(100 points) Programming Assignment 1 – Bulls and Cows game Service (Group Programming – groups of 1-2; up to 1 partner)

1. Grading

--- 70% of the grade depends on your program's correctness. Brute force solutions and extremely inefficient programs that produce correct results may lose credit.
--- 30% of the grade depends on your program's readability. Programs should conform to the programming style emphasized throughout the semester. Meaningful variable names and a docstring for the main program and every function are REQUIRED.

Make sure to follow the instructions precisely! Understanding and delivering programs according to the program specifications are the keys to good programming.

Submit one program file named program1.py. 2 pts will be deducted for each program file that's not named program1.py.

Add comments and docstrings for the program and functions with the description of the functions/program, any variables and parameters with their types, and any return values and their types.

Make sure all the names in your group are in the first line of your submitted file (-3pts if any partner is not indicated).

70% of the grade depends on your program's correctness. 30% of the grade depends on your program's readability.

2. Objectives

_ First develop a general outline of how your program will be structured. This outline is often called Pseudocode showing the list of steps to take for solving the given problem. The outline is not in Python. In this process, you should identify functions that are to be used, and any parameters that go with each function.
_ Develop a Python program based on the designed outline.
_ Correct the program for any syntax errors.
_ Test the program EXTENSIVELY for semantic (i.e., logic) errors. When appropriate, use assertEquals to test your functions.

3. Assignment

3.1 Overview

You are to write a program that provides a free "Bulls and Cows" game service.

See https://en.wikipedia.org/wiki/Bulls_and_Cows for the game definition and rules.

This game is typically played with 4 digits with 2 players. However, you will be writing the program with only 1 player. The player will try to guess the number based on the "bulls" and "cows" from his/her guesses.

3.2 Program Design

Write a program in a file named *program1.py.* Your program design should make extensive use of functions. Here is an outline of what the main program should do.

1. Display a welcome message and general instructions to the user. (Hint: call a function that outputs the welcome and instructions.)

2. Ask the player whether he/she wishes to play or exit.

3. If exit is entered, end the game. If not, start a new game.

4. Randomly generate an integer with 4 unique digits. Each digit must be in the range of 1 to 9.

5. Ask the player to enter his/her guess.
   Verify the player enters 4 unique digits. If not, gracefully request the player to reenter a valid number. (Note: when an assignment says "Verify …", that implies your program must perform input testing, providing a meaningful error message and, if appropriate, prompting the player to repeat his/her action.). <u>You must use a while loop to continue to ask the user for valid input.</u>
   You can assume the user will always enter the input in a positive integer format or 'exit' to end the game.

6. Calculate and display the results of the number of bulls and cows.

7. Continue the game until the correct guess is made or the player wishes to quit. <u>You must use a while loop.</u>

8. When the correct guess is made, display a congratulatory message. If the player "quits" the game prematurely, then simply display the number and end the game.

9. Ask the player whether he/she wishes to continue or exit. Start a new game by repeating from step 4 or exit the game. <u>You must use a while loop to repeat the game.</u>

You should modularize your program by writing functions and dividing large functions into smaller functions. You can have as many levels of sub-modules as you need.

No assertEqual tests required for this program.

<u>What to submit:</u>

1) Submit one program file named program1.py. 2 pts will be deducted for each file that's not named program1.py.
2) Submit a shell script that must be named *program1shellscript.py* showing a session where you executed your *program1.py* (-5 pts if the script file is not submitted)

After you are ABSOLUTELY sure your program works, start a new Python shell with IDLE, and open your *program1.py.* Run the program. In the shell window, exercise all of the functionality of your program. That is, sometimes have the user (i.e., you) enter invalid inputs, sometimes valid inputs, choose different options. After fully exercising your program, save the *program1shellscript.py* shell script session.

3) Make sure to save the submission confirmation email from sakai as the proof that you have submitted the assignment. If you have any issue submitting, you are responsible to report immediately to your TA and the instructor, and send an email with program1.py and program1shellscript.py to your TA and the instructor to prove that you had the assignment ready by the due date.

Sample Run:

See program1shellscript.py in the program 1 folder in sakai resources.