**NATIONAL UNIVERSITY OF COMPUTER & EMERGING SCIENCES**

**ISLAMABAD CAMPUS**

**CS217 Object Oriented Programming- Fall 2020**

**ASSIGNMENT- 5**

**Section (A, B, C, D, and F)**

**Due Date: Friday 11th December 2020 at 11:59 pm on Google Classroom**

**Instructions:**

1. Assignments are to be done individually. You must complete this assignment by yourself. You cannot work with anyone else in the class or with someone outside of the class. The code you write must be your own and you must understand each part of coding. You are encouraged to seek help from the instructors through email, on google classroom or individually visiting their respective offices.

2. The AIM of this assignment is to practice with Inheritance in C++.

3. No late assignments will be accepted.

4. Displayed output should be well mannered and well presented. Use appropriate comments and indentation in your source code.

5. Plagiarism:

Plagiarism of any kind (copying from others and copying from internet, etc.,) is not allowed. If found plagiarized, you will be awarded zero marks in the assignment. Repeating such, an act can lead to strict disciplinary actions and failure in course.

**Submission Guidelines:**

We will be using auto-grading tools, so failure to submit according to the following format would result in zero marks in the relevant evaluation instrument:

i) For each question in your assignment, make a separate .cpp file e.g. for question 1, make q1.cpp and so on. Each file that you submit must contain your name, student-id, and assignment # on top of the file in the comments.

ii) Combine all your work in one folder. The folder must contain only .cpp files (no binaries, no exe files etc.,).

iii) Run and test your program on a lab machine before submission.

iv) Rename the folder as ROLL-NUM\_SECTION (e.g. 19i-0001\_B) and compress the folder as a zip file. (e.g. 19i-0001\_B.zip).

v) Submit the .zip file on Google Classroom within the deadline.

vi) Submission other than Google Classroom (e.g. email etc.) will not be accepted.

vii) The student is solely responsible to check the final zip files for issues like corrupt file, virus in the file, mistakenly exe sent. If we cannot download the file from Google classroom due to any reason, it will lead to zero marks in the assignment.

**Total Points: 100**

**Problem1**

Design a class named **Employee**. The class should keep the following information in member variables:

**Employee name**

**Employee number**

**Hire date**

Write only one constructor (**Constructor with Default Parameters**) and the appropriate accessor and mutator functions for the class.

Next, write a class named **ProductionWorker** that is derived from the Employee class. The ProductionWorker class should have member variables to hold the following information:

**Shift (an integer)**

**Hourly pay rate (a double)**

The workday is divided into two shifts: day and night. The shift variable will hold an integer value representing the shift that the employee works. The day shift is shift 1 and the night shift is shift 2. Write one or more constructors and the appropriate accessor and mutator functions for the class. Demonstrate the classes by writing a program that uses a **ProductionWorker** object.

**Problem 2**

In a particular factory a shift supervisor is a salaried employee who supervises a shift. In addition to a salary, the shift supervisor earns a yearly bonus when his or her shift meets production goals. Design a **ShiftSupervisor** class that is derived from the Employee class you created in **Problem1**. The **ShiftSupervisor** class should have a member variable that holds the annual salary and a member variable that holds the annual production bonus that a shift supervisor has earned. Write one or more constructors and the appropriate accessor and mutator functions for the class. Demonstrate the class by writing a program that uses a **ShiftSupervisor** object.

**Problem 3**

In a particular factory, a team leader is an hourly paid production worker who leads a small team. In addition to hourly pay, team leaders earn a xed monthly bonus. Team leaders are required to attend a minimum number of hours of training per year. Design a **TeamLeader** class that extends the **ProductionWorker** class you designed in **Problem1**. The **TeamLeader** class should have member variables for the monthly bonus amount, the required number of training hours, and the number of training hours that the team leader has attended. Write one or more constructors and the appropriate accessor and mutator functions for the class. Demonstrate the class by writing a program that uses a TeamLeader object.

**Problem 4**

***class Time***

***{***

***protected:***

***int hour;***

***int min;***

***int sec;***

***public:***

***// Default constructor***

***Time()***

***{***

***hour = 0; min = 0; sec = 0;***

***}***

***// Constructor***

***Time(int h, int m, int s)***

***{***

***hour = h; min = m; sec = s;***

***}***

***// Accessor functions***

***int getHour() const***

***{***

***return hour;***

***}***

***int getMin() const***

***{***

***return min;***

***}***

***int getSec() const***

***{***

***return sec;***

***}***

***};***

Design a class called **MilTime** that is derived from the Time class (See class time above). The **MilTime** class should convert time in military (24-hour) format to the standard time format used by the Time class.

The class should have the following member variables:

**milHours:** Contains the hour in 24-hour format. For example, 1:00 pm would be stored as 1300 hours, and 4:30 pm would be stored as 1630 hours.

**milSeconds:** Contains the seconds in standard format.

The class should have the following member functions:

***Constructor():*** The constructor should accept arguments for the hour and seconds, in military format. The time should then be converted to standard time and stored in the hours, min, and sec variables of the Time class.

***setTime():*** Accepts arguments to be stored in the milHours and milSeconds variables. The time should then be converted to standard time and stored in the hours, min, and sec variables of the Time class.

**getHour():** Returns the hour in military format.

**getStandHr():** Returns the hour in standard format.

Demonstrate the class in a program that asks the user to enter the time in military format. The program should then display the time in both military and standard format.

***Input Validation: The MilTime class should not accept hours greater than 2359, or less than 0. It should not accept seconds greater than 59 or less than 0.***

**Problem 5**

Design a class named **TimeClock**. The class should be derived from the **MilTime** class you designed in Problem 4. The class should allow the programmer to pass two times to it: starting time and ending time. The class should have a member function that returns the amount of time elapsed between the two times. For example, if the starting time is 900 hours (9:00 am), and the ending time is 1300 hours (1:00 pm), the elapsed time is 4 hours.

***Input Validation: The class should not accept hours greater than 2359 or less than 0.***

**Problem 6**

Design a class named **PersonData** with the following member variables:

* lastName
* firstName
* address
* city
* state
* zip
* phone

Write the appropriate accessor and mutator functions for these member variables. Next, design a class named **CustomerData**, which is derived from the **PersonData** class. The **CustomerData** class should have the following member variables:

* customerNumber
* mailingList

The customerNumber variable will be used to hold a unique integer for each customer. The mailingList variable should be a bool. It will be set to true if the customer wishes to be on a mailing list, or false if the customer does not wish to be on a mailing list. Write appropriate accessor and mutator functions for these member variables. Demonstrate an object of the **CustomerData** class in a simple program.

**Problem 7**

A retail store has a preferred customer plan where customers may earn discounts on all their purchases. The amount of a customer s discount is determined by the amount of the customer s cumulative purchases in the store.

* When a preferred customer spends $500, he or she gets a 5% discount on all future purchases.
* When a preferred customer spends $1,000, he or she gets a 6% discount on all future purchases.
* When a preferred customer spends $1,500, he or she gets a 7% discount on all future purchases.
* When a preferred customer spends $2,000 or more, he or she gets a 10% discount on all future purchases.

Design a class named **PreferredCustomer**, which is derived from the **CustomerData** class you created in Problem 6. The **PreferredCustomer** class should have the following member variables:

* purchasesAmount (a double)
* discountLevel (a double)

The **purchasesAmount** variable holds the total of a customer s purchases to date. The discountLevel variable should be set to the correct discount percentage, according to the store s preferred customer plan. Write appropriate member functions for this class and demonstrate it in a simple program.

***Input Validation: Do not accept negative values for any sales figures.***

**Problem 8**

***class GradedActivity***

***{***

***protected:***

***double score; // To hold the numeric score***

***public:***

***// Default constructor***

***GradedActivity()***

***{***

***score = 0.0;***

***}***

***// Constructor***

***GradedActivity(double s)***

***{***

***score = s;***

***}***

***// Mutator function***

***void setScore(double s)***

***{***

***score = s;***

***}***

***// Accessor functions***

***double getScore() const***

***{***

***return score;***

***}***

***virtual char getLetterGrade() const;***

***};***

Design an **Essay** class that is derived from the **GradedActivity** class presented above. The Essay class should determine the grade a student receives on an essay. The student s essay score can be up to 100, and is determined in the following manner:

* Grammar: 30 points
* Spelling: 20 points
* Correct length: 20 point
* Content: 30 points

Write appropriate member functions for this class and demonstrate the class in a simple program.