COMP2710: Homework 5

Points Possible: 20
Note: You do not need to submit hard copies.

Goals:

- To learn how to use structures
- To learn how to use strings
- To learn creating multiple versions via conditional compilation
- To use arrays (Note: this topic was covered in Homework 4)
- To design and implement functions (Note: this topic was covered in Homework 3 and 4)
- To perform unit testing (Note: this topic was covered in Homework 3 and 4)

In this homework assignment, you will write a simple trivia quiz game. You program first allows players to create their trivia questions and answers. Then, you program asks a question to the player, input the player's answer, and check if the player's answer matches the actual answer. If so, award the player the award points for that question. If the player enters the wrong answer your program should display the correct answer. When all questions have been asked display the total amount that the player has won.

Please perform the following steps to finish this assignment.
- **Step 1:** Creating a Trivia **structure** that contains information about a single trivia question. This structure must contain a **string** for the question, a **string** for the answer to the question, and an integer representing points the question is worth. Note that a harder question should be worth more points.
- **Step 2:** Create an array of Trivia using the structure defined in Step 1.
- **Step 3:** Design and implement a function that initialize the array by hard-coding the following three (3) trivia questions (including answers and award points). The
    - Trivia 1:
        - Question: How long was the shortest war on record? (Hint: how many minutes)
        - Answer: 38
        - Award points: 100
    - Trivia 2:
        - Question: What was Bank of America's original name? (Hint: Bank of Italy or Bank of Germany)?
        - Answer: Bank of Italy
        - Award points: 50
    - Trivia 3:
        - Question: What is the best-selling video game of all time? (Hint: Call of Duty or Wii Sports)?
        - Answer: Wii Sports

- ▪ Award points: 20
- **Step 4:** Design and implement a function to create and add new Trivia into the array. Please remember to check that adding a new trivia does not exceed the array's capacity (i.e., size).
- **Step 5:** Design and implement a function that asks a question to the player, input the player's answer, and check if the player's answer matches the actual answer. If so, award the player the dollar amount for that question. If the player enters the wrong answer your program should display the correct answer.
  - o Input: a Trivia array, size of array, index of the question in the array
  - o Output: int − 0 indicates success and 1 indicate failure.
  - o Note: A possible failure is that the index parameter is larger than the array size.
- **Step 6:** Write a test driver to perform unit testing for the function implemented in Step 5. You must cover at least the following cases: (see **Fig. 1** on page 3 for the sample user interface.)
  - o Case 1: ask the question of the first trivia in the array.
    - ▪ Correct answer
    - ▪ Wrong answer
  - o Case 2: ask the question of the last trivia in the array.
    - ▪ Correct answer
    - ▪ Wrong answer
  - o Case 3: ask the question of any trivia in the middle of the array.
    - ▪ Correct answer
    - ▪ Wrong answer
  - o Case 4: ask the question that does not exist (i.e., the index of the trivia is larger than the array size).
- **Step 7:** Write the main function that performs the following: (see Fig. 2 on page 4 for the sample user interface)
  - o Create hard-code trivia quizzes (i.e., questions/answers/awards) (Note: just call the function implemented in step 3).
  - o Create more than 1 trivia quiz from a keyboard (Note: just call the function implemented in step 4).
  - o Write a for loop; in each iteration do the following:
    - ▪ asks a question to the player,
    - ▪ input the player's answer,
    - ▪ if the player's answer matches the actual answer, then award the player the award points for that question
    - ▪ else (i.e., the player enters the wrong answer) your program should display the correct answer.
  - o When all questions have been asked display the total award points the player has won.
- **Step 8:** Creating two versions using conditional compilation.
  - o Version 1: simply run the test driver implemented in Step 6.

o Version 2: a regular version run the main function implemented in Step 7. Note: this version does not include the test driver.

You must provide the following user interface for the **debugging version**. The user input is depicted as **Bold**, but you do not need to display user input in bold. In this version, your program must run the test driver you build in Step 6.

```
*** This is a debugging version ***
Unit Test Case 1.1: Ask the question of the first trivia in the
array. The tester enters a correct answer.
Question: How long was the shortest war on record? (Hint: how many
minutes)
Answer: 38
Case 1.1 passed...

Unit Test Case 1.2: Ask the question of the first trivia in the
array. The tester enters an incorrect answer.
Question: How long was the shortest war on record? (Hint: how many
minutes)
Answer: 85
Case 1.2 passed...

Unit Test Case 2.1: Ask the question of the last trivia in the array.
The tester enters a correct answer.
Question: What is the best-selling video game of all time? (Hint:
Call of Duty or Wii Sports)?
Answer: Wii Sports
Case 2.1 passed...

Unit Test Case 2.2: Ask the question of the last trivia in the array.
The tester enters an incorrect answer.
Question: What is the best-selling video game of all time? (Hint:
Call of Duty or Wii Sports)?
Answer: Call of Duty
Case 2.2 passed...

Unit Test Case 3.1: Ask any question in the trivia array. The tester
enters a correct answer.
Add your question here
Add your answer here
Case 3.1 passed...

Unit Test Case 3.2: Ask any question in the trivia array. The tester
enters an incorrect answer.
Add your question here
Add your answer here
Case 3.2 passed...

Unit Test Case 4: Add your description here.
Add your information here
Case 4 passed...

*** End of the Debugging Version ***
```

Fig. 1. Sample user interface for the debugging version.

You must provide the following user interface for the **product version**. The user input is depicted as **Bold**, but you do not need to display user input in bold. Please replace "Xiao Qin" with your name. In this version, your program must run the test driver you build in Step 6.

```
*** Welcome to Xiao Qin's trivia quiz game ***
Enter a question: enter your first question here.
Enter an answer: enter your first answer here.
Enter award points: enter your first award points here.
Continue? (Yes/No): Yes
Enter a question: enter your second question here.
Enter an answer: enter your second answer here.
Enter award points: enter your second award points here.
Continue? (Yes/No): No

Question: How long was the shortest war on record? (Hint: how many
minutes)
Answer: 38
Your answer is correct. You receive 100 points.
Your total points: 100

Question: What was Bank of America's original name? (Hint: Bank of
Italy or Bank of Germany)?
Answer: Bank of Germany
Your answer is wrong. The correct answer is: Bank of Italy
Your Total points: 100

Question: What is the best-selling video game of all time? (Hint:
Call of Duty or Wii Sports)?
Answer: Wii Sports
Your answer is correct. You receive 120 points.
Your total points: 120


...
Display more questions/answers and information here...
...

*** Thank you for playing the trivia quiz game. Goodbye! ***
```

Fig. 2. Sample user interface for the product version.

How to Create Two Versions?

You can use the preprocessor directive #ifdef to create and maintain two versions (i.e., a debugging version and a product version) in your program. If you have the sequence

```
#ifdef UNIT_TESTING
add your unit testing code here
#else
add your code for the product version here
#endif
```

in your program, the code that is compiled depends on whether a preprocessor macro

by that name is defined or not. For example, if there has been a "#define UNIT_TESTING" macro line), then `` add your unit testing code here '' is compiled and `` add your code for the product version here '' is ignored. If the macro is not defined, `` add your code for the product version here '' is compiled and `` add your unit testing code here '' is ignored.

These macros look a lot like if statements, but macros behave completely differently. More specifically, an if statement decides which statements of your program must be executed at run time; #ifdef controls which lines of code in your program are actually compiled.

Unit Testing:

Unit testing is a way of determining if an individual function or class works. You need to isolate a single function or class and test only that function or class. For each function in this homework, you need to check <u>normal cases and boundary cases.</u>

       Examples for tested values:
- string – empty string, medium length, very long
- Array – empty array, first element, last element
- Int – zero, mid-value, high-value

You must implement a unit test driver for each function implemented in your program. You may need to use assert() to develop your unit test drivers if tested results are predictable.

Integration Testing:

Integration testing (a.k.a., Integration and Testing) is the phase in software testing in which individual software modules are combined and tested as a group. You may use the sample user interface illustrated in Fig. 2 on page 4 to perform an integration testing for your program.

Requirements:
1. (1 point) Use comments to provide a heading at the top of your code containing your name, Auburn Userid, filename, and how to compile your code. Also describe any help or sources that you used (as per the syllabus).
2. (1 point) Your source code file should be named as "<username>_hw5.cpp" (for example, mine would read "xzq0001_hw5.cpp").
3. (1 point) Your program must use structures and an array of structures. (see steps 1-2)
4. (1 point) Your program must use string rather than char array. (see steps 1-2)
5. (1 point) A function that creates 3 hard-coding trivia quizzes. (see step 3)
6. (2 points) A function that creates new quiz from a keyboard. (see step 4)

7. (3 points) A function that asks a question and checks a player's answer. (see step 5)
8. (3 points) Write a test driver for function implemented in Step 5.
9. (2 points) Correctly implement the main function. (step 7).
10. (2 points) Creating two versions using conditional compilation.
11. (1 point) You must reduce the number of global variables and data
12. (1 point) Usability of your program (e.g., user interface)
13. (1 point) Readability of your source code.

Note: You will lose **at least 2 points (and up to 10 points)** if there are compilation errors or warning messages when the TA compiles your source code. You will lose points if you: do not use the specific program file name, or do not have a comment on each function in your program you hand in.

Programming Environment:
Write a short program in C++. Compile and run it using the g++ compiler on a Linux box (either in Shop 3, computer labs in Shelby, your home Linux machine, a Linux box on a virtual machine, or using an emulator like Cygwin).

Deliverables:
- Submit your source code file named as "<username>_hw5.cpp" through the Canvas system.

Late Submission Penalty:
- Twenty percent (20%) penalty per day for late submission. For example, an assignment submitted after the deadline but up to 1 day (24 hours) late can achieve a maximum of 80% of points allocated for the assignment. An assignment submitted after the deadline but up to 2 days (48 hours) late can achieve a maximum of 60% of points allocated for the assignment.
- Assignment submitted more than 3 days (72 hours) after the deadline will not be graded.

Rebuttal period:
- You will be given a period of 72 hours to read and respond to the comments and grades of your homework or project assignment. The TA may use this opportunity to address any concern and question you have. The TA also may ask for additional information from you regarding your homework or project.