CSE 231 Spring 2008

Programming Project #4

Assignment Overview
This project focuses on programming with loops and conditionals, as well as our first project working extensively with strings. It is worth 30 points (3% of your overall grade). It is due Monday, February 4th before midnight.

The Problem
We are going to play the classic game Mastermind. If you don't know the rules, you can look at:
http://www.cut-the-knot.org/ctk/Mastermind.shtml

We are going to play the game with a slight variation, using numbers instead of colors. Thus each guess will be a 4 digit number, and the **key** (the number being decoded) will be a 4 digit number. We will also stipulate that no number will repeat in the key. Thus 1234 is a legal key but 1123 is not legal and should not be used in our game. (To play the game on the web site then, you need 10 colors and 4 pegs.)

In our game, you provide the key to be guessed at the start of the program, and then your program allows a player to try and guess the entered key. The program does not generate a random key so you can more easily test your program. Your program does not try to play mastermind, it simply enforces the rules and provides feedback to the player.

On each guess, your program provides two numbers. How many of the 4 numbers in the guess are exactly correct (the correct number in the correct position of the key) and how many of the guess numbers are in the key but not in the correct position.

The player gets twelve guesses and if they do not get the key in 12 guesses, they lose.

Program Specifications
Your program will play the mastermind game as follows:

1. Prompt for the 4 digit key we will play with. No error checking here.
2. Prompt the user for a guess.
    a. we check to see if the guess if exactly length 4
    b. we check to see if the guess contains all numbers
    c. we check to see if the guess has no repeats

   any violation of these rules prints an error message and prompts for a guess again. **Bad input does not count as a guess**.
3. End the game if:
    a. the guess is correct. Print out a winner message
    b. it was the last guess (the twelth) and the guess is incorrect. Print out a loser message and the value of the key
4. If no end condition occurs, print out how many of the guess numbers are exactly right (correct number in the correct position) and how many of the guess numbers are correct but not in the correct position. Example. The key is 1234 and the guess is 4256. The feedback would be 1 number exactly correct (2) and 1 number in the wrong position (4). That is, each digit in the guess produces at most one result (exact or wrongPosition).
5. You should keep a history of guesses so that the user can see how they are progressing.

6. Continue the game until an end condition occurs

As an example, two programs are provided in the project04 directory. One is called test.py and the other master.pyc. The pyc file is a compiled solution to the problem. You can't read it, but you can run it using the test.py file. Move both files to your directory and then double click the test.py file. It should play the game for you. **Remember**, move both files and place them in the same location!

Deliverables

proj04.py -- your source code solution (remember to include your section, the date, project number and comments).

1. Please be sure to use the specified file name, i.e. "proj02.py"
2. Save a copy of your file in your CSE account disk space (H drive on CSE computers).
3. You will electronically submit a copy of the file using the "handin" program: http://www.cse.msu.edu/handin/webclient

Assignment Notes:

We are working with strings here, so here is a reminder of some functions that would be useful. Look at the online Python manual about strings for more details. Assume we have a string variable `myString = 'abcde'`

1. `len(myString)` returns the length in characters of the string. In this case it is 5
2. `myString.isdigit()` returns a boolean indicating whether the string contains only numbers. In this case it returns False.
3. `'a' in myString` returns a boolean indicating whether a string (in this case 'a') is part of the string. In this case, it returns True

4.
   ```
   for member in myString:
       print member
   ```
   This loop goes through each individual character, one at a time, and sets `member` (a variable name, you can choose it to be something else) to that character. In this case,it prints out the characters in myString, one line at a time.
5. `str(1)` converts an integer to a string

Getting Started

1. Do the normal startup stuff (create proj04.py, back it up to your H: drive, etc)
2. Write a high-level outline of what you want to do. Think before you code
3. Start to break the problem down. How you do that is up to you, but here are some suggestions.
   a. prompt for a key, and then loop through and collect guesses. No error checking for evaluation yet
   b. add one error checking routine (there are three)
   c. add one evaluation check (there are two)
   d. check for winners and losers

Sample Interaction

Here, **exist** are those numbers that are in the key but not in the right position, **position** are those numbers correct and in the right position.

```
>>> ================================ RESTART ==========
>>> 
What is the key:1234

Guess:4321
4321: exist:4, position:0

Guess:5678
4321: exist:4, position:0
5678: exist:0, position:0

Guess:1243
4321: exist:4, position:0
5678: exist:0, position:0
1243: exist:2, position:2

Guess:123
****guess must have lenght of 4, try again
4321: exist:4, position:0
5678: exist:0, position:0
1243: exist:2, position:2

Guess:12345
****guess must have lenght of 4, try again
4321: exist:4, position:0
5678: exist:0, position:0
1243: exist:2, position:2

Guess:1b34
****contains non-numbers, try again
4321: exist:4, position:0
5678: exist:0, position:0
1243: exist:2, position:2

Guess:1123
****no repeated digits, try again
4321: exist:4, position:0
5678: exist:0, position:0
1243: exist:2, position:2

Guess:1234
4321: exist:4, position:0
5678: exist:0, position:0
1243: exist:2, position:2

You guessed the key: 1234
It took you 4 guesses
>>>
```