Task 1

```
Welcome to Battle of Dices!

Press ENTER to roll the dice...
Player 1 rolled:  6
Player 2 rolled:  4

Press ENTER to continue...
Player 1 wins this round!
Because  6  is greater than  4
The game score is Player1  1  vs.  0  Player 2.
This heated Battle of Dices is still going on! Who will win in the end?
PS C:\Users\jaden\Desktop\JadenPython>
```

Task 2

```
Welcome to Battle of Dices!

Press ENTER to roll the dice...
Traceback (most recent call last):
  File "c:\Users\jaden\Desktop\JadenPython\Lab3.py\lab-battle-of-dices.py", line 28, in <module>
    print("Player 1 rolled: " + player1_roll)
          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~^~~~~~~~~~~~~~
TypeError: can only concatenate str (not "int") to str
PS C:\Users\jaden\Desktop\JadenPython> '
```

Task 3

```
Welcome to Battle of Dices!

Press ENTER to roll the dice...
Player 1 rolled:  1
Player 2 rolled:  6

Press ENTER to continue...
Player 2 wins this round!
Because  6  is greater than  1
The game score is Player1  0  vs.  1  Player 2.
This heated Battle of Dices is still going on! Who will win in the end?
PS C:\Users\jaden\Desktop\JadenPython>
```

Task 4

```
Press ENTER to roll the dice...
Player 1 rolled:  2
Player 2 rolled:  1

Press ENTER to continue...
Player 1 wins this round!
Because  2  is greater than  1
The game score is Player1  3  vs.  0  Player 2.
Player 1 is the newest Battle of Dices Champion!
PS C:\Users\jaden\Desktop\JadenPython>
```

```
while player1_wins < 3 and player2_wins < 3:
```

Task 5
Nothing happens, it means the same thing, it is written the code differently only.

Task 6

```
if player1_wins == 3:
    print("Player 1 is the Greatest Battle of Dices Champion! ")
elif player2_wins == 3:
    print("Player 2 is the Greatest Battle of Dices Champion! ")
else:
    print("This Amazing Battle of Dices is still going on! Who will win in the end? ")
```

Task 7 check attached code

Task 8

```
player1_roll = random.randint(1, 20)
player2_roll = random.randint(1, 20)
```

Task 9
Recursion is an interesting concept, I always assumed the best way to perform a loop type sequence within python was done with a while or a for loop. Now knowing this can be done with defining a function calling itself adds layers to what can be done with coding.

Task 10

```
while player1_wins < 3 and player2_wins < 3:
```

Task 11

```
round_num += 1
print("You are at Round", round_num)
```

Task 12, Task 13, Task 14 check attached code.