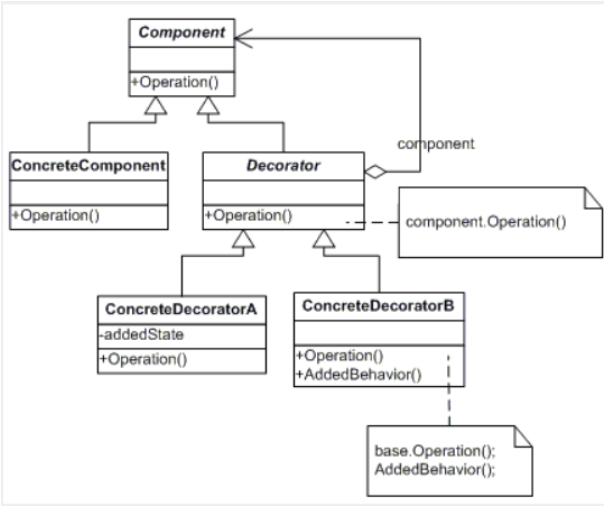
Quiz 3

Subject : Encapsulation and Inheritence
Due Date : 12.03.2022 23:59

Decorator Pattern

The Java language provides the keyword extends for subclassing a class. Those with enough knowledge of object-oriented programming know how powerful subclassing, or extending a class. By extending a class, you can change its behavior. The Decorator Pattern is used for adding additional functionality to a particular object as opposed to a class of objects. With the Decorator Pattern, you can add functionality to a single object and leave others like it unmodified. Any calls that the decorator gets, it relays to the object that it contains, and adds its own functionality along the way, either before or after the call. This gives you a lot of flexibility since you can change what the decorator does at runtime, as opposed to having the change be static and determined at compile time by subclassing. Decorators add the ability to dynamically alter the behavior of an object because a decorator can be added or removed from an object without the client realizing that anything changed. It is a good idea to use a Decorator in a situation where you want to change the behavior of an object repeatedly (by adding and subtracting functionality) during runtime.



Problem

Suppose you are developing a part of a simple Pizza Restaurant System. Your task is designing "Add Pizza" operation within specified design approach.

It is expected from you to give a **UML class diagram** showing the pattern usage in case of the pattern structure.

Operation: Add Pizza

Add a pizza to a given order with given toppings.

AddPizza < OrderID > <pizza type > < *topping* >

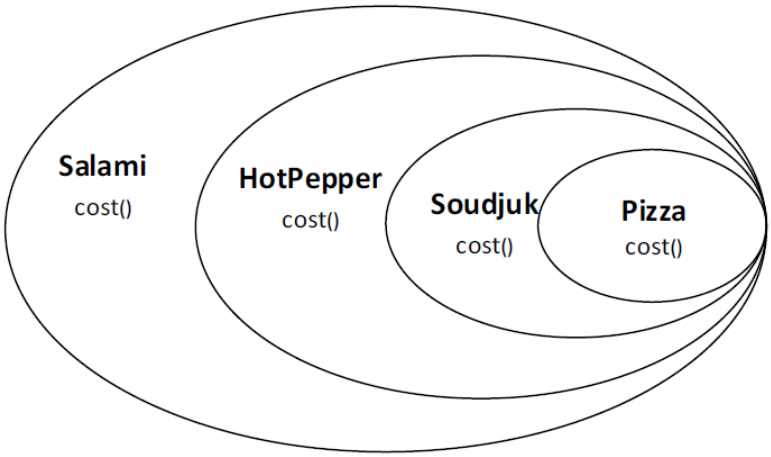
A pizza can include 3 toppings at most. There are 2 types of pizzas. These are:

- American Pan Pizza 45 tl
- Neapolitan Pizza 55 tl

A pizza can include 3 toppings at most. There are 4 types of toppings.

- Soudjouk 15 tl
- Salami 10 tl
- Pepper 5 tl
- Onion 3 tl

You should use decorator pattern for add topping mechanism. Add topping mechanism should look like this:



Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as data hiding.You should also be careful about showing how to act this mechanism in your design.

Notes

- Do not miss the submission deadline.
- Save all your work until the quiz is graded.
- You can ask your questions via Piazza and you are supposed to be aware of everything discussed on Piazza.
- You must submit your work with the file hierarchy as stated below:

→ <student id.zip>
  – report.pdf