### Sample Question Paper

**Subject – C++ Programming**

### Assignment - Wonka Shop

**Time: 3.30 Hours. Total Marks: 100**

### Problem Statement:

Wonka the Chocolate Shop is an exclusive outlet for members only. They issue membership cards for a year which can be renewed. Our assignment is to implement a billing system for their self-checkout counter.

The customer can select chocolates from the shelves, place them in the cart, and proceed to the checkout counter.

**Functional Specification:**

1. **Initialize** : Initialize the necessary in-memory data structures in the system for Products in the shop and the Customer Information:
2. The chocolate shop product master is given in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Item Code** | **Item Name** | **Unit Price** | **Available Count** |
| 1 | Milky Way | 12.0 | 7 |
| 2 | Dark Secrets | 25.0 | 4 |
| 3 | Swiss Rendezvous | 32.0 | 12 |
| 4 | Pulp Fiction | 22.0 | 1 |

Use this data and create an in-memory data structure for the shop.

**Use following method to implement:**

**List<Product> initProducts()**

**This method has been provided to you in Client.h. Please do not change the method signature. You are free to add more data members and methods.**

**The *Product* *class* has been provided to you. Use the class as provided. You are free to add more methods or data members.**

1. When a customer is ready to check out, he/she will go to a counter and initiate the procedure by entering a valid membership id.

Some of the members/customers you can test the system with are given in the table below.

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Card Number** |
| 1 | Mr. Cadbury | 12345 |
| 2 | Mrs. Lindt | 67890 |

Create in-memory data structure of all the members using following method signature:

**public List<User> initUsers()**

**This method has been provided to you in Client.java. Please do not change the method signature. You are free to add more data members and methods.**

**The Bean User has been provided to you.**

1. **Authenticate a user based on users created above**:

If the verification fails due to an invalid membership number, the system should throw a user defined exception “InvalidUserException”. For invalid password, system should throw “InvalidPasswordException”. If authentication is successful, return status as true.

Use following method to authenticate:

**bool authenticate(String username,String password) throw InvalidUserException, InvalidPasswordException**

**This method has been provided to you in Client.h. Please do not change the method signature. You are free to add more data members and methods.**

***This is an incomplete problem statement. The Passion Challenge Assignment will require a lot of other features like Method / Operator Overloading, file I/O, exception handling etc… to design and implement.***

***For example in this problem statement, functionalities like processing order, generating bill etc...will be included. Corresponding method signatures will also be provided which you will have to use as it is.***