Mid-Term Individual Assignment
Connect 4

18 February 2019

This is an individual assignment and will contribute 20% of your final mark for CS1022.

You must submit your solution using Blackboard no later than **10:00 on Friday, 15th March, 2019**. Late submissions without a satisfactory explanation will receive zero marks.

You will be asked to demonstrate your solution during the scheduled lab on **Friday, 15th March, 2019**. (Your demonstration will be worth up to 10% of your mark for the assignment.) You should also expect someone grading your program to execute it. It will be assumed that programs that do not assemble without errors do not work.

Submit your `.s` assembly language source files and your report **in PDF format** as attachments to the Assignment in Blackboard. **Your `.s` assembly language source files must be suitably commented. Solutions will be checked for plagiarism.**

1 [60%] Problem Description

Connect 4 is a two-player puzzle game in which players take turns to drop discs into a vertically mounted board with seven columns and six rows. One of the players inserts yellow discs and the other inserts red discs. The objective for each player is to create a horizontal, vertical or diagonal run of four of their colour discs. Figure 1 illustrates a game in which the yellow player wins after dropping a disc into column 1 to connect a diagonal run of 4 yellow discs.

Design, write and test an ARM Assembly Language program that implements the game of Connect Four for two interactive players. Your program should use the console to prompt each player in turn to enter the column in which to insert their next disc. After each turn, your program should display the state of the board. Finally, your program should detect and announce when a player makes a winning move. An illustration of the expected console output is shown in Appendix A.

Your solution must demonstrate problem decomposition through the use of appropriate subroutines. You must document the interface for every subroutine that you write. For example, you might write subroutines to (i) innitialise the Connect4 board, (ii) make a move, (iii) check whether a move is a winning move and (iv) display the Connect4 board.

Hint!! To check whether a move is a winning move, assuming a player's disc has just landed at row r in column c, you could calculate the longest run of the same colour disc horizontally, vertically or diagonally, centred around row r, column c. If the longest run is greater than or equal to four, the player has won!

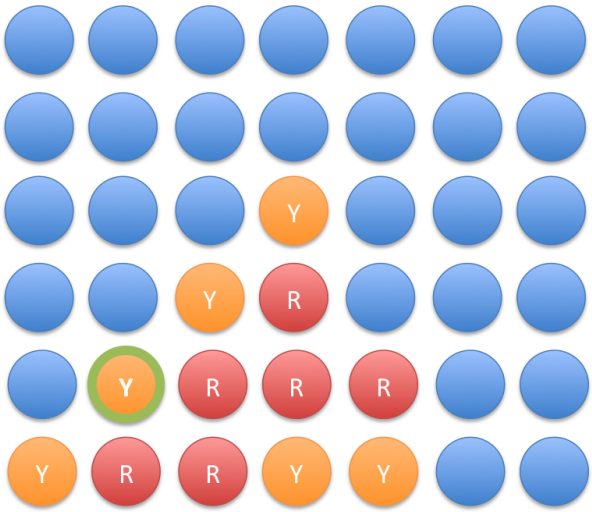

Figure 1: Connect 4

[20%] Extra Mile

Extend your Connect 4 program to allow a single player to play against the computer.

2 [20%] Documentation

Document your Connect 4 game. Your documentation should be in the form of a typed document **in PDF format**. This document must be submitted on Blackboard with your assembly language `.s` source file (or files). Documents in a format other than PDF may receive zero marks.

Your documentation should make use of examples, diagrams and pseudo-code where appropriate. **It must not take the form of a "narrative" for your assembly language program ("I moved the value from R5 into R4. Then I added R4 to R3 and stored the result in R0"). Such narratives will receive very low marks.** Instead, try to describe how your program works at a higher, conceptual level.

Your documentation must also include a detailed description of your methodology for testing each subroutine that you have written. This should include a tabulated list of the inputs used to test your subroutine and the corresponding results, along with an explanation. You should give careful consideration to choosing inputs that test different parts of your subroutines.

Note that marks will be awarded for both the content and presentation of your documentation.

A Sample Console Output

```
Let's play Connect4!!

  1 2 3 4 5 6 7
1 O O O O O O O
2 O O O O O O O
3 O O O O O O O
4 O O O O O O O
5 O O O O O O O
6 O O O O O O O

RED: choose a column for your next move (1-7, q to restart): 2

  1 2 3 4 5 6 7
1 O O O O O O O
2 O O O O O O O
3 O O O O O O O
4 O O O O O O O
5 O O O O O O O
6 O R O O O O O

YELLOW: choose a column for your next move (1-7, q to restart): 2

  1 2 3 4 5 6 7
1 O O O O O O O
2 O O O O O O O
3 O O O O O O O
4 O O O O O O O
5 O Y O O O O O
6 O R O O O O O

RED: choose a column for your next move (1-7, q to restart): 3

  1 2 3 4 5 6 7
1 O O O O O O O
2 O O O O O O O
3 O O O O O O O
4 O O O O O O O
5 O Y O O O O O
6 O R R O O O O

YELLOW: choose a column for your next move (1-7, q to restart):
```