Project 1: The HIGH/LOW Game

This project requires many hours of work to complete, so start on this project immediately.

YOUR NAME:______________________________ DEMO DATE:______________

DUE DATE: Wednesday, 8th February, 2012

Disclaimer: ALL games/projects assigned in EGR261 are designed to help students understand basic C programming and have some fun learning some of these techniques. The project can only be used for entertainment/educational purposes.

Objective:

Write a C program to play the 2-person high/low game. Person "A" picks a WHOLE (i.e., integer) number (called the target number) from a range of numbers starting at the number 1 up to the number 100 and does not reveal the number that was picked. (The upper range number 100 can vary, so for this project we will call this number the maxRange number.) A different person, person "B", now attempts to find out the target number by picking numbers and person "A" indicates if the number that was picked is higher than the target number, lower than the target number or is the target number. Your assignment is to play this game against the computer. REMEMBER, the computer can be person A or B.

For a grade of "C", the following is required:

0. Create a C program with proper documentation and indentation
1. Print out a description of the game.
2. Ask the user if he/she wants to play or exit the game.
3. Start Round:
    a. Computer is person A ("thinks" of a number):
        i. Have the computer ask the user what the maxRange number is.
        ii. Computer randomly selects a number between 1 and maxRange but does not inform the user.
        iii. The user now attempts to find the target number, and the computer indicates whether the number the user picked is high or low. This is an interactive part of the game. That is, the user guesses a number and the computer states "High" or "Low" until the target if found.
        iv. Your program should warn the user for all inconsistencies, for example, the user picks a negative number, or the user picks a number higher than the maxRange number and so on.
        v. Count the number of guesses it takes the user to find the number and display this value.
    b. Computer is person B (guesser):
        i. Have the computer ask the user what the maxRange number is.
        ii. The user picks a target number in their head.
        iii. The computer now attempts to find the target number, and the user indicates whether the number the user picked is high or low. This is an interactive part of the game. That is, the computer guesses a number and the user states "High" or "Low" until the target if found.
        iv. Your program should warn the user for all inconsistencies. (As you play the game with the computer, think about how you can trick the computer, i.e., if you are lying, can the computer determine you are lying.)
        v. Count the number of guesses it takes the computer to find the number.
    c. The winner of the round is the one with the fewest guesses.
4. Ask the user if he/she wants to continue to play or exit the game.
5. If the user chooses to continue, increment the round number and go to step 2. If the user elects to exit the game display a final winner (highest number of rounds won).

For a grade of "B", you must do the above and add the following:

- In a file named "HighLow.dat", track the historic game statistics. Games won by user and computer, **maximum** number of guesses it took the computer to find the user's target number (ComputerMax) and the maximum number of guesses it took the user to guess the computer's target number (UserMax). When the games starts (step 1), display these numbers.

For a grade of "A", you must do the above and add the following:
NO points will be given to this section unless you complete all of sections B and C.

- Change step 4 to the following: "Ask the user if he/she wants to continue to play, exit the game, or save the current game." If the user elects to save the game, then save ALL aspects of the game, round, total points for the players, etc.
- Change step 2 to ask the user if they want to load a previous game. If so, the game should start exactly where it left off, with all values the same. If not, then start the round at 1 and begin a new game.

Project Tips:
The look and feel of the game is up to you (aside from the functional requirements stated above). Have stopping points in the game so the display can be read without scrolling backwards. I suggest you have a friend play your game and give you some feedback on your game. This really is an excellent way to get suggestions for improvement before the instructor plays the game. Don't wait until the demonstration of your program to ask questions. Start early and clear up any confusion or questions before that point.

Turn in Requirements (in this order) in lab:

- Hard copy of your source code (output not required) and be ready to demo in lab. Attach a printout of this document as a cover sheet.

Your program will be demonstrated in lab. IMPORTANT: I will have about 5 minutes per student to check out your program. When your name is called for demo, you must be ready to go.

Tentative Rubric:

Basic operations of the game, flow of the game	50 pts
Displays wins for each player	10 pts.
Computer guessing (you will lose points if the computer guesses: 1 then 2 then 3 then 4 and so on)	10 pts.
Section B	10 pts
Section A	10 pts
Good design code (module, functions) Format, comments, etc	10 pts