Institute of Information Technology
University of Dhaka
Bachelor of Science in Software Engineering (BSSE)
Lab Examination, 2021
SE 206: Object Oriented Concepts I
Marks: 20 # Time: 1.5 Hours

Professionalism	Excellence	Respect

Instructions:

1. You must follow OOP guidelines during the implementation. Code that works but does not follow the given instructions/ OOP guidelines (e.g: codes written in one class) will not be marked.
2. Bonus questions will only be marked if you can correctly answer it. There is no partial marking in bonus questions.
3. There is scope for partial marking in other questions. If your code is 80% functional and follows some OOP guidelines, you will receive some marks.
4. Using any resources is allowed. However, sharing resources with others and using any resource sharing platform are not allowed.
5. When you finish a question, submit it to the google classroom assignment and demonstrate it immediately. There is no extra time for demonstrating your solution.

1. A Simple Password Manager

Consider that you have to build a password manager application. The application takes the URL of a website, username and password as input and stores it in an encrypted format. At first, the user enters the application and chooses the **New Login Info** option from the console menu. Then, he needs to enter the URL, username and password of the website. These will be stored in an arraylist. The password must be stored in an encrypted format. In order to encrypt the password, it is left shifted n times, where n is less than the password length. For example, if your password is "hello" and n is 2, then the encrypted password is "llohe". When the user wants to see the details of the website, he will choose the **View Login Info** option. Then, he will enter the name of the website. The application will show the username and password of the website in plain text decrypted format. Note that, in order to decrypt the encrypted password, you can perform the right shift n times. For example, if you right shift "llohe" 2 times, it will be "hello".

Now, perform the following tasks.

a) Create a class named *AuthenticationInfo* which will contain the website *URL, username* and *password* as fields and *getter-setters* for these fields. Create a *constructor* to initialize these fields. **2**

b) Create a class named *EncryptionUtil* that contains two methods: *encrypt* and *decrypt*. The *encrypt* method receives a *String* and returns an encrypted version of that *String*. The *decrypt* methods work in a similar way, except that it implements the decryption function. **2**

c) Create a class named *Application* which has an *ArrayList* of *AuthenticationInfo* objects. It also has a method named *start* that shows the menu in the console. When the **New Login Info** option is selected and URL, *username* and *password* input is complete, it calls another method named *createLoginRecord* which is also inside the *Application* class. The *createLoginRecord* method receives an *AuthenticationInfo* object as parameter and returns an *AuthenticationInfo* object by encrypting the password in that object. This Returned *AuthenticationInfo* is stored inside the Arraylist. When the **View Login Info** option is selected and the URL is inputted, a method in the *Application* class named *viewLoginRecord* is called which receives a *String* URL and returns an *AuthenticationInfo* object where the *password* is in a decrypted format. **6**

Bonus: Alter the program such that multiple username and passwords for a single website URL can be created and stored. (2 marks)

2. **FruitPanda: A Fruit Delivery System for Local Sellers**

Consider a Fruit delivery system that enables the delivery of fresh fruits from local fruit sellers to consumers.

Now, implement the following tasks.

a) Create the following classes - **3**

i) *User* the contains the *firstName*, *lastName* and *phoneNumber* fields
ii) *Fruit* class that contains *name*, *variety, totalQuantityKg* and *pricePerKg*
iii) *Seller* that is a child of the *User* class and contains two extra fields - *address* and a list of *Fruit* objects
iv) *Order* that consists of a list of *Fruit* objects, *quantityInKg*, a reference of *Buyer* and a reference of *Seller* classes
v) *Buyer* that is a child of the *User* class and contains three extra fields - address, rewards and a reference of *Order* class
vi) *RoutePlanner* interface that contains a method called *calculateRoute*. This method takes two address *Strings* as parameters and calculates the shortest distance between these addresses. Then, it will return a String denoting the route name. (Leave the class as an interface, you don't need to implement the algorithms inside.). Create another class named *Delivery* that contains a reference of the *RoutePlanner* class.

Create getter-setters and constructors in each of these classes <u>if it is necessary</u>.

b. Create a method named *placeOrder* in the *Buyer* class. This method will take the *Seller* object as the parameter and display the list of *Fruits* (its name, variety and pricePerKg) that the seller has. Then, it will take the console input of the name of the *Fruits* and *quantityInKg* to be bought, and populate the *Order*. Then, it will return an *Order* object. **3**

c. Create a method named *deliver* in the *Delivery* class. It will receive an *Order* as its parameter. Then, it will use the RoutePlanner class to calculate the route. Then, it will decrease the *totalQuantityKg* in the Fruit class and increase the reward of the *Buyer* by 1% of the money spent for this order. **2**

d) Create a custom *Exception* named *NoFruitsException*. Trigger this exception only when the buyer either chooses a non-existent *Fruit* or the amount of the chosen *Fruit* is zero for the particular *Seller.* **2**

<u>GOOD LUCK!!</u>

[Please return the question with answer script.]