A7-Programming 1

kia Babashahi Ashtiani

Dec 2020

You need to have proper setters, getters, constructors, and toString methods for all of your classes

1 About this assignment

This assignment is an optional assignment designed to help those students who want to score higher in this course. It will be marked as zero or 100. So either all instructions have been respected (You have a proper documentation for all methods, have respected OOP design ,...) and your code works properly and generates the right output, in that case you get 100% or else it will be marked to zero. Late submissions are not accepted under **any** circumstances.

2 Class Resource

A resource class has three attributes:

- name: String.
- int: Double.
- divisible: boolean
- int resCount (it keeps track of the current resource number you can use an extra static variable if you like to count).

Each field has its corresponding setters and getters. For this class you need to have an equals method and you also have to have a copy constructor.

A resource is divisible if it would make sense dividing it. A cake for example is divisible however a wedding ring is not.

3 Class ResourceBundle

This class has the following fields:

- 3 resources of type Resource.

- double bundlePrice.

It also consists of the following methods:

- calcBundlePrice() which is a void method. Description can be found below.
- isPrime(int n): Which takes an integer n and checks if it is prime or not. (In case you deliver this code, you have to be able to explain prime numbers and your algorithm. Otherwise the assignment will be marked to zero).
- double getBundlePrice();

This class will generate 3 resources. For each resource the price will be a random number between 10 to 150. Also the name of each resource has to start with the resource name and it should continue with the resCount. So for resource 1 you will have "res01", for resource 2 "res02" ,... .

This class has also a Method called calcBundlePrice() which is a void method that sets the price field of the bundles in the following fashion.

3.1 calcBundlePrice()

Assume the price or resource r is shown by r_p. The method has to do the following computation for all of the resources to update the bundlePrice variable. **You are allowed to use arrays if you know how to use them**. Also, in order to check if a number is prime or not you must call the isPrime() method.

If the resource is indivisible, and its price is a prime number, the price of the bundle will be incremented by the price of the item raised to the power of three. So by r_p^3.

If the resource is divisible and its price is not prime, the price of the bundle will be incremented by $2 * \sqrt{p_r}$.

If the resource is divisible and its price is prime, the bundle price will be incremented by the price of the resource it self i.e., p_r.

If the resource is indivisible and its price is not a prime number, you simply have increment the price of the bundle by the absolute value of the $cos(p_r)$ or $|cos(p_r)|$.

You have to initialize these objects yourselves and no user input should be demanded. In order to do so you first need to call a method that creates or sets your resources then you will use those resources to set the price of the bundle. This can be done either in the constructor of the class or, via a separate method (preferably in the constructor).

4 Bargain

In this class you will create three different resource bundles and then find the bundle with the cheapest price. Once you found this bundle you will find its distance in price with the second cheapest bundle. If the distance was greater than 200 (meaning that the first bundles price was more 200$ cheaper than the second bundles), print: " Buying the first bundle" and print the bundle using its toString method . Else check if the distance is a multiple of 10 or not. If it were, print: "Buying the second bundle" and print the bundle using its toString method.

Good luck