# COMP 2710: Homework 5

**Due: 3/16/2018 at 11:50 pm**
Points Possible: 100

**Note: You do not need to submit hard copies.**

***Goals:***

- To learn how to use structures
- To learn how to use linked data structures (Note: no array is allowed)
- To use strings
- To learn how to create multiple versions via conditional compilation
- 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, you will write a simple trivia quiz game. Your program first allows players to create their trivia questions and answers. Multiple questions should be organized and managed using **a linked data structure**. Then, your 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 points that the player has won.

Please perform the following steps to finish this assignment.

- **Step 1:** Create a TriviaNode **structure** that contains (1) information about a single trivia question and (2) a pointer pointing to other TriviaNode. This structure must contain a **string** for the question, a **string** for the answer to the question, an integer representing points the question is worth, and **a pointer of the TriviaNode type**. Please keep in mind that a harder question should be worth more points.
- **Step 2:** Create a linked list of Trivia using the TriviaNode structure defined in Step 1.
- **Step 3:** Design and implement a function that initialize the Trivia linked list by hard-coding the following three (3) trivia questions (including answers and award points).
    - 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: Minecraft or Tetris)
    - Answer: Tetris
    - Award points: 20

- **Step 4:** Design and implement a function to create and add a new TriviaNode into the linked list. You must use operator **new** to dynamic allocate memory to a new TriviaNode. Please remember to check that a new triviaNode is successfully created.
- **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 points for that question. If the player enters the wrong answer, your program should display the correct answer.
    - Input: a linked list of TriviaNode, the number of trivia to be asked in the list.
    - Output: void or int – 0 indicates success and 1 indicate failure.
- **Step 6:** Write a test driver to perform unit testing for the function implemented in Step 5. Assume there are three trivia in your created list, you must cover at least the following cases: (see Fig. 1 on page 3 for the sample user interface.)
    - Case 1: ask 0 questions.
    - Case 2: ask 1 question (i.e., the first one) from the list.
        - Correct answer
        - Wrong answer
    - Case 3: ask 3 questions (i.e., all the questions) from the list.
        - Correct answers
        - Wrong answers
    - Case 4: ask 5 questions that exceed the number of available trivia in the linked list.
- **Step 7:** Write the main function that performs the following: (see Fig. 2 on page 4 for the sample user interface)
    - Create hard-coded trivia quizzes (i.e., questions/answers/awards). Note: just call the function implemented in Step 3.
    - Create more than 1 trivia quiz from a keyboard (Note: just call the function implemented in Step 4).
    - Write a for loop; in each iteration do the following (Note: just call the function implemented in Step 5):
        - ask the player a question,
        - input the player's answer,

        - if the player's answer matches the actual answer, then award the player the points for that question,
        - otherwise (i.e., the player enters a wrong answer) your program should display the correct answer.
    - After all questions have been asked, display the total award points the player has won.
- **Step 8:** Creating two versions using conditional compilation.
    - Version 1: simply run the test driver implemented in Step 6.
    - 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 **debug version**. The user input is in red, but you do not need to display user input in red. In this version, your program must run the test driver you build in Step 6.

```
*** This is a debug version ***
Unit Test Case 1: Ask no questions. The program should give a warning
message.
Warning – The number of trivia to be asked must equal to or larger
than 1.

Unit Test Case 2.1: Ask 1 question in the linked list. The tester
enters an incorrect answer.
Question: How long was the shortest war on record? (Hint: how many
minutes)
Answer: 15
Case 2.1 passed...

Unit Test Case 2.2: Ask 1 question in the linked list. The tester
enters a correct answer.
Question: What is the best-selling video game of all time? (Hint:
Minecraft or Tetris)
Answer: Tetris
Case 2.2 passed...

Unit Test Case 3: Ask the last trivia in the linked list.
Question: What is the best-selling video game of all time? (Hint:
Minecraft or Tetris)
Answer: Tetris

Unit Test Case 4: Ask five questions in the linked list.
Warning – There are only three trivia in the list.


*** End of the Debug Version ***
```

**Fig. 1. Sample user interface for the debug version.**

You must provide the following user interface for the **production version**. The user input is in red, but you do not need to display user input in red. Please replace "Aubie" with your name.

```
*** Welcome to Aubie'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:
Minecraft or Tetris)
Answer: Tetris
Your answer is correct. You receive 20 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 production version.**

***How to Create Two Versions?***

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

```
#ifdef UNIT_TESTING
add your unit testing code here
#else
add your code for the production 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 production version here '' is ignored. If the macro is not defined, `` add your code for the production 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.

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 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. (2.5 points) 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. (2.5 points) Your source code file should be named as "hw5.cpp"
3. (12.5 points) Your program must use structures and a linked list. (see Steps 1-2)
4. (5 points) Your program must use string rather than char array. (see Steps 1-2)
5. (5 points) A function creates 3 hard-coding trivia quizzes. (see Step 3)
6. (10 points) A function creates new quizzes from a keyboard. (see Step 4)
7. (15 points) A function asks a question and checks a player's answer. (see Step 5)
8. (15 points) Write a test driver for the function implemented in Step 5.
9. (10 points) Correctly implement the main function. (Step 7).
10. (10 points) Creating two versions using conditional compilation.
11. (5 points) You must reduce the number of global variables and data

12. (5 points) Usability of your program (e.g., user interface)
13. (2.5 points) Readability of your source code.

Note: You will lose **at least 20 points (and up to 40 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 program in C++. Compile and run it using the g++ compiler on a Linux box (either in 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 "hw5.cpp" through Canvas.

***Late Submission Penalty:***

- Ten (10) points 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 90% 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 80% 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 7 days 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.