National University of Computer & Emerging Sciences, Karachi

Spring-2023 School of Computing (BSCS, BSCS-R, BSSE, BSCY, BSAI)

Assignment # 2

Subject: Object Oriented Programming - CS1004	**Post Date: 26/3/2023**
Total Marks: 40	**Due Date: 09/4/2023, 11:59 PM**
Course Instructors: Dr. Farooque Hassan Kumbhar, Dr. Abdul Aziz, Mr. Zain ul Hassan, Ms. Sobia Iftikhar, Ms. Eman Shahid, Ms. Abeer Gauher, Mr. Basit Ali, Ms. Javeria Farooq, Ms. Sumaiyah Zahid, Mr. Shahroz Bakht, Ms. Abeeha Sattar	

Instructions to be strictly followed.

- Each student should submit these files:
  - ***A zip of all source files named as "A2-Q#[StudentID]" where # is the question number and Student ID is your ID.***
  - ***A DOC file where they copy code for each question and screen shot of the output. This document contains all the questions, answer codes and output in sequence. Name this document as "A2-[StudentID].docx".***
  - ***All the submissions will be made on Google Classroom.***
- Each output should have STUDENT ID and NAME of the student at the top.
- It should be clear that your assignment would not get any credit if the assignment is submitted after the due date.
- Zero grade for plagiarism (copy/ cheating) and late submissions.

Question One. Total Marks /10 (2.5 Marks each)

Suppose you are designing a program for a juice shop that sells different types of juices. You should create a parent Juice class with common members, and three child classes for the three main juice categories: FruitJuice, VegetableJuice, and MixedJuice. Each of these child classes should have their own sub-classes, such as CitrusJuice, BerryJuice, and TropicalJuice for FruitJuice, and LeafyGreenJuice, RootVegetableJuice, and MixedVegetableJuice for VegetableJuice. Finally, the MixedJuice class can have its own sub-classes such as FruitAndVeggieJuice and SmoothieJuice. Fruit juices should have additional variable of *season*. Vegetable juices have *origin* string member.

All the juice classes have common members such as price, ingredients, and taste. For example, a citrus fruit juice might have oranges, lemons, and grapefruits as its ingredients, a price of 399, and a sweet and tangy taste. Similarly, a leafy green vegetable juice might have kale, spinach, and celery as its ingredients, a price of 499, and a bitter and earthy taste.

On the other hand, juice shops are also keeping track of Sales. Each sale has members of sale ID, CustomerName, total bill, numberOfFuitJuice, numberOfVegetableJuice and numberOfMixedJuice.

a) Draw a class diagram to represent the defined hierarchy. Write the code to design these classes with proper inheritance. Use constructors with initializers list to assign initial values and have setters/getters to change at a later stage. The program should be able to track the number of objects created in each class.

b) Include a print function in parent class that prints the details of the objects. Override the print function in all child classes where the function first call parent print function and subsequently print additional information. For example, object of citrus fruit juice call parent function to print its values but it also print specialized values of season as well. Each function should print a statement with your name and students id as such: “22k-1234 Fahim also likes citrus fruit juice”.

c) The main function should welcome the customer and get name first. Afterwards, it presents menu to the user where first menu has major categories and after selection, the menu presents the sub categories. After selection, an object of the selected juice is created. The process repeats until the user enter selects check out from the menu.

d) Write a global overloaded + operator, the operator should be able to add two sales objects and return updated sale object with sum of both bills, and juices count. The function should make sure that the customer name in both the sales is the same. The following operation should work SALES ob1, ob2; SALES ob3 = (ob1+ob2);

Question Two. Total Marks /10

Your University is updating software for grading and student record saving. The grading report is prepared by the administration based on the payment of tuition fees by individual student. Grades are awarded and shown on report if student semester fees is paid. If in case of not paid student is grades are removed from report as well as late penalty charges along with fees are charged.

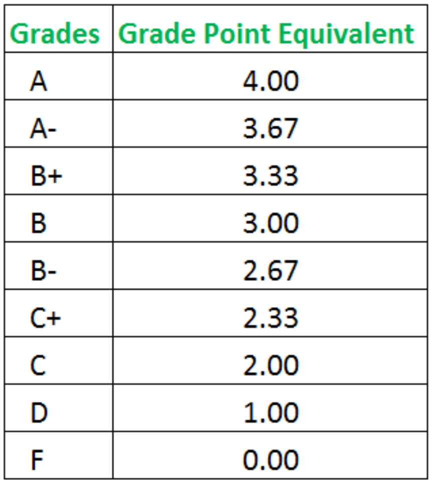
Courses: (1 Mark)
Attributes of courses will be course name, course code, credit hours. The functionality for courses will be:

1. Set the course information. (Contain all attributes of course)
2. Print the course information.
3. Show the credit hours.
4. Show the course number

Student: 7 Marks (0.5+1+0.5+2+3)
Attributes of student will be ID, student name, number of courses enrolled, course name, grade for each course. A variable indicating whether the student had paid fees or not. Every student is a person, and every student takes courses**.** The basic operations to be performed on an object of type student are as follows:

1. Set the student information.
2. Print the student information.
3. Calculate the number of credit hours taken.
4. Calculate the GPA. Take marks of N **courses**,
5. Calculate the GPA and the GPA percentage of the student.
6. Calculate the semester fees**.**

In case of nonpayment 5% of per credit fees is additionally included in fees voucher.



Grades	Grade Point Equivalent
A	4.00
A-	3.67
B+	3.33
B	3.00
B-	2.67
C+	2.33
C	2.00
D	1.00
F	0.00

Check whether the student is enrolled in particular course of not. Create a function that prints the result grades. If the student has paid semester fees, the grades and the GPA are shown. Otherwise, **"LOCKED GRADES"** are printed in place of each grade, the GPA is not shown, a message indicates that the grades are being held for nonpayment of the Semester fees, due amount along with penalty charges are shown. **(2 Marks)**

For test program create main function take your roll number as input for example: K22-0328. The batch code divided by half gives per credit hour charge for the students of year. Here 22k will give 11 as result. Later create **n** array of students. The **n** is second digit of your roll number. The first line indicates the number of students enrolled and the tuition rate per credit hour. The students' data is given thereafter.

Sample input:

```
3 11000
AMMAR K22-0933       Y 4
DLD EL1004              4 A
OOP CS2004              3 B
Communication Skills CS431    3 B
Islamiyat IS356           3 A
```

AMMAR ID is K22-0328, student has paid the tuition indicated in first line, and he is taking four courses.

OUTPUT SAMPLE 1

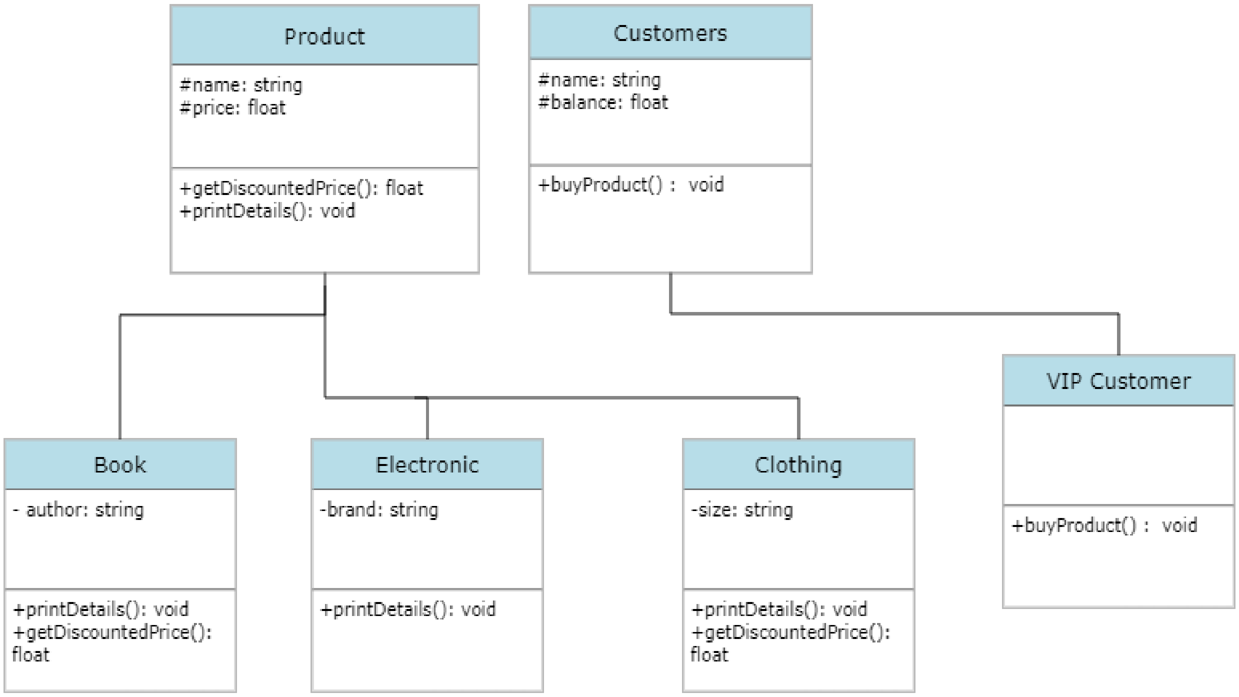
```
Student Name: AMMAR
Student ID: K22-0933
 Number of courses enrolled: 4
Course No Course Name Credits Grade
DLD  EL1004               4 A
OOP CS2004               3 B
Communication Skills CS431    3 B
Islamiyat  IS356            3 A
Total number of credit hours: 13
Semester GPA: 3.54
```

OUTPUT SAMPLE 2

```
Student Name: Bilal
Student ID: 20K-0978
Number of courses enrolled: 5
Course No Course Name Credits Grade
Calculus CaO234 4 LOCKED GRADES
DlD EL1004    4 LOCKED GRADE
English ENG378 3 LOCKED GRADE
Mathematics MTH346 3 LOCKED GRADE
Data Structures CS2008 3 LOCKED GRADE
Total number of credit hours: 17
*** Grades are being held for not paying the tuition. ***
Amount Due: 5865.00
```

Question Three. Total Marks /10

You are hired as a software engineer to automate the Retail store system.



Part A)

In this system, there is a base class `Product` for all the products in the store. This class has two data members: `name` and `price`, and two virtual functions: `getDiscountedPrice()` and `printDetails()`. The `getDiscountedPrice()` function returns the discounted price of the product, and the `printDetails()` function prints the details of the product. **(2 Marks)**

There are three derived classes: `Book`, `Electronic`, and `Clothing`. These classes inherit from the `Product` class and add some additional data members and functions. The `Book` class has an additional data member `author` and overrides the `printDetails()` function to include the author's name. The `Electronic` class has an additional data member `brand`. The `Clothing` class has an additional data member `size` and overrides the `getDiscountedPrice()` function to provide different discounts for clothing items. **(2 Marks)**

There is another a base class `Customer` for all the customers in the store. This class has two data members: `name` and `balance`, and a virtual function `buyProduct()` that allows the customer to buy a product. **The name variable must take your roll number as a value**. The `buyProduct()` function takes a pointer to a `Product` object and uses the `getDiscountedPrice()` function of the product to calculate the discounted price. If the customer has enough balance, the function deducts the discounted price from the balance and prints a message saying that the customer has bought the product. If the customer does not have enough balance, the function prints a message saying that the customer cannot buy the product. **(1 Mark)**

We have also defined a derived class `VIPCustomer` that inherits from the `Customer` class and adds no additional data members but overrides the `buyProduct()` function to provide additional discounts for VIP customers. **(1 Mark)**

Finally, in the `main()` function, we have created some products and customers and tested the program by allowing customers to buy products and printing relevant details. **(1 Mark)**

Part B)

The & operator is overloaded as a member function for the Product class. The operator&() function takes a constant reference to another Product object other. It calculates the average price of the two products and creates a new Product object with a concatenated name and the new price. The new product is then returned by the function. **(2 Marks)**

In the main() function, we create two Product objects p1 and p2 with prices of 10 and 2 respectively. We then use the overloaded & operator to create a new Product object p3, which has a concatenated name of "Book & Pen" and an average price of 6. We finally call printDetails() on p3 to verify that the new product has been created successfully.

Question Four. **Total Marks /10 (2 Marks Each)**

Imagine you have been hired by Binary Vibes to work on the gamming project. They have given a base class structure and have asked to complete the given requirements below:

GameObject
- Name : String - X : int - Y : int
+ GameObject(string, int, int) + draw() : Void

Now, create three subclasses that inherit from the "GameObject" class: "Player," "Enemy," and "PowerUp." Health should be an additional property for the Player class, damage should be an additional property for the Enemy class, and effect should be an additional property for the PowerUp class.

1. Utilizing the protected inheritance mode, implement the classes (GameObject, Player, Enemy & PowerUp). Each subclass should have constructors that initialize their properties, furthermore the attribute "Name" in "GameObject" class is constant and it should be initialized by your valid student ID. The format of valid student ID is XXK-XXXX, whereas "X" is the number. Additionally, if required, create accessors and mutators.
2. To incorporate the additional properties of the Player, Enemy, and PowerUp objects, override the draw() method in each subclass.
3. Create a new class called Game with a dynamic array of GameObject objects in it. Implement a functionality of DrawAll() in it. DrawAll() is a Game class method that calls the draw() method on each GameObject object while looping through the array.
4. In the main() function, create instances of the Player, Enemy, and PowerUp classes and add them to the array of GameObject objects in the Game class. Then call the drawAll() method on the Game object to draw all the objects in the array.
5. Overload a "==" operator for Player class for comparing the health of two players.