will raise an exception if the wrong information is entered, so test for appropriate boundary conditions and data types:
    - If a 0 is input for any dimension, including the weight, an exception should be raised and the user should be re-prompted for the information (e.g., the program should not end). Note that you may choose to implement this differently (or in a different method) than was done in HW 2 as long as an exception using a try.. catch is used.
    - It will create a Box, Cone, Cylinder, and Sphere.
- It will only allow a numeric value to be input for height, length, width. There are several ways to do this – here is one: https://stackoverflow.com/questions/18804872/only-let-users-input-positive-integers-no-decimals-or-strings
- Weight should take a double as input. If an exception is raised, the user should be re-prompted for the information (e.g., the program should not end).
- It will create an instance of the 3 objects and output their values.
- It will calculate of volume and density using calculateDensity(), calculateBestFit() and calculateWaste() to calculate the appropriate values.
- For all values correctly input by the user, preserve their precision (see example 2 below). For all calculated output values, output to two decimal places, e.g. 12.30).
- 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. It can determine maximum values of user inputs.

Examples:

```
Enter the length: 0

Invalid entry.  Only positive values > 0 allowed.  Please re-enter.

Enter the length: 6.3

Enter the width: 10.0

Enter the height: C
```

```
Invalid entry.  Only positive values > 0 allowed.  Please re-enter.
Enter the height: 2.3
Enter the weight: 1234.56

A Box of length 6.3, width 10.0, height 2.3, and weight 1234.56 lb.
created.
A Cone of height 10.0, diameter 6.3, and weight 1234.56 lb. created.
A Cylinder of height 10.0, diameter 6.3, and weight 1234.56 lb.
created.
A Sphere of diameter 10.0 and weight 1234.56 lb. created.

Box volume: 144.90 cu ft        Box density: 8.52 lbs./cu ft
Cone volume: 103.91 cu ft       Cone density: 11.88 lbs./cu ft
Cylinder volume: 311.72 cu ft   Cylinder density: 3.96 lbs./cu ft
Sphere volume: 523.60 cu ft     Sphere density: 2.36 lbs./cu ft

An object container would need to be:
144.90 cu ft for a Box (0.00% waste)
396.90 cu ft for a Cone (73.82% waste)
396.90 cu ft for a Cylinder (21.46% waste)
1000.00 cu ft for a Sphere (47.64% waste)
```

Another example:

```
Enter the length: 2.576
Enter the width: 0
Invalid entry.  Only positive values > 0 allowed.  Please re-enter.
Enter the width: -4
Invalid entry.  Only positive values > 0 allowed.  Please re-enter.
Enter the width: 4.06
Enter the height: 3.781
Enter the weight: 0
Invalid entry.  Only positive values > 0 allowed.  Please re-enter.
Enter the weight: 313.13
```

```
A Box of length 2.576, width 4.06, height 3.781, and weight 313.13 lb.
created.

A Cone of height 4.06, diameter 3.781, and weight 313.13 lb. created.

A Cylinder of height 4.06, diameter 3.781, and weight 313.13 lb.
created.

A Sphere of diameter 4.06 and weight 313.13 lb. created.


Box volume: 39.54 cu ft        Box density: 7.92 lbs./cu ft

Cone volume: 15.20 cu ft       Cone density: 20.61 lbs./cu ft

Cylinder volume: 45.59 cu ft   Cylinder density: 6.87 lbs./cu ft

Sphere volume: 35.04 cu ft     Sphere density: 8.94 lbs./cu ft


An object container would need to be:

39.54 cu ft for a Box (0.00% waste)

58.04 cu ft for a Cone (73.82% waste)

58.04 cu ft for a Cylinder (21.46% waste)

66.92 cu ft for a Sphere (47.64% waste)
```

What to submit:

- A single BlueJ project with five classes (one for Shape, Box, Cone, Cylinder, Sphere, and your main) – The BlueJ project will be named <*your last name*>HW3. A deduction will be made if you submit only a pdf file or other types of files.
- A narrated short video (or link to the video on YouTube or other content sharing service) showing your code and illustrating your code running. In your video include the examples given above in your demo. A deduction will be made for not including a video.
- The video and the BlueJ project can be zipped together in a single file or submitted as separate files.

Grading:

As with HW2, 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 HW 2 for each class file. There will be a deduction of up to 5 points each if these are not included.

Submit your code and videos in the format provided in "what to submit" – a deduction of up to 10 points will be made if these are not followed.

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, box, cone, and sphere classes correctly and outputting the values to the user: 30 points

- Creating the main class correctly (e.g. without any calculations except determining maximum values of user inputs): 10 points
- Other methods calculate values correctly and output data in the correct format: 30 points
- Input and output given as described in the examples in the assignment: 20 points
- Video clearly illustrates the code and examples given above: 10 points

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

A **bonus** of **5 points** will be given if your program prints everything from the class constructors, from main() or from a separate method you create designed for printing in the main class. This will require more passing of values back and forth through your main() method (e.g., you would not print anything but exceptions from calculateVolume(), calculateDensity(), calculateBestFit() or calculateWaste(), but only from main() or a separate method for printing that you create that is only called from main()). If you attempt the extra credit, please indicate this in your comments and, if possible, in the submission comments in the Canvas dropbox.

Another **bonus** of **5 points** will be given if your program the information out for the box volume and density that follows a hard left column for each that is variable based on the inputs (see example below).

The following will earn extra credit (if pointed out in your narrated video):

```
Box volume: 39.54 cu ft        Box density: 7.92 lbs./cu ft
Cone volume: 15.20 cu ft       Cone density: 20.61 lbs./cu ft
Cylinder volume: 45.59 cu ft   Cylinder density: 6.87 lbs./cu ft
Sphere volume: 35.04 cu ft     Sphere density: 8.94 lbs./cu ft
```

The following will earn full points but no extra credit:

```
Box volume: 39.54 cu ft    Box density: 7.92 lbs./cu ft
Cone volume: 15.20 cu ft     Cone density: 20.61 lbs./cu ft
Cylinder volume: 45.59 cu ft   Cylinder density: 6.87 lbs./cu ft
Sphere volume: 35.04 cu ft   Sphere density: 8.94 lbs./cu ft
```