ENGGEN 131, Semester Two, 2013
Uno project

Marking schedule

Section 0: Style (10 marks)

0.1> File name is correct (1 mark)

A single source file must be submitted. The name of this source file must be "`xxxxYYY_project2.c`", where "`xxxxYYY`" is your UPI.

0.2> All required functions are present and named correctly (1 mark)

Even if you have not completed all tasks for the project, the four required functions must be present in your source file. These functions must be named as follows:

```
void xxxxYYY_CreateDeck(Deck *deck)
void xxxxYYY_ShuffleDeck(Deck *deck, int numberOfCards)
int xxxxYYY_ValidSelection(GameState *gameInfo, int selection)
int xxxxYYY_Play(GameState *gameInfo)
```

where `xxxxYYY` is your UPI.

0.3> Any non-required functions are named correctly (1 mark)

And other functions you have written must be named as follows:

```
xxxxYYY_TheNameOfTheFunction()
```

where `xxxxYYY` is your UPI and `TheNameOfTheFunction` is the name of the function (I can't believe I am actually writing this).

0.4> The srand() function is NOT called (1 mark)

None of the functions in your source file may call `srand()`. This will be called for you in the `main()` function of the program that is used to test your functions.

0.5> Code compiles without any warning messages (2 marks)

Regardless of the system or environment in which you developed your code, it will be compiled and tested from the Command Prompt Tool using Visual Studio on Windows. It is your responsibility to have tested your code in this environment. The following command will be used to compile your submission:

```
cl /W4 xxxxYYY_project2.c do_not_submit.c
```

where `xxxxYYY` is your UPI and `do_not_submit.c` is the **unmodified** source file that was provided to you on Cecil. Your code must compile cleanly without ANY warnings at all.

0.6> Indentation is consistent throughout (1 mark)

Correct code indentation is a very important component of code readability. The indentation of your code should be consistent throughout the source file. While there are guidelines for indentation outlined on page 6 of the coursebook, minor variations on these rules are acceptable as long as the indentation is consistent - all functions, loops and conditional statements should place opening and closing braces in the same place as other functions, loops and conditionals. All code within a block should be indented by one level.

0.7> Comment describing your strategy implemented in the Play() function (2 marks)

You must write a comment at the top of your source file that clearly and precisely explains the strategy that your `Play()` function implements. This description is expected to be of high quality, using full English words, and must not include grammatical or spelling errors (feel free to use a spell checker). Moreover, the strategy that you implement must match this description (i.e. you can't describe one strategy, but have implemented another). You should aim for this comment to be a **minimum** of 150 words (there is no maximum).

0.8> Global variables are not allowed (1 mark)

You must not have any variables declared outside of function definitions in your source file (i.e. global variables). You may have new structure definitions if you wish (although this is not required), but there must not be any global variables declared.

Section 1: Task 1 (5 marks)

1.1> The deck is created correctly (3 marks)

All cards are present and initialised correctly

1.2> Code style regarding loops (2 marks)

At least one loop has been used to improve upon using individual assignment statements for initialising every element of the array.

Section 2: Task 2 (5 marks)

2.1> Random shuffling (5 marks)

The deck is shuffled randomly such that every card has a possibility of ending up in any position of the deck when the shuffle function finishes (with a probablity of 1/108 if the pseudo-random number generator was perfect).

Section 3: Task 3 (10 marks)

3.1> Selection testing (7 marks)

The `xxxxYYY_ValidSelection()` function always returns the correct result, for any input structure and any selection.

3.2> Unusual selections (3 marks)

You should also correctly handle selections that are too small (less than -1) or too large (where the selected index is greater than the number of cards in the hand) to be valid.

Section 4: Task 4 (10 marks)

4.1> No penalties generated (7 marks)

The `xxxxYYY_Play()` function always returns a valid selection, without generating any penalties during a tournament.

4.2> Better than naive (3 marks)

The strategy does not need to be highly sophisticated, however it must outperform the naive strategy of playing the first matching card or playing a random matching card.

Academic Integrity

This is an individual project. The code you submit must be your own work. Do not take a digital copy of someone else's source code, then make changes to it, and then submit it as your own work. This is unfair to your classmates and is a dishonest attempt to earn a grade for the project that is not deserved. The penalties for this are severe (failure of the course and University level disciplinary action).

Do not give anyone a digital copy of your source code (by email, facebook, USB stick or any other means). If you are concerned about this in any way, please talk to us **before** the project deadline.