

CSE-3216: Software Design Patterns

Assignment 04

Submitted to:

Dr. Mohammad Samiullah

Assistant Professor

Department of Computer Science and Engineering

University of Dhaka

Submitted by:

Kazi Shadman Sakib

Roll: FH-97

3rd Year

Department of Computer Science and Engineering

University of Dhaka

**Pattern Used:**

To solve this assignment it was required to use the State Design Pattern. The State Design Pattern is a behavioral design pattern. It allows an object to alter its behavior when its internal state changes. The object will appear to change its class.

**Classes Used:**

* State : Interface
* NoCoinInsertedState : class
* CoinInsertedState : class
* PrepareState : class
* CupTakenState : class
* Beverage : class
* Inventory : class
* VendingMachine : class
* VendingMachine97 : class

**Assumptions Made:**

* A coffee vending machine goes through 4 states
  + NoCoinInsertedState
  + CoinInsertedState
  + PrepareState
  + CupTakenState
* Every state has 4 methods or functions, i.e, void insertCoin(double amount); void pressButton(int drinkType); void prepare(int drinkType); void cupTaken(int takenStatus);
* It is assumed that the Vending Machine only accepts 10, 20 and 50 cents from its user.
* It is also assumed that the Vending Machine has only two types of beverages, 1) Coffee (1.20 Dollars) and 2) Cappuccino (1.50 Dollars).
* In NoCoinInsertedState, we can only insert coins to the Vending Machine. If a coin is inserted, it sets the amount in the vending machine and then changes its state to CoinInsertedState.
* In CoinInsertedState, if we insert coins again then it will add the inserted amount with the vending machines already accounted amount. In this state, the user chooses the beverage type from the vending machine.
* If the user chooses an expensive beverage and did not add sufficient money to buy that beverage to the vending machine, then an exception will be generated with “Insufficient amount to buy this beverage!! Ejected your inserted money due to insufficient amount.”
* The vending machine will eject the inserted money of the user if the user has insufficient amount to buy a beverage.
* If the user chooses a beverage which is not available in the vending machine anymore then an exception will be generated with “Beverage is not available anymore.”
* In the PrepareState, the vending machine prepares the beverage and returns the changes to the user after calculating the payment. In this state, deduction of the beverage item count is calculated by the vending machine, as this beverage is sold to the user and is now 1 less in the vending machine.
* The Vending Machine will return the users 10, 20 and 50 cent of changes while the beverage is being prepared for the users.
* It is assumed that there are at most 5 cups of each beverage and after it runs out, the user may not be able to buy it anymore.
* It is assumed that, until the user does not take out the beverage cup from the vending machine, the vending machine will not take the next order. The user has to press any integer key to take out the beverage in the vending machine simulation. By pressing any integer key beverage will be taken out by the user.
* A beverage has only its name, id and price.
* The inventory maps to which aisles have which beverages using,

Map<Integer, Beverage> aisleToBeverageMap;

* The inventory also maps which beverage id number has how many quantities left using,

Map<Integer, Integer> beverageIdToCountMap

* The Vending Machine owner can add more beverages to the vending machine using, addBeverage(Beverage beverage) method.
* Only the VendingMachine class and its methods are visible to the users.