S C S S E

University of Wollongong CSCI114/MCS9114
Spring Session 2014: Coding Assignment 4
Due via Submit Monday 3 November 2014 at 23:59

(Individual Work – 10% of subject marks)

This assignment covers all the topics covered by assignment 1, assignment 2 and assignment 3 plus the following topics:

- **Cstrings**
- **File I/O**
- **Structures**
- **Recursion**

Remember that:
1. You must put the following information on the header of each text and source file you will be submitting in this assignment:
Student's first name:
Lab: (Day and time)
Modification Date:
Purpose of this file (or program):
2. Answers without the respective working get ZERO marks!
3. You must only use the C++ features that have already been covered in the lectures
Language: cin, cout, cerr, const, static_cast, if, else, else if, switch, while, for
Operators: +, -, /, *, %, ++, --, <, <=, >, >=, ==, !=, &&, ||, !, ?
Types: int, double, float, char, bool
Names: Valid variable names
Libraries: iostream, iomanip, cctype, cstring, fstream, cmath, cstdlib
Compiler: g++
Miscellaneous: arrays, strings, files,

Part 1. Random Entries (2 marks)

Consider a game of 2048. So far we have been inserting a 2 in the lowest available position of the game board. In reality the game insert a 2 or a 4 (weighted towards a 2) in one of the available (ie empty) spots of the game board. Modify your 2048 game (as A4Part1.cpp) so that new entries may be either a 2 or a 4 and are inserted in one of the empty gameboard spots. Provide a debug option that starts the random number generators with a set seed.

Part 2. File Output (2 marks)

A full game of 2048 can take a considerable amount of time. To allow players to (B)ookmark the current game state to a file, with the name designated by the user (`<FILENAME>.sav`), add an option to your 2048 game (as A4Part2.cpp) to save the current game . To allow players to (C)ontinue a previous add an option to load a game which prompts the player for a saved game filename (excluding extension). Both options should be read when the game normally accepts a move input.

Tasks:

Modify the basic 2048 game file from Part1 (as A4Part2.cpp) so that one of the options is to bookmark or save the current game (B). This option will.

1. Confirm the player wants to bookmark their game
2. Prompt the user for the saved game file name excluding extension

3. Open the named file for output
4. Provide a warning if the saved file name exists that data will be overwritten and ask user if they wish to overwrite or give a new file name
5. Save the current game state to the file
6. Close the file
7. Prompt user to ask if they wish to continue taking a 'y' or 'n' as an acceptable response.

Part 3. File Input **(2 marks)**

Modify the 2048 game file from Part 2 (as A4Part3.cpp) so that one of the options is load or continue a previously saved game (C). This option will.

1. Prompt the user for the saved game file name
2. Open the named file for input
3. Display the gameboard in its saved state.
4. Allow play to continue unless no more moves are available.

Part 4. Recursion **(2 marks)**

Implement the following recursive algorithm (as A4Part4.cpp) for summing two numbers.

$$for\ i \geq 0, sum(i,j) = \begin{cases} j, & i = 0 \\ sum(i-1, j+1), & x \geq 0 \end{cases}$$

Part 5. Design Decisions **(2 marks)**

In a file called A4Design.txt describe your approach to designing solutions to this assignment including (but not limited to) variable names, function names, parameters passed to functions, modularisation, library functions used, self-written functions.

There are other modifications implied in this specification, You need to identify these assumptions in your design document.

Explain and justify your design decisions.

Submission:

Use the submit program to submit your assignment as follows:

```
$ submit -u <your_login_name> -c CSCI114 -a 4 A4Part1.cpp
A4Part2.cpp A4Part3.cpp A4Part4.cpp A4Design.txt
```

Note that the above command is written as one single line in Linux.

Submission via e-mail is NOT acceptable.

NOTES:

1. **DO NOT FORGET TO LOGOUT AFTER SUBMITTING YOUR ASSIGNMENT.**

2. **SUBMIT AS EARLY AS POSSIBLE. YOU CAN RE-SUBMIT LATER IF NECESSARY. ONLY THE LATEST SUBMISSION WILL BE MARKED.**

3. **IF YOU SUBMIT YOUR ASSIGNMENT TWICE, ONE SUBMMISSION BEFORE THE DUE DATE AND ANOTHER AFTER THE DUE DATE, THEN YOU WILL BE PENALIZED FOR LATE SUBMISSON.**

4. **IF YOU DECIDE TO RESUBMIT YOUR ASSIGNMENT, ALL FILES MUST BE RESUBMITTED – NOT JUST THE ONE THAT YOU WANT TO MODIFY. FAILURE TO DO SO WILL RESULT IN HAVING ALL THE FILES THAT ARE NOT SUBMITTED UNMARKED.**

5. **YOU MUST USE THE SUBMIT COMMAND TO SUBMIT YOUR WORK.**

6. **DO NOT FORWARD THE ACKNOWLEDGEMENT FROM THE SERVER THAT YOU WILL RECEIVE AFTER SUBMITTING YOUR ASSIGNMENT, TO ANY OTHER EMAIL ACCOUNT. THIS ACKNOWLEDGEMENT WILL BE NEEDED IF YOU WANT TO QUERY ABOUT YOUR ASSIGNMENT. WITHOUT THE ACKNOWLEDGEMENT FROM THE SERVER ON YOUR UNIVERSITY EMAIL ACCOUNT, YOUR QUERY WILL NOT BE REPLIED.**