# CSC 225

**Homework 3**

Due: Tuesday, November 22, 2022 at 11:59pm

# Instructions

|  |
| --- |
| 1. **You must work on this assignment individually.** You may ask another person to look over your project for errors, but they cannot instruct you the correct way to write. (For example: the person helping can say “The problem is in the ‘switch’ statement” or “Your logic is incorrect in the second ‘if’ statement.”) In addition, you may not use code from other sources, such as the Web or other students. 2. You will need to upload the .cpp and .h files from your solution to eLearn. |

Your task in this program will be to write a version of musical chairs, implemented using a circular, singly-linked list. The program will ask the user for the number of players (2 to 7) and then for their names. The program will select a random number of players to loop through (from a single player to three times the initial number of players). It will then display a list of the players, with arrows pointing to the current player (see below), and move through the current players the chosen number of times. The player who is chosen at this point will be eliminated. This continues until there is one player remaining. At the end of each round, the program should print the name of the eliminated player. At the end of the game, the program should print the name of the winner.

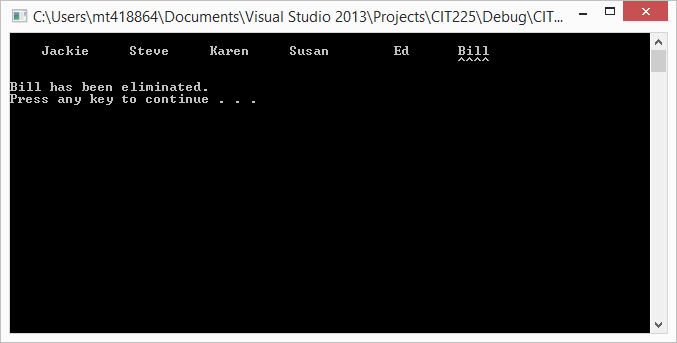
You must also create the following functions as specified and use them in your program:

|  |  |
| --- | --- |
| **Function Prototype** | **Description** |
| void addPlayer(ListNode\*&, ListNode\*&, string) | Add a player to the end of the list when passed the head, tail, and that player’s name (similar to listAppend). |
| void printPlayers(ListNode\*, ListNode\*, string) | Prints all the players in the list, given the head, tail, and the name of the player to draw arrows under. It will have two list traversals in it. |
| void playRound(ListNode\*&, ListNode\*&, long) | Plays a single round of the game. It is passed the head, tail, and the total number of players to loop through before stopping, and will delete the player that is chosen once the looping is complete. |
| void removePlayer(ListNode\*&, ListNode\*&, string) | Deletes a player from the list, given the head, tail, and the name of the player. |
| void removePlayerAfter(ListNode\*&, ListNode\*&, ListNode\*) | Deletes a player from the list, given the head, tail, and a pointer to node before the player to delete. (similar to listRemoveAfter) |
| ListNode\* listSearch(ListNode\*, ListNode\*, string); | Returns the ListNode *before* the ListNode holding the name of the player passed to it and nullptr if the name is not in the list. |

In addition to the program functioning correctly as described above, you will also be graded on:

* Using appropriate variable names, data types, and constants
  + Limit globals
  + Declare variables at the beginning of your functions
  + Use constants appropriately
  + Name constants using all capital letters
* Using comments appropriately, including header comments
* Formatting your code appropriately using tabs and blank lines
* Having user-friendly and well-formatted input and output, following the examples provided
* Appropriate data validation

Additional information:

* Make sure you understand the above requirements before you begin. Dr. Thompson can provide any clarifications you need.
* The end of a round should look as follows:  
  
* As a round progresses, the arrows beneath the names should move as the program moves through the list.
  + There should be a random delay from 100ms to 600ms between each move
  + Use the function Sleep(ms), which requires Windows.h
* The user should press a key to advance to the next round.
* Each round should start with the first player in the list.
* To clear the screen, use system(“cls”).
* You can use the length() function of a string to determine how long it is.