Introduction to Python Individual Assignments

Download the files *guess_my_number_starter.py* and *khansole_academy_starter.py*. Complete the programs in each file based on the instructions in the problem statement. Upload the file on your GitHub profile and share the link to the GitHub project.

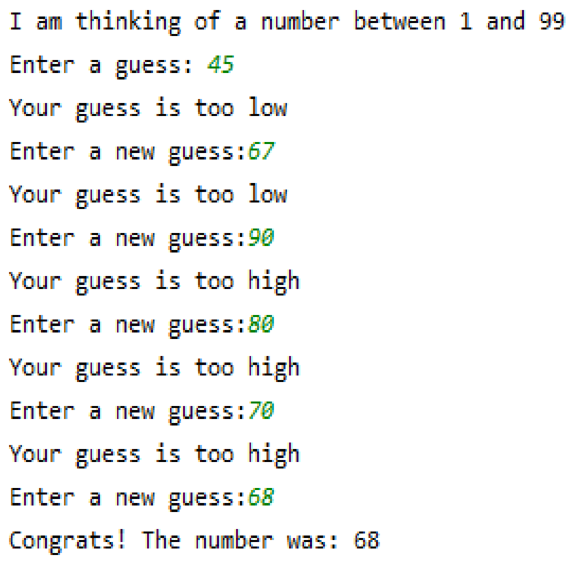
1. Guess My Number

In this problem, you will write a program in the file *guess_my_number_starter.py* that randomly generates a random number which is not shown to the user (some sort of secret number). The random number should specifically be an integer between 1 and 99.

The user is then asked to guess a number within the given range (1 to 99). If the user's guess is higher than the one guessed by the program, the user should be prompted to try again as their guess is too high. If the user's guess is lower than the one guessed by the program, the user should be prompted to guess again as their number is too low.

The process should be repeated until the user guesses the number correct.

A sample run of the program is shown below (user input is in blue font).



```
I am thinking of a number between 1 and 99
Enter a guess: 45
Your guess is too low
Enter a new guess:67
Your guess is too low
Enter a new guess:90
Your guess is too high
Enter a new guess:80
Your guess is too high
Enter a new guess:70
Your guess is too high
Enter a new guess:68
Congrats! The number was: 68
```

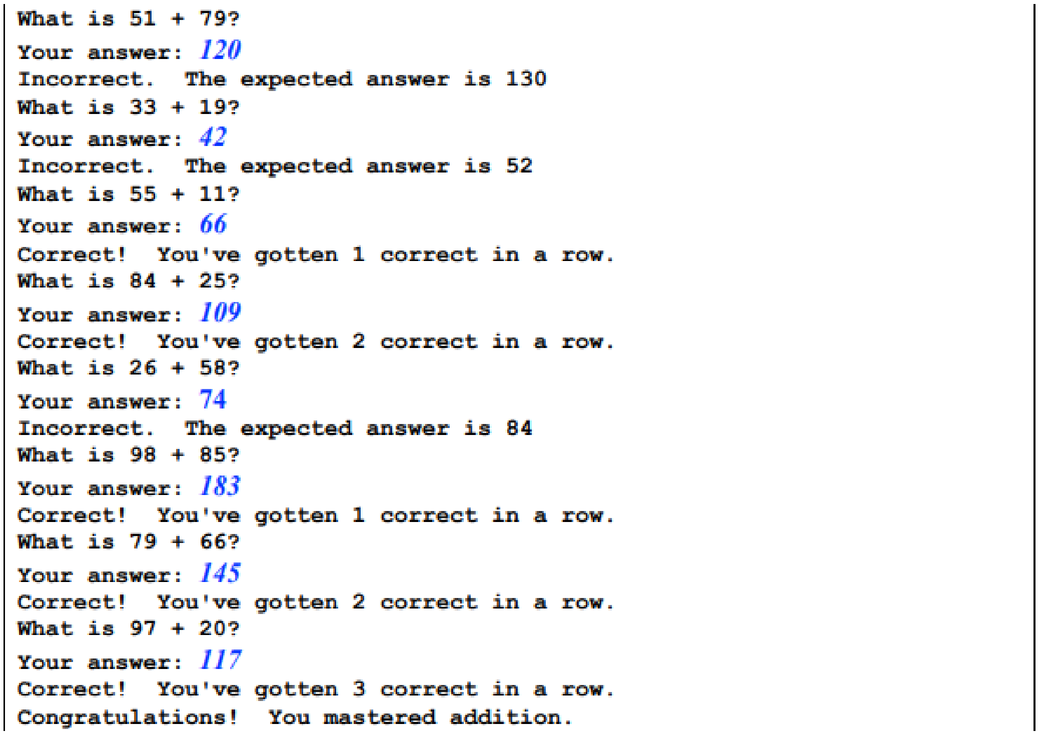
2. Khan-Sole Academy

In this problem, you will write a program in the file *khansole_academy_starter.py* that randomly generates simple addition problems for the user, reads in the answer from the user, and then checks to see if they got it right or wrong, until the user appears to have mastered the material.

More specifically, your program should be able to generate simple addition problems that involve adding two 2-digit integers (i.e., the numbers 10 through 99). The user should be asked for an answer to each problem. Your program should determine if the answer was correct or not and give the user an appropriate message to let them know.

Your program should keep giving the user problems until the user has gotten 3 problems correct in a row. (Note: the number of problems the user needs to get correctly in a row to complete the program is just one example of a good place to specify a constant in your program).

A sample run of the program is shown below (user input is in blue font).



```
What is 51 + 79?
Your answer: 120
Incorrect.  The expected answer is 130
What is 33 + 19?
Your answer: 42
Incorrect.  The expected answer is 52
What is 55 + 11?
Your answer: 66
Correct!  You've gotten 1 correct in a row.
What is 84 + 25?
Your answer: 109
Correct!  You've gotten 2 correct in a row.
What is 26 + 58?
Your answer: 74
Incorrect.  The expected answer is 84
What is 98 + 85?
Your answer: 183
Correct!  You've gotten 1 correct in a row.
What is 79 + 66?
Your answer: 145
Correct!  You've gotten 2 correct in a row.
What is 97 + 20?
Your answer: 117
Correct!  You've gotten 3 correct in a row.
Congratulations!  You mastered addition.
```