National University of Computer & Emerging Sciences,
Karachi
Spring-2024, School of Computing (BSCS, BSSE, BSAI, BSCY)

Assignment # 3

Subject: Object Oriented Programming Total Marks: 30	**Post Date: 29th April 2024** **Due Date: 5th May 2024**
Course Instructors: Mr. Basit Ali, Ms. Sobia Iftikhar, Ms. Sumaiyah Zahid, Ms. Abeeha Sattar, Ms. Bakhtawer Abbasi, Ms. Atiya Jokhio, Mr. Minhal Raza, Ms. Rafia Shaikh, Ms. Abeer Gauher	

Instructions to be strictly followed.

1. **Each student should submit these files:**
   a. **A zip of all source files named as "A2-Q#[StudentID]" where # is the question number and Student ID is your ID.**
   b. **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 "A1- [StudentID].docx".**
   c. **All the submissions will be made on Google Classroom.**
2. **Each output should have STUDENT ID and NAME of the student at the top.**
3. **It should be clear that your assignment would not get any credit if the assignment is submitted after the due date.**
4. **Zero grade for plagiarism (copy/ cheating) and late submissions**

Question 01:

Times Medico is committed to providing customers with prescribed medicines efficiently. With various categories of medicines and specialized staff roles, Times Medico ensures smooth operations while maintaining revenue. Being a programmer, You have to design classes with following details

1) **Medicine Class:** Attributes: name, formula, retail price, manufacture date, expiration date.
   - Functions:
     - Constructor(s) to initialize attributes.

- Getters and setters for attributes.
- Virtual function for printing medicine details.

2) **Tablet Class (Derived from Medicine):** Additional Attribute: sucrose level (0 to 1).
    a. Functions:
        i. Constructor(s) to initialize Tablet-specific attributes.
        ii. Overridden function for printing tablet details.
3) **Capsule Class (Derived from Medicine):**
    a. Additional Attribute: absorption percentage (1 to 100).
    b. Functions:
        i. Constructor(s) to initialize Capsule-specific attributes.
        ii. Overridden function for printing capsule details.
4) **Syrup Class (Derived from Medicine):**
    - Functions:
        - Constructor(s) to initialize Syrup-specific attributes.
        - Overridden function for printing syrup details.
5) **Pharmacist Class**
    - Functions:
        - Search_Medicine() to search and print medicine details based on formula.
6) **Counter Class**
    - Functions:
        - Search_Medicine() to search and print medicine details based on name.
        - Update_Revenue() to update overall revenue.

- **Flow of operation:**
  The customer enters the medico and presents a prescription to the counter staff. The counter staff forwards this prescription to the pharmacist who checks and recommends the appropriate medicine type. The counter staff then collects the payment, hands over the medicines and updates the overall revenue.
- **<u>Tasks to be performed:</u>**
    - Identify all the classes, attributes and functions in the above scenario. Only make a skeleton of classes and give prototypes of functions.
    - Overload the **"= ="** operator to compare two medicines and find if both are going to expire in the same year.

Question 02

Design a C++ program for managing a collection of pets using specialized templates and generic programming concepts. The program should include a base Pet class and specialized pet classes (Cat, Dog, Bird) to demonstrate inheritance, polymorphism, and template specialization.

Pet Class Template: Implement a template Pet class that serves as the base for all pet types.

Use a template parameter T to make the Pet class generic.

Include the following features in the Pet class:

Constructor to initialize the pet's name and age.

A pure virtual function **makeSound()** to represent the pet's unique sound.

B. Specialized Pet Classes: Define specialized pet classes **(Cat, Dog, Bird)** that inherit from the Pet<T> class.

Each specialized pet class should implement the **makeSound()** function to produce a specific sound (Meow, Woof, Chirp).

C. Write a main() function to demonstrate the functionality of the pet system.

Create instances of different pet types **(Cat, Dog, Bird)** and interact with them using polymorphism.

Display each pet's information **(name, age)** and make them produce their unique sound using the **makeSound()** method.

Question 03

You are tasked with implementing a generic matrix class in C++ that supports basic matrix operations using operator overloading. The matrix class should be designed using generic programming techniques to handle matrices of different numeric types (int, double, etc.) and support operations such as addition, subtraction, and multiplication.

A. Generic Matrix Class Template:

Implement a template Matrix class that can represent a 2D matrix of any numeric type (T). The class should support the following functionalities:

Constructor to initialize the matrix dimensions (rows and cols).

Method to access and modify individual elements of the matrix (get and set).

Overloaded operators (+, -, *) to perform matrix addition, subtraction, and multiplication.

Abstract Method to displaying the matrix.

Specialized Matrix Classes:

Implement specialized matrix classes (IntMatrix, DoubleMatrix) that inherit from the Matrix<T> base class. Override the display() method to display matrices of specific numeric types (int, double).

write a main() function to demonstrate the functionality of the matrix classes. Create instances of IntMatrix and DoubleMatrix, perform matrix operations (addition, subtraction, multiplication), and display the results using the display() method.

Question 04

Imagine you're developing a simulation software for autonomous drones. The software includes a Drone abstract class that encapsulates common properties and functionalities of drones, such as their position (latitude and longitude), altitude, and speed. Additionally, it provides methods like adjustAltitude(float meters) to change the drone's altitude and setSpeed(float speed) to modify its speed.

a) Your software also includes interfaces for different drone capabilities. The Flyable interface outlines methods for movement in three-dimensional space, including takeoff(), land(), and navigateTo(float latitude, float longitude, float altitude). The Scannable interface defines methods for scanning the environment, such as scanArea(float radius) to detect objects within a specified radius.

b) Using your software, create a class named ReconDrone that extends the Drone class and implements both the Flyable and Scannable interfaces. The ReconDrone class should have additional attributes like cameraResolution and maxFlightTime, along with methods to control its flight and scanning operations.

Override the navigateTo(float latitude, float longitude, float altitude) method in ReconDrone to calculate the distance between the current position and the target position using GPS coordinates. Display the estimated time required for the drone to reach the destination based on its current speed and altitude.

Implement the scanArea(float radius) method in ReconDrone to simulate scanning the environment within the specified radius. Display the number of objects detected and their coordinates.

c) Create a main function to demonstrate the functionality of the ReconDrone class. Instantiate a ReconDrone object, set its initial position and altitude, then navigate it to a different location while scanning the area. Display the results of the navigation and scanning operations.

e) Handle any potential exceptions that may occur during the navigation and scanning processes, such as GPS signal loss or communication errors with the drone hardware.