CPS 125 − Project 5 Name: _____ Fall 2013

DUE: in class Tuesday, December 3rd - a detailed, easy to read flowchart for the following program

DUE: Sunday, December 8th before 11:59pm

Project 5: Write a number guessing game program that

1. asks for a natural number between 1 and 10 (inclusive) that is set to the variable `numDig`.
2. picks a random integer with `numDig` digits using $0, 1, 2, \ldots, 9$ with no repeats that is set to the variable `answer`.
3. asks for a difficulty: easy, medium, or hard.
   (a) If the user picks difficulty easy, then there is no limit to the number of guesses.
   (b) If the user picks difficulty medium, then set the variable `numGuesses` to `numDig` times 7.
   (c) If the user picks difficulty hard, then set the variable `numGuesses` to the minimum of `numDig` times 4 or the factorial of `numDig+1`.
4. asks for a guess, `guess`, consisting of `numDig` digits using $0, 1, 2, \ldots, 9$ with no repeats
   (a) If the answer is guessed in `numGuesses` or less attempts, the program announces that the correct number was picked and terminates.
   (b) If the current guess is incorrect and guesses remain, the program prints how many of the digits in `guess` are digits that appear in `answer` and how many digits in `guess` are in the current location. These two computations should be coded as two functions separate from `main()`.
   (c) If the answer is not guessed within `numGuesses` attempts, the program announces that the maximum number of attempts has been reached and reveals the correct number.
   (d) If the user types "quit" when prompted for a guess, the program prints the answer and terminates.
   (e) If the user types "history" when prompted for a guess, the program prints a table of previous guesses with the digit/location counts. This should be done by reading the file "p5history.txt" that contains the information needed.

When your project is complete, be sure you have followed the following guidelines:

- Comment your code. There should be a comment at the top with your name and a description of the program as well as comments throughout the code as needed.
- Any time your code asks for input, the program should verify that the input is valid. If the user enters something unexpected, the program should repeat the request for an input.
- Re-read the above directions to be sure your program follows them.
- Your program should mimic the output given by my program on the math server exactly.
- Your code must compile – check it carefully. When finished, name your final version of the source file "<your last name>.c". Send me your .c file as an attachment to an email with subject "CPS 125 - Project 5".