Homework 5

100 Points

Classes

Project: A variation of the **Trivia Game** // see next pages

- Project 19, Page 809 // 8th edition

- Project 16, Page 797 // 7th edition

Grading

Create a project consisting of four files (// See Examples: Rectangle):

Question.h	-20
Question.cpp	
Homework5.cpp	
TriviaQuestions.txt	
Create an array of objects (read from file)	-20
Randomize the array of questions	-10
Randomize the answers for each question	-10
Play Trivia Game	-20
The main() function	-10
Report (Design – UML Diagram included)	-10

Write a short report (not more than one or two pages) to explain the design of your program:

→ Show how data are organized in classes and arrays
// What are the classes used in this program?
// UML diagram
// What are the arrays used in this program?

→ Show the structure of your program
// structure chart or pseudocode
// What are the stand-alone functions used in this program

Run the program as required and save the output at the end of the source file as a comment. Compress the source file, input and output files (if any), and the report, and upload the compressed file: 22B_LastName_FirstName_H5.zip

Project: A variation of the **Trivia Game**

- **Project 19, Page 809** // 8th edition

- **Project 16, Page 797** // 7th edition

Create your own input file with questions about classes in C++. You have to provide 18 questions, but your program should work with any number of questions. On the first line in the file place the number of questions. Use it to dynamically allocate the array of questions. On the next page you have 9 questions. Use these questions in your input file (do not forget to add 9 more such questions). The correct answer is always the first one. You will have to keep track of the correct answer after randomizing the answers.

Prompt the user to enter the names of the players (first name only)

A. Randomize the array before playing the game.
B. Randomize the answers for each question before playing the game.
C. Play the game: Each player must answer 3 questions. Question 1 – Player 1, if incorrect, display a "Sorry" message and show the correct answer; if correct, show a "Congratulations" message, and continue: Question 2 – Player 2, Question 3 – Player 1, Question 4 – Player 2, and so on.
D. Display the winner.

Prompt the user to enter the names of the next two players, and repeat steps A, B, C, and D, until the user enters #(to stop).

Run the program as required and save the output at the end of the source file as a comment. Compress the source file, input and output files (if any), and the report, and upload the compressed file: 22B_LastName_FirstName_H4.zip

Extra Credit: In addition to the Question class design and use a Player class. Store the players into an array (assume there are not more than 50 players). The player's array should be sorted by name. Do not allow duplicate names, but allow the same player to play again (Hint: for each new player store in addition to his name, other member variables of your choice, including a password). Write the player's array to an output file. Create a project consisting of the following files (// See Examples: Rectangle):

Question.h
Question.cpp
Player.h
Player.cpp // if needed
Homework5.cpp
TriviaQuestions.txt

9
An object is, conceptually, a self-contained unit consisting of
Attributes(data) and procedures(functions)
Attributes
Procedures
Data

Encapsulation refers to
Combining data and code into a single object
Storing data into an object
Creating functions for an object
Protecting data

Data hiding refers to an object's ability to
Hide its data from code that is outside the object
Hide its data from code that is inside the object
Hide its data
Hide its procedures

A class is not an object, but it is
A description of an object
An instance of an object
Code that specifies the attributes that a particular type of object may have
Code that specifies the functions that a particular type of object may have

An object is not a class, but it is
An instance of a class
A description of a class
Code that specifies the attributes that a class may have
Code that specifies the functions that a class may have

The operator that identifies the function as a member of a class is
::
:
->
.

The complete name of the operator that identifies the function as a member of a class is
Scope resolution operator
Scope operator
Identification operator
Scope identification operator

Some programmers refer to mutators as
Setter functions
Getter functions
Change functions
Accessor functions

Some programmers refer to accessors as
Getter functions
Setter functions
Getter functions
Change functions
Mutators functions