Assignment 3

In this Assignment you are going to program a number guessing game. Using a loop is not necessary to complete the assignment, however it can make the code much easier to read rather than using nested IF statements.

Using the nested IF statements approach, the following steps need to be performed:

1. Initialize a variable called *number* with a value of 10 and a variable called *numAttempts* with a value of 0
2. Ask the user for their name
3. Ask the user to guess a number between 1 and 20 and store that in a variable *guess*, increment the *numAttempts* variable
4. If *guess* is same as *number* then say your guess is correct and you guessed it in *n* attempt(s), where *n* is the value of *numAttempts* variable and your program ends.
5. If *guess* is lower than *number* then print "Your guess is too low".
   If *numAttempts* variable < 3, the goto step #7 else goto step #8.
6. If *guess* is higher than *number* then print "Your guess is too high".
   If *numAttempts* variable < 3, the goto step #7 else goto step #8
7. Repeat step #3 - #6.
8. If the user does not guess the number even after three attempts then print “Your three guesses are over, the number I was thinking of was 10”

Sample 1

Hello! What is your name?
Albert
Well, Albert, I am thinking of a number between 1 and 20.
Take a guess.
15
Your guess is too high.
Take a guess.
4
Your guess is too low.
Take a guess.
12
Your guess is too high. Your three guesses are over. The number I was thinking of was 10

Sample 2

Hello! What is your name?
Albert
Well, Albert, I am thinking of a number between 1 and 20.
Take a guess.
8
Your guess is too low.

Take a guess.
12
Your guess is too high.
Take a guess.
10
Good job, Albert! You guessed my number in 3 guesses!

Optional: Instead of fixing the number to be guessed to 10, you can initialize 'number' variable with a random number between 1 and 20 that you generate using the functions in the random module. This way every time you re run the program, it is a different number to be guessed.

Part - II (5 points)
Rock, Paper, Scissor:

Write a program to simulate the paper-rock-scissor game. Each of the two players type in P, R, or S. The program then announces the winner based on the rule: Paper covers Rock, Rock breaks Scissors, Scissors cut Paper, or Nobody wins. Be sure to allow the players to use lowercase as well as uppercase letters.

You **MUST** implement a **function** that takes the choices that both the players made and returns 1 if the player 1 won and 2 if the player 2 won and 0 if it is a tie. So the logic of deciding who won based on the rock, paper, scissor rule must be programmed inside this function.
You **MUST** call this function to decide who won and you must display the result of the game based on the return value of the function.
Sample run:
Player 1: Please enter either (R)ock, (P)aper, or (S)cissors: P

Player 2: Please enter either (R)ock, (P)aper, or (S)cissors: s

Player 2 wins.

Optional: Your program can allow the users to play repeatedly say 5 times. Your program should then keep a score for each player and display that after each play.
Sample run (with optional part):
Player 1: Please enter either (R)ock, (P)aper, or (S)cissors: P

Player 2: Please enter either (R)ock, (P)aper, or (S)cissors: s

Player 2 wins.

Scores after this play:

Player 1: 0

Player 2: 1

Thanks!!