Assignment #6

Introduction to C Programming – COP 3223

Objectives

1. To learn how to use arrays to store and retrieve data to help solving problems.
2. Reinforce use of input files.

Introduction: Mission to Mars

Your friend has been playing a new Mars Colony simulator nonstop! They are always talking about how cool it would be if they could be a on the first real-life mission to Mars! To amuse your friend, you have decided to create a series of programs about the possible first colony on Mars.

Problem: Malfunction! (marsmalfunction.c)

Our shuttle of autonomous robots has landed on Mars! There is a slight problem: they cannot leave the shuttle! There has been a malfunction in the automatic deployment software and we will need to send an access code from Earth.

For security, this access code is not known ahead of time and instead must be derived by our mission programmers. There are seven numbers in the access code and we will need to try different combinations to open our shuttle remotely. We do know that each number in the access code in between 1 and 100, inclusive.

Our automatic deployment software cannot tell us what the correct access code is, but it can tell us if any of the numbers we've entered are correct. Each number in the access code is unique and the software can also remind us if we enter any duplicate values.

Your program will need to read in a file of the correct access code, which contains seven number between 1 and 100, inclusive. Then, prompt the user for the seven numbers indicating the seven values in the access code they wish to try. They cannot use the same value twice in one attempt. If they have all the correct values in the exactly correct order, they can open the door to the shuttle. If they have some correct numbers, regardless of order, let the user know how many numbers are correct. If none of the values are correct, state that none of those numbers will work.

Input Specification

1. The file name will be a string less than 50 characters in length
2. The user will enter integers one at a time (pressing enter after each integer) for their guess
3. The user will enter integers that are between 1 and 100, inclusive.

Input File Format

The input file will contain 7 unique integers from 1 to 100.

Program Specification
You must use arrays to solve the problem.

Your program should first prompt the user for the name of the input file. Then, your program should process the input file and copy the correct access code into the program. Then you can prompt the user for their first guess. If the user attempts to use a number more than once in a single guess, tell them they can only use each number once. If the user exactly matches the numbers and the order, let them open the shuttle door. If the user identifies some of the correct numbers, let them know how many values are correct but tell them they may not be in the right order. If none of the valuess are correct, tell the user that none of those numbers will work.

Output Specification
Output to the screen. Request the name of the file from the user and prompt them for the first access code:

> The shuttle door requires an access code of 7 numbers. Each number will be between 1 and 100, inclusive. What is the access code?

For each guess the user makes, respond with one of the following phrases:

> Each number can only be used once.
>
> None of those numbers are present in the access code.
>
> X of those values are correct, but may not be in the correct order.
>
> > where X is a value from 1 to 7.
> > If 0 values are correct, a different output should be printed.
>
> Success! Your access code has opened the door to the shuttle.

For subsequent guess, prompt the user again for the access code:

> What is the access code?

Input/Output Sample(s)
Sample input and output files will be provided on the webcourse.

Acceptable Resources
Remember, the use of online help sites is strictly prohibited. The only acceptable resources for these assignments are below:

- Course Webcourse
  - In particular: Week 7 – Files, Week 8 – Arrays
- Course Textbook
  - Programming Knights: An Introduction to Programming in Python and C by Arup Guha
- Professor Guha's Course Archive
  - http://www.cs.ucf.edu/~dmarino/ucf/transparency/cop3223/

- Course TAs and Instructor Office Hours
    - Getting Help: https://webcourses.ucf.edu/courses/1336411/pages/getting-help

Style Notes

Please review the course Style Guide on the webcourse, with special attention to the following notes:

- comment major sections of code addressing: "What does this block do?" and "Why did I implement this block in this way?"
- place comments above the line(s) to which it applies
- use inline comments (//) and leave one space between // and the comment's first character
- All variables should be declared at the top of your functions (in this program, only main is needed) and should have meaningful names
- Be sure to declare `main` with: int main(void) {
- Indent the contents of main four spaces or one tab
- leave a space on both sides of any binary operators you use in your code (i.e., operators that take two operands). For example, use (a + b) - c instead of (a+b)-c.
- keywords if, while, and for should have a single space after them
- contents of if statements and loops should be indented four spaces or one tab
- conditions should not have any space immediately after each `(` or immediately before each `)`.

Deliverables

One source file: *marsmalfunction.c* for your solution to the given problem submitted over WebCourses.

Restrictions

Although you may use other compilers, your program must compile and run using Code::Blocks. Your program should include a header comment with the following information: your name, course number, section number, assignment title, and date. Also, make sure you include comments throughout your code describing the major steps in solving the problem.

Grading Details

Your programs will be graded upon the following criteria:

1) Your correctness

2) Your programming style and use of white space. Even if you have a plan and your program works perfectly, if your programming style is poor or your use of white space is poor, you could get 10% or 15% deducted from your grade.

3) Compatibility – You must submit C source files that can be compiled and executed in a standard C Development Environment. If your program does not compile, you will get a sizable deduction from your grade.