CS1004 - Object-oriented Programming (OOP)
Assignment # 2
BS(CS)

Max Points: **80**
Due Date: **Sunday, April 17, 2022, 11:59 p.m.**

Carefully read the following instructions!

- It should be clear that your assignment would **not get any credit** if the assignment is submitted after the **due date**. **No** assignment will be **accepted after the due date**.
- Strict action will be taken if the submitted solution is copied from any other student.
- If you people find any mistake or confusion in the assignment (Question statement), please consult before the deadline. After the deadline no queries will be entertained in this regard.
- For any query, feel free to email at: **farah.sadia@nu.edu.pk**
- Submission: Submission will only be accepted through **GOOGLE CLASSROOM**.
- Submit all your codes with your Student ID and task number. **"K211234_Q1"**.
- Each output must contain your ID and name on the screen.
- Every code should be with proper **commenting**.
- Plagiarism is punishable with zero grades in the task.

Question # 01: *Programming (Total Marks: 20)*

Create an inheritance hierarchy that a bank might use to represent customers' bank accounts. All customers at this bank can deposit (i.e., credit) money into their accounts and withdraw (i.e., debit) money from their accounts. More specific types of accounts also exist. **Savings accounts**, for instance, earn interest on the money they hold. **Checking accounts**, on the other hand, charge a fee per transaction (i.e., credit or debit). Create an inheritance hierarchy containing base class **Account** and derived classes **SavingsAccount** and **CheckingAccount** that inherit from class **Account**.

1. **Account** should include one data member of type **double** to represent the account **balance**. The class should provide a **default constructor** and **parameterized constructor.** The constructor should validate the initial balance to ensure that it's greater than or equal to **0.0**. If not, the balance should be set to 0.0 and the constructor should display an error message, indicating that the initial balance was invalid.
    a. Member functions **Credit()** should add an amount to the current balance.
    b. Member functions **Debit()** should withdraw money from the Account and ensure that the debit amount does not exceed the Account's balance. If it does, the balance should be left unchanged and the function should print the message **"Debit amount exceeded account balance."**
    c. **getBalance()** should return the current balance.

2. **SavingsAccount** should inherit the functionality of an Account, but also include a data member of type **double** indicating the **interest rate (percentage)** assigned to the Account.
    a. The class should provide a **default constructor** and **parameterized constructor.** Constructor should receive the initial balance, as well as an initial value for the interest rate.
    b. Member function **double calculateInterest()** should determine this amount by multiplying the interest rate by the account balance. [Note: SavingsAccount should inherit member functions credit and debit as is without redefining them.]
3. **CheckingAccount** should inherit from base class Account and include an additional data member of type **double** that represents the fee charged per transaction.
    a. The class should provide a **default constructor** and **parameterized constructor** should receive the initial balance, as well as a parameter indicating a **fee amount**.
    b. Class should redefine member functions **credit()** and **debit()** so that they subtract the fee from the account balance whenever either transaction is performed successfully.

After defining the classes in this hierarchy, write a program that creates objects of each class and tests their member functions. Add interest to the SavingsAccount object by first invoking its calculateInterest function, then passing the returned interest amount to the object's credit function.

Question # 02: *Programming (Total Marks: 20)*

Design a class to perform various matrix operations. A matrix is a set of numbers arranged in rows and columns. Therefore, every element of a matrix has a row position and a column position. If A is a matrix of five rows and six columns, we say that the matrix A is of the size **5x6** and sometimes denote it as A 5x6.

Clearly, a convenient place to store a matrix is in a two-dimensional array. Two matrices can be added and subtracted if they have the same size. Suppose $\mathbf{A = [a_{ij}]}$ and $\mathbf{B = [b_{ij}]}$ are two matrices of the size **m x n**, in which $\mathbf{a_{ij}}$ denotes the element of A in the i^{th} row and the j^{th} column and so on. The sum and difference of A and B are given by:

$\mathbf{A+B = [a_{ij} + b_{ij}]}$

$\mathbf{A-B = [a_{ij} - b_{ij}]}$

The multiplication of A and B **(A*B)** is defined only if the number of columns of A is the same as the number of rows of B. If A is of the size m * n and B is of the size n * t, then A * B = $\mathbf{[c_{ik}]}$ is of the size m*t and the element $\mathbf{c_{ik}}$ is given by the formula:

$\mathbf{C_{ik} = a_{i1}b_{1k} + a_{i2}b_{2k} + \ldots + c_{in}b_{nk}}$

Design and implement a Class matrixType that can store a matrix of any size. Overload the operators +, - and * to perform the addition, subtraction, and multiplication operations, respectively, and overload the operator << to output a matrix. Also, write a test program to test various operations on the matrices.

Question # 03: *Programming (Total Marks: 20)*

Diner's is a top-notch brand that sells shirts, pants and ties to its dedicated buyers. Apart from selling shirts, pants and ties, Diner's also offers complete suits that comprise shirt, pant and a tie. There is also a special **tagID** for each clothing type. The classes for this scenario are presented in Figure. Read the assumptions below and then perform the tasks.

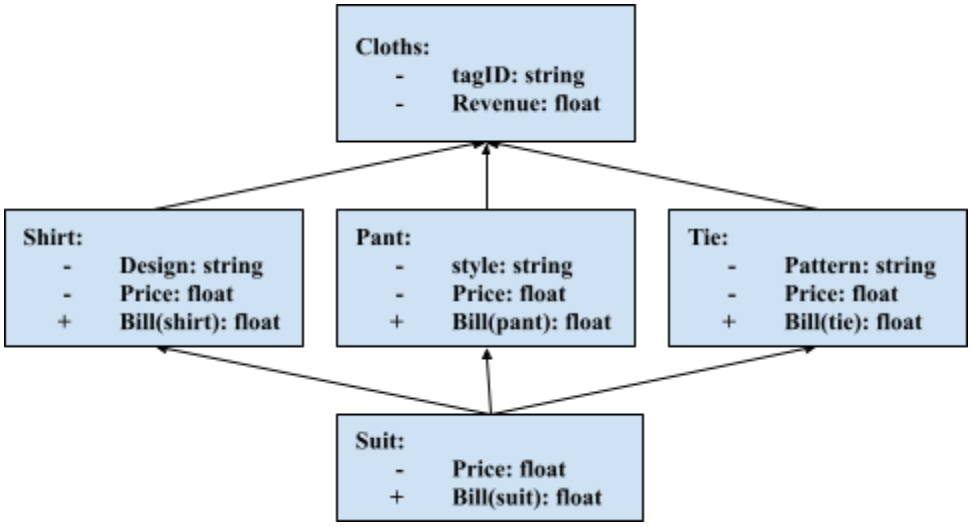
Assumptions:

1. You may create or modify any variables and override any functions if needed to satisfy the requirements of the question. But their role in the program has to be justifiable.
2. The design for shirts can either be **formal, traditional** or **casual**. The unit price for each of these types is **Rs. 1499**.
3. **Shirts** and **pants** are taxable items, but ties are not. The **tax rate** for **shirts** is **7.5%** of the price and for **pants** it is **4.5%** of the price.
4. The pant styles available are either **bell-bottomed, straight fit** or **narrow fit**. The unit price for pant is **Rs. 1199** for **bell-bottomed, Rs. 1599** for **straight fit** and **Rs. 1999** for **narrow fit**.
5. The pattern for tie can either be stripes or checks.
6. The price for **striped tie** is **Rs. 699** and **checkered tie** is **Rs. 799**.

Tasks to be performed:

1. Declare **variables** and also provide suitable implementation for **default and parameterized constructor(s)** of each class.

2. Add a method **bill( )** in the class **Suit** that calculates the **total price** of a suit based on the items present in the suit.
3. Overload the function **bill( )** in **Shirt** class to accept discount vouchers **(7-letter string as discount code)** which is **23%** along with the shirt object and return the discounted price.
4. Overload the " < " operator such that it displays if a given Shirt instance gives more profit than a Pant instance.
5. For keeping track of the **inventory**, there must be a mechanism to find how much of each individual item (shirt, pant or tie) is remaining in stock. Also provide a mechanism to see the current revenue (overall profit). Keep in mind that each type of clothing has a different price.
6. Provide a copy constructor for copying objects of Pant class.
7. Provide a global mechanism for tax calculation such that it has access to all of Zed
8. Clothing's revenue details.



Question # 04: *Programming (Total Marks: 20)*

Implement the following scenario in C++:

1. All the values are required to be set through the constructor's parameter.
2. Provide necessary accessor functions where required.
3. Create an object of class bus by initializing it through a parameterized constructor in the main function and display all data members by calling the display function of the class bus.