Snakes & Ladders

Design and implement a Snakes and Ladders game in Python.

- There are 6 face dice which are being rolled by the player to their chance.
- The player starts from 0 and has to reach the final position (in our case, it's 100).
- There are some ladders which turn out to be lucky for the player as they shorten the way.
- There are some snakes present in between the game which turns out to be the enemy of the player as they just lengthen their way to 100.
- The position of snakes and ladders are stored in start:end format where start denotes the position of ladder/snake and end denotes the value that it reaches.

Position of snakes: {17 : 7, 54 : 34, 62 : 19, 98 : 79}
Position of ladders: {3 : 38, 24 : 33, 42 : 93, 72 : 84}

Structure:

- The game will only be **text output** based, i.e. no GUI will be shown to the user. Instead, it will use print statements to display the position of a particular player, end results etc.
- There are only **2 players** in the game and no spectators are allowed.
- The first thing that the game will ask to input will be names of the players. The input should first ask the name of Player 1 and then after giving input, it should ask the name of Player 2.
- The game will alternate between these 2 players.
  For a particular player's chance, it will ask to input his/her chance. If Player

1 enters " **roll** " (excluding quotes), then a random number will be generated between **1 and 6** (both inclusive).
The output will be shown and then the **final position will be displayed each time a player plays** his/her chance.
The output should resemble the following format :

```
###### Welcome to Snakes & Ladders Game #####
Enter the name of Player 1: Rahul
Enter the name of Player 2: Sachin
###### Let us Start #####
Player 1: roll
You got a 5
 Your final position is 5
Player 2: roll
You got a 3
 Your final position is 3
```

- The game consists of 2 modes: **Auto mode and manual mode**. For auto mode, a player will type **roll** and a random number will be generated imitating the roll of dice. For a manual mode, instead of entering roll, players can enter **any number between 1 and 20** and that particular number will imitate that throw. For example, if player 1 types 19, the output will be as shown:

```
Player 1: 19
You got a 19
 Your final position is 19
```

- The game will display your final position as a sum of last position and value of current chance. So let's say your last position was 16, and you got a 6, then the final position will be 22.

- Now here comes the interesting part. You will be provided the positions which contain **snakes or ladders**. You will also be provided with the values they are mapped to. If the final position corresponds to a snake or ladder then display the final position as that particular value.

Pro tip: *Use 2 different dictionaries to store the mappings, one for snakes*

and one for ladders.

For example, your dice value was 5 and the last position was 10, but position 15 contains a ladder that takes you to 35, then your final position will be 35.

- If either of the players reaches position 100, then that player will win and "**Game Successfully Finished**" will be printed. Also it will display which player won the game.

```
Player 1 won the game
###### Game Successfully Finished #####
```

Remember **not to overshoot the position 100**. For example, if current position is 97 and value of dice is 6, then position will be the same i.e. 97 as 97+6 = 103 which is not possible.

- During any player's chance, if he/she enters "**quit**" (without quotes), then the game will instantaneously end for that particular player and the other player will win the game.

Concepts required:

- List, Tuple, Dictionary
- Random number generation
- Iterations, conditional statements and functions

Libraries allowed:

- Random
  Use this link to know more about random number generators.

That's it. You are not allowed to use any other library for this assignment.
So now hop in and get ready to turn your ideas to code.

Evaluation will be done on the basis of **completion of project**, **code quality**, **handling of corner test cases** etc.

Feel free to reach out to your POD Leads for any clarification.