

Problem 1 (Adapted from Programming Exercise 13.11 - The Octagon class, 10 points): Download the file `GeometricObject.java` from Blackboard. In a file named `Octagon.java` write an `Octagon` class that extends `GeometricObject` and implements the `Comparable` and `Cloneable` interfaces. Assume that all eight sides of an octagon are of equal length. Write methods for computing the area and the perimeter. The perimeter is obviously 8 times the side length. The area can be computed using the following formula:

$$\text{area} = (2 + 4 / \text{Math.sqrt}(2)) * \text{side} * \text{side}$$

In addition to a side data field, every `Octagon` should have a `wasCloned` data field of type `boolean`. By default the `wasCloned` value is `false`. However, if an `Octagon` was created by your clone method, then the `wasCloned` value should be `true`. Write a `toString` method in the `Octagon` class that returns a string containing the side length, the perimeter, the area, and the value of `wasCloned`. Write a test program (`TestOctagon.java`) that creates an `Octagon` object with side value 5. Display the object, then clone it and display the clone. Finally, display the value that you get when you compare the two objects using the `compareTo` method:

```
oct1: Octagon with side = 5.0, perimeter = 40.0, area = 120.71067811865476, wasCloned
= false
oct2: Octagon with side = 5.0, perimeter = 40.0, area = 120.71067811865476, wasCloned
= true
oct1.compareTo(oct2): 0
```

Problem 2 – In a file `Laptop.java`, create a `Laptop` class and make the class implement the `Comparable` interface. The member variables should include the following laptop configuration details – `cpu`, `ram`, `hdd`, graphics card (boolean 1=yes/2=no), screen size, weight, battery life (hours), and price. The constructor for the class should allow the initialization of all the member variables. The constructor for a `Laptop` instance should also calculate a ‘score’ variable out of 10, which will be calculated as follows:

$$\text{laptopScore} = (2 * \text{cpu}/\text{cpuMax}) + (2 * \text{ram}/\text{ramMax}) + (1 * \text{hdd}/\text{hddMax}) + (\text{graphics}) + (1 * \text{screen}/\text{screenMax}) + (1 * \text{weight}/\text{weightMax}) + (1 * \text{battery}/\text{batteryMax}) + (1 * \text{price}/\text{priceMax})$$

Here, use the following max values: `cpuMax = 3.0`, `ramMax = 32`, `hddMax = 2048`, `screenMax = 17.0`, `weightMax = 6`, `batteryMax = 9`, `priceMax = 2000`

Create a method `randomLaptopCreator` to create a list of 5 laptops with randomly generated configurations for each of the specification variables. Specify the range for the random values for each specification item based on a (*reasonably assumed*) min value and the max value. You will also create a `toString` method, which will print out the configuration of a laptop (similar to the octagon problem above) including the laptop score. The overridden `compareTo` method in the `Laptop` class should compare the configurations of the laptops based on the laptop score. Next, you will use the `Arrays.sort` method to sort the list of randomly generated laptops, and print them out (using the `toString` method) in ascending order.