Homework 3

This is an individual homework assignment. You may receive help from other class mates, but you may NOT copy their work. Copied work will be considered plagiarism and dealt with according to the syllabus and university policy.

Objective: For this assignment, you will demonstrate that you can properly implement the builder design pattern into existing Java code while conforming to all convention and expectations.

Description: Using design patterns is a requirement of contemporary object-oriented programming. Refactoring existing code to implement such functionality is an important skill to develop. Using the class diagram provided below in Figure 1, refactor the utility class and IceCream classes that you created in skill assignment 5 and 6 respectively. The refactoring must implement the capabilities and characteristics as described in the details below.

Assignment:

- Using the Driver.java class file attached to this assignment in Blackboard, refactor your Utilities.java and IceCream.java class files to include the following new functions:
  - Works with the provided Driver class test harness
    - When executed, the output should be like what is shown below in Table 1
    - ***The Driver.java class file logic is not to be modified***
      - If you wish to add extra credit functionality, do so without modifying lines 10-22 in the main method and add your extra credit logic after the required lines
  - Implement 2 new enumerations with appropriate values
    - Flavor: CHOCOLATE, VANILLA, STRAWBERRY
    - ConeType: REGULAR, SUGAR
  - You must refactor the IceCream class from skill assignment 6 to incorporate any fixes indicated by the professor
    - Refactor the IceCream class from skill assignment 6 to use the newly created 2 enumerations instead of String data types
      - ***Helpful tip:*** Make sure you refactor all necessary places (variables, mutators, ctor, etc.)
    - Add a nested builder class that fully conforms to all expectations and conventions related to the builder design pattern
      - ***Helpful tip:*** Make sure the builder is using the newly implemented enumerations instead of String data type values
  - Make sure to implement proper discount calculations based upon the discount value passed to the IceCream instance from the Driver class main method
    - You will need to appropriately modify the .calcGrandTotal() method in the Utilities class
      - Taxes are calculated on the entire subtotal, discounts are taken after taxes
      - For example: \$10.00 subtotal with 10% discount and 5% sales tax rate is calculated as follows:
        - Tax = 10.00 * .05 → \$0.50
        - Discount = 10.00 * 0.10 → \$1.00
        - Grand Total = 10.00 + 0.50 – 1.00 = \$9.50

- ➢ Upload your .JAVA code file(s) to the appropriate Blackboard assignment

```
The STRAWBERRY ice cream with a REGULAR cone and 3 scoops costs subtotal: $4.08 tax: $0.35 grand total: $4.43
```

Table 1 – Example Input/Output

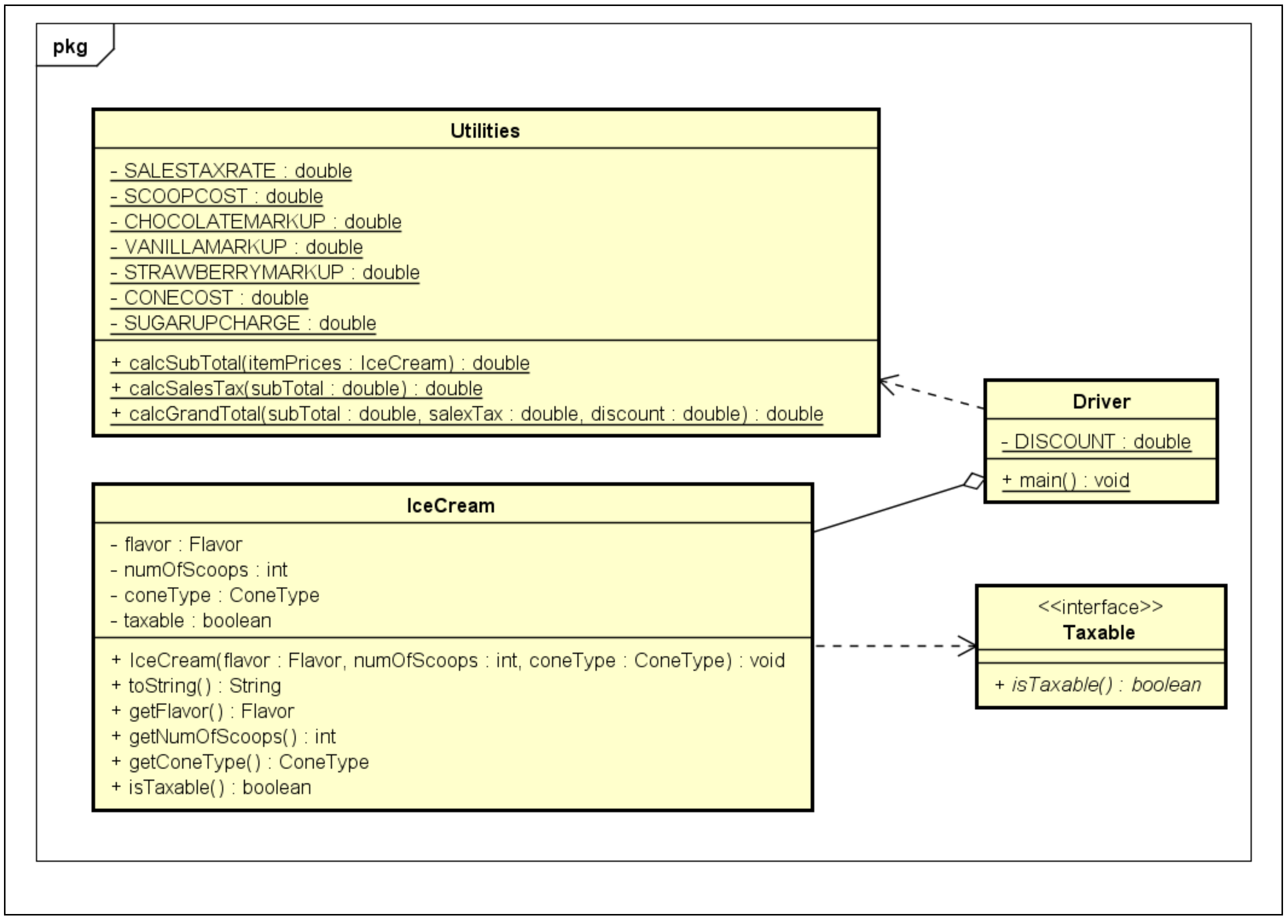

Figure 1 - Class Diagram