Homework 2: Pig

(Deadline as per Coursera)

This homework deals with functions and modular programming.

General Idea of the Assignment

Pig is a very simple game. Two players take turns; on each turn, a player rolls a six-sided die ("die" is the singular of "dice") as many times as she wishes, or until she rolls a 6. Each number she rolls, except a 6, is added to her score this turn; but if she rolls a 6, her score for this turn is zero, and her turn ends. At the end of each turn, the score for that turn is added to the player's total score. The first player to reach or exceed 50 wins.

For example:

- Alice rolls 3, 5, 3, 1, and stops. Her score is now 12.
- Bob rolls 5, 4, 1, 1, 2, and stops. His score is now 13.
- Alice rolls 5, 3, 3, 5, 4, and stops. Her score is now 32 (12 + 20).
- Bob rolls 4, 6. He has to stop, and his score is still 13 (13 + 0).
- etc.

As defined above, the first player has an advantage. To make the game more fair, we will say that if the first player reaches or exceeds 50, the second player gets one additional turn. (If the second player is the first to reach 50, the first player does not get an additional turn.)

Your assignment is to implement the game of Pig. You will play against the computer. The computer always goes first, so you get one more turn if the computer is the first to reach 50. If both players are tied with 50 or more, each gets another turn until the tie is broken.

Each player must roll the die at least once per turn.

Additionally, the program should ask if the user wants to play again. Any response that begins with ‘y’ (capital or lowercase) should play again. Any response that begins with ‘n’ (capital or lowercase) means the user wants to exit. Any response that begins with any other character should ask the user again.

Define and use at least the following functions:

- *def print_instructions():*
  - Tell the user the rules of the game. What words you use is up to you.
- *def computer_move(computer_score, human_score):*
  - The computer rolls some number of times, displays the result of each roll, and the function returns the result (either 0 or the total of the rolls). The function should use its parameters in order to play more intelligently (for example, it may wish to gamble more aggressively if it is behind).

- *def human_move(computer_score, human_score):*
    - Repeatedly asks whether the user wants to roll again and displays the result of each roll.
        - If the user chooses to roll again, and DOES NOT roll a 6, this function adds the roll to the total of the rolls made during this move.
        - If the user chooses to roll again, and DOES roll a 6, the function returns 0.
        - If the user chooses not to roll again, the function returns the total of the rolls made during this move.
- *def is_game_over(computer_score, human_score):*
    - Returns True if either player has 50 or more, and the players are not tied, otherwise it returns False. (Hint: Call this only after the human's move.)
- *def roll():*
    - Returns a random number in the range 1 to 6, inclusive. (Hint: Find the random module on https://docs.python.org/3/library/index.html and follow the link to find the *randint* method.)
- *def ask_yes_or_no(prompt):*
    - Prints the prompt as a question to the user, for example, "Roll again? ". If the user responds with a string whose first character is 'y' or 'Y', the function returns True. If the user responds with a string whose first character is 'n' or 'N', the function returns False. Any other response will cause the question to be repeated until the user provides an acceptable response.
- *def show_current_status(computer_score, human_score):*
    - Prints the user's current score and the computer's current score, how far behind (or ahead) the user is, or if there is a tie. (Hint: Call this before and after the human's move.)
- *def show_final_results(computer_score, human_score):*
    - Tells whether the human won or lost, and by how much. (Hint: Call this when the game has ended.)
- *def main():*
    - This is where your program will start execution.

Add starter code to the bottom of your script to run the main function.

```
if __name__ == '__main__':
    main()
```

Submission

Your submission should include:

- *pig.py* - the source code for your game
- This file must include a header, in the form of a multi-line comment at the top of your script, that contains (each on its own line):
    - Your name
    - Your PennID

- Statement of work. Either:
  - A list of resources you used and/or people you received help from (including TAs/Instructor)
  - A statement that you worked alone without help

Evaluation

Correctness - 10 pts

- Does the game work as expected? Did you follow the directions exactly? Is your math correct? Is each player forced to roll the die at least once per turn? Do you get one more turn if the computer is the first to reach 50?

Code reuse – 5 pts

- Are there common pieces of code that you can separate out into "helper" functions instead of having them copied in multiple (two or more) places? If there are, we want you to create these helper functions.

Docstrings/Comments - 5 pts

- Does every function have a docstring defined? Does it appropriately describe what the function does? Are all non trivial lines of code commented?

Style – 5 pts

- Style includes things like variable names, comments, function names, and general readability of your code. Are variables named based on the information they store? Are your user-defined functions named based on what they do?