ISTA 130: Spring 2020
Programming Assignment 5
`pig.py`: `while` Loops and `if` Statements

Due: Thursday, April 2nd, by 11:59 pm
(submit via D2L Assignments)

Please read the instructions below carefully and follow them closely. All problems (and parts of problems) are required except as noted below.

For any functions you are asked to write, you must use the exact function names given in the descriptions, and you must have parameters in the order shown in the description.

To people who know what default values and/or keyword arguments are: do not use default values and do not use keyword arguments. If you don't know what these are it is unlikely you'll do it by accident. Ask us if you're not sure.

Preparation

We have seen the `random` module in action, but before you begin working on your program you may want to practice a bit. In a Python shell:

`import` the `random` module and do the following:

```
>>> help(random.randint) # read the documentation
>>> random.randint(0,1)  # repeat this until you know what it does
>>> random.randint(0,3)  # continue experimenting
>>> random.randint(4,6)
>>> random.randint(1,100)
```

Now let's meet `random.seed`:

```
>>> random.seed(100)
```

Next, 5 times in a row run this line: `random.randint(0, 10)`
Each time write down the random number.
Exit the Python shell, open a new shell, and enter: `random.seed(100)`
5 times in a row run this line: `random.randint(0, 10)`
Each time write down the random number. Compare to the numbers you already wrote down.

Random numbers are generated by a complicated mathematical process. By setting a seed value, we can ensure that the exact same sequence of "random" numbers will be generated. This is very useful for testing code.

Dice Game: Pig (100 points)

For this program you will build a simple dice game called Pig. In this version of Pig, two players alternate turns. Players each begin the game with a score of 0. During a turn a player will roll a six-sided die one or more times, summing up the resulting rolls. At the end of the player's turn the sum for that turn is added to the player's total game score.

If at any time a player rolls a 1, the player's turn immediately ends and he/she earns 0 points for that turn (i.e. nothing is added to the player's total game score). This is called "pig". After every roll that isn't a 1, the player may choose to either end the turn, adding the sum from the current turn to his/her total game score, or roll again in an attempt to increase the sum.

The first player to 50 points wins the game.

Details

Open a text editor and create a new file called `pig.py`. In the file:

1. Write a function called `print_scores` that has four parameters – these hold, in this order, the name of the first player (a string), his/her score (an int), the second player's name (a string), and score (an int). The function will:
   a) Print the following message, using the current parameter values for the player names and scores (NOT necessarily Ziggy, 18, Elmer, and 23):

      ```
      --- SCORES Ziggy: 18  Elmer: 23 ---
      ```

      - Print an empty line before this line.
      - There is a single tab between `SCORES` and the name of player 1.
      - There is a single tab between player 1's score and player 2's name.
      - Every other gap is a single space.

2. Write a function called `check_for_winner` that has two parameters: a name (a string) and a score (an `int`):
   a) If the score is 50 or more print the following message (using the current parameter value for the name, not Ziggy):

      ```
      THE WINNER IS: Ziggy!!!!!
      ```

   b) The function returns `True` if the score is 50 or more, otherwise it returns `False`.

3. Write a function called `roll_again` that has a single string parameter to hold a player's name. In the function:

   a) Enter a `while` loop which repeats the following steps:

      - Ask whether the player would like to roll again, using the following message:

        `Roll again, Ziggy? (Y/N)`

        ➢ There is a single space after `(Y/N)`.

      - If the player answers with `Y`, y, N, or n, exit from the loop.

      - Otherwise print the following message (replacing XXX with the user's entry):

        `I don't understand: "XXX". Please enter either "Y" or "N".`

   b) The function returns either `True` or `False` depending on whether the player wants to roll again.

4. Write a function called `play_turn` that has a single string parameter, which holds a player's name. The function will:

   a) Print the player's name as shown in the following example:

      `---------- Ziggy's turn ----------`

      - There are 10 hyphens on either side of the message.

   b) Initialize a variable to keep track of the points the player earns on this turn.

   c) The function then enters a `while` loop in which the player attempts to earn points. It repeats the following steps:

      - Generate a random integer between 1 and 6 inclusive using the `random` module.

      - Print a message displaying the roll as shown in the following example:

        `<<< Ziggy rolls a 4 >>>`

        ➢ Put a tab to the left of the first angle bracket.

      - If the roll is a 1:

        ➢ Print the following message:

        `!!! PIG! No points earned, sorry Ziggy !!!`

        ❖ Put a tab to the left of the first exclamation point.

        ➢ Make sure the player earns no points for the turn.

        ➢ Make a change to ensure you exit the loop or exit directly.

        ➢ Use the following message to pause the program until the user is ready to continue:

        `(enter to continue)`

      - If the roll is not a 1:

- Add the roll to the player's points earned for the turn.
- Print the player's turn total as in the following example:

  `Points: 12`

  - Put a tab to the left of the word `Points`.
- Use the `roll_again` function to ask the player if he/she would like to roll again.
- If not, make a change to ensure you exit the loop or exit directly.

d) Make sure that you returned the number of points earned by the player for this turn in all cases.

5. In `main`:

a) Get an integer to use for the `random.seed` function from the user with the prompt:

`Enter seed value:`

b) Pass `random.seed` the value retrieved from the user, (**MAKE IT AN `int`**).

c) Print two blank lines followed by the title of the game:

`Pig Dice`

d) Ask the user to enter the first player's name with this prompt:

`Enter name for player 1:`

e) Ask the user to enter the second player's name with this prompt:

`Enter name for player 2:`

f) Print a greeting message in the following form:

`Hello Ziggy and Elmer, welcome to Pig Dice!`

- Put a tab to the left of `Hello`.

g) Initialize the players' scores to 0.

h) Print their scores using the `print_scores` function.

i) Enter a `while` loop in which repeats the following steps:

- Call the `play_turn` function for player 1 and add any earned points to his/her total score.
- Print the players' scores (use the appropriate function).
- Check whether player 1 won the game (use the appropriate function).
- If player 1 didn't win, call the `play_turn` function for player 2 and add any earned points to his/her total.
  - Print the players' scores.
  - Check whether player 2 won the game.

- If either player won the game, ensure that you exit the `while` loop.

6. Verify that your documentation makes sense and that you've added documentation to each of your functions.

7. Verify that your program works properly.

8. Upload your file to the Hw5 Assignments folder on D2L.

Example Output

Below is an example of the output from running the finished program. If you enter 125 for the seed and make the same choices as the players do in this example your output should look like this. Printed output is in blue, user input is in green.

```
Enter seed value: 125


Pig Dice
Enter name for player 1: Ziggy
Enter name for player 2: Elmer
    Hello Ziggy and Elmer, welcome to Pig Dice!

--- SCORES Ziggy: 0   Elmer: 0 ---
---------- Ziggy's turn ----------
    <<< Ziggy rolls a 2 >>>
    Points: 2
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 2 >>>
    Points: 4
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 5 >>>
    Points: 9
Roll again, Ziggy? (Y/N) N

--- SCORES Ziggy: 9   Elmer: 0 ---
---------- Elmer's turn ----------
     <<< Elmer rolls a 3 >>>
     Points: 3
Roll again, Elmer? (Y/N) Y
     <<< Elmer rolls a 5 >>>
     Points: 8
Roll again, Elmer? (Y/N) Y
     <<< Elmer rolls a 6 >>>
     Points: 14
Roll again, Elmer? (Y/N) Y
     <<< Elmer rolls a 2 >>>
     Points: 16
```

```
Roll again, Elmer? (Y/N) N

--- SCORES Ziggy: 9   Elmer: 16 ---
---------- Ziggy's turn ----------
    <<< Ziggy rolls a 3 >>>
    Points: 3
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 2 >>>
    Points: 5
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 5 >>>
    Points: 10
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 3 >>>
    Points: 13
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 5 >>>
    Points: 18
Roll again, Ziggy? (Y/N) N

--- SCORES Ziggy: 27  Elmer: 16 ---
---------- Elmer's turn ----------
    <<< Elmer rolls a 6 >>>
    Points: 6
Roll again, Elmer? (Y/N) Y
---------- Elmer's turn ----------
    <<< Elmer rolls a 1 >>>
    !!! PIG! No points earned, sorry Elmer !!!
(enter to continue)

--- SCORES Ziggy: 27  Elmer: 16 ---
---------- Ziggy's turn ----------
    <<< Ziggy rolls a 1 >>>
    !!! PIG! No points earned, sorry Ziggy!!!
(enter to continue)

--- SCORES Ziggy: 27  Elmer: 16 ---
---------- Elmer's turn ----------
    <<< Elmer rolls a 6 >>>
    Points: 6
Roll again, Elmer? (Y/N) N

--- SCORES Ziggy: 27  Elmer: 22 ---
---------- Ziggy's turn ----------
    <<< Ziggy rolls a 6 >>>
    Points: 6
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 2 >>>
    Points: 8
```

```
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 4 >>>
    Points: 12
Roll again, Ziggy? (Y/N) Y
    <<< Ziggy rolls a 5 >>>
    Points: 17
```

and so on…