Project [9]: Final Project

Project Goals

The goal of this project is to:

1. Serve as a culmination of skills learned throughout the semester.
2. Provide an opportunity to work in teams to establish goals, plan tasks, and meet objectives.

Important Notes:

1. **Formatting:** Make sure that you follow the precise recommendations for the output content and formatting. For example, do not change the text from `"Enter your coordinates from 1 to 4 "` to `"Enter coords"`. Your assignment will be auto-graded and any changes in formatting will result in a loss in the grade.
2. **Comments:** Header comments are required on all files and recommended for the rest of the program. Points will be deducted if no header comments are included.

Program

Save your program as `memory.c`

Write a program that will allow the user to play a simplified version of Memory, the Game: https://www.helpfulgames.com/subjects/brain-training/memory.html.

The program should behave as follows:

The menu is displayed which provides the user some options. The user may play a game, or the user may view scores.

If the user chooses to play the game, they should first be prompted to choose difficulty of the game, and therefore the size of the board. The user must choose a value from 1 to 3.

Then, a grid should be displayed which indicates all the "cards" that are on the board. The user should be prompted for which board locations they'd like to look at. The symbols are then revealed to the user. If they match, the user sees a message and those symbols are displayed on the board for the rest of the game. The user keeps choosing card locations until they've matched all the pairs.

The user may enter coordinates even if they've already discovered a match there. If a user enters coordinates which would be out of bounds of the board, they should receive an error message and be prompted to enter new coordinates.

When they win, the user should be presented with their score and an option to save their score. If they choose to save it, they're prompted to enter their name. Their score is then saved in order from highest to lowest and written to a file. At most 10 top scores will be saved in the file.

If the user chooses to view scores, all the players and their associated scores are displayed to the screen.

Download the provided executable to get a better idea of how the game is played.

Requirements:

All game information must be saved to a file, named `scores.txt`. This will include the top 10 players' names and their corresponding scores.

The board should be loaded with a random assortment of paired symbols. The available symbols are those between the '!' and '@' symbol, inclusive.

You may use the stdio.h, string.h, time.h, and stdlib.h libraries.

Scoring is calculated by taking the number of cards on the board and subtracting the number of moves the player made. Each time a player "flips" over 2 cards, that counts as a move unless they get a match on that turn.

A design presentation must be created to indicate the algorithmic approach to each problem, which functions are going to be used in the program, and the division of labor among teammates. This will also serve as a contract for individual team member's responsibilities. An outline of the presentation requirements will be provided on WebCampus.

EXTRA CREDIT: Instead of entering two coordinates at once, enter one coordinate at a time for a turn. The board would have to display each time a coordinate is entered.

Grading Rubric

Grading for the presentation will be done for the program as follows:

	Excellent (2)	Effective (1)	Missing (0)
Design Communication			
Design Correctness			
Responsibility Division			
Responsibility Justification			
Future Problems Anticipation			

Grading will be done for the program as follows:

Team feedback survey (on cooperation and contribution)	10%
Program compiles and runs	10%
Correctly declaring variables (and arrays)	10%
Correct decision making	20%
Correctly reading data from file	20%
Appropriate use of functions	20%
Appropriate use of pointers and references	10%
Extra Credit	10%

Submission details

ONLY ONE TEAM MEMBER NEEDS TO SUBMIT THE PROJECT.

To submit your project, you will have to use the submission script. You do this by:

1. Working on an ECC machine
2. Working on the provided VirtualBox installation

To Submit your project:

- Have a directory called "project9"
- Save your *.c files in that directory
- To submit: **(don't type the '>' symbols)**

  > cd project9
  > submit

The submission script copies all files in the current directory to our directory. You may submit as many times as you like before the deadline, we only keep the last submission.

Academic Honesty

Academic dishonesty is against university as well as the system community standards. Academic dishonesty includes, but is not limited to, the following:

Plagiarism: defined as submitting the language, ideas, thoughts or work of another as one's own; or assisting in the act of plagiarism by allowing one's work to be used in this fashion.

Cheating: defined as (1) obtaining or providing unauthorized information during an examination through verbal, visual or unauthorized use of books, notes, text and other materials; (2) obtaining or providing information concerning all or part of an examination prior to that examination; (3) taking an examination for another student, or arranging for another person to take an exam in one's place; (4) altering or changing test answers after submittal for grading, grades after grades have been awarded, or other academic records once these are official.

Cheating, plagiarism or otherwise obtaining grades under false pretenses constitute academic dishonesty according to the code of this university. Academic dishonesty will not be tolerated and penalties can include cancelling a student’s enrolment without a grade, giving an F for the course, or for the assignment. For more details, see the University of Nevada, Reno General Catalog.