Inspiring Excellence

Course Title: Programming Language II

Course Code: CSE 111

Assignment No: 2 | Functions (Homework)

Total Tasks	6
Submission Type	Handwritten

Home Task 1: Hospital Fee

Suppose, you work for a hospital. Your job is to find out the highest fee received by the hospital and the person(s) who paid the highest amount. Now, write a function that takes multiple keyword arguments that returns the highest amount of fee taken by the hospital and the person(s) who paid the highest fees.

*[Hint: You might need to use **kwargs to solve this task.]*

Function Call: max_amount, max_payer = hospital_fee(Neymar = 1000, Dembele = 600, Reus = 500, Bale = 1000)	**Sample Output:** Highest fee was 1000 tk which was paid by Neymar, Bale.
Function Call: max_amount, max_payer = hospital_fee(Mashrafe = 400, Bumrah = 900, Steyn = 1200, Cummins = 900, Wood = 400, Marsh = 700)	**Sample Output:** Highest fee was 1200 tk which was paid by Steyn.

Home Task 2: 007

Write a function that takes in a list of integers and returns True if it contains 007 in order.

Function Call: is_james_bond([1, 2, 4, 0, 0, 7, 5])	**Sample Output:** True
Function Call: is_james_bond([1, 7, 2, 0, 4, 5, 0])	**Sample Output:** False
Function Call: is_james_bond([1, 0, 2, 0, 4, 7, 5])	**Sample Output:** True

Home Task 3: Section Assigning

"You are tasked with organizing students into different sections in a school. There are 'ABCDE' (five) sections available, and you have a list of student names. You need to create a function that will take these inputs and assign students to sections based on their names.

The first parameter, 'sections', should be a string representing the available sections

(e.g., 'ABCDE' for five sections).

The second parameter should be a variable number of student names.

The function should calculate the sum of ASCII values of characters in each student's name, then use the modulo operation with the number of sections to determine the section based on the calculated value. It should return a dictionary where each section is a key, and the associated value is a list of students assigned to that section.

Function Call:	**Output:**
assign_students_to_sections ('ABCDE', 'Alice', 'Bob', 'Charlie', 'David', 'Eve', 'Frank', 'Grace')	{'A': ['Bob'], 'B': ['Charlie'], 'C': ['Grace'], 'D': ['Alice', 'David', 'Eve', 'Frank'], 'E': [ ]}

Explanation:
For example, your name is Charlie and there are 5 Sections "ABCDE". So, the ASCII values for your name is:
C = 67
h = 104
a = 97
r = 114
l = 108
i = 105
e = 101
So, the summation is 67+104+97+114+108+105+101= 696
Now, as you have 5 sections,
696%5 = 1
So, Charlie will be on section 1 which means section B.

Home Task 4: Username Generator

Complete the **username_generator** function in such a way that returns a username based on the following equation:

User name = First 3 letters of the first name upper-cased + Entire middle name + Last 3 letters of the last name lower-cased + Underscore + Last 4 digits of the student id

Your function should be able to handle user inputs having no middle name.

```
Function Call
first_name, middle_name, last_name, student_id= input ("First Name:"),
input  ("Middle Name:"), input ("Last Name:"), int (input ("Student ID:"))

print(username_generator (first_name, last_name, student_id))
print(username_generator(first_name, last_name, student_id,
middle_name))
```

Sample Input: First Name: Jahanara Middle Name: Last Name: Islam Student ID:18101954	**Output:** JAHlam_1954
Sample Input: First Name: Shakib Middle Name: Bin Last Name: Hasan Student ID:17305165	**Output:** SHABinsan_5165
Sample Input: First Name: MD Middle Name: Last Name: Ishmam Student ID:19992564	**Output:** MDmam_2564

Home Task 5: Key Generator

As a security expert working in a company, you are required to create encrypted keys for some employees. Your supervisor will give you a list of names of the employees. Now write a function that creates an encrypted key for each of these employees by following the given conditions:

- The first character and last character will remain in their position. But the first character will be converted to lowercase and the last character will be converted to uppercase.
- The characters in the middle will be reversed and substituted by their ASCII values.

Function call: key_list = key_generator ("Alex", "Bob", "Trudy")	**Sample Output:** Encrypted Keys: ['a101108X', 'b111B', 't100117114Y']
Explanation: Let's say, for "Trudy", T ⇒ t y ⇒ Y ASCII of r = 114, u = 117 and d = 100 Encrypted key = t100117114Y	

Home Task 6: Rock-Paper-Scissor

You have to build a Rock-Paper-Scissor (Computer Vs Player) game. Players can set the rounds. If the player gets to win more rounds than the computer then the player wins, otherwise the computer wins. The game can be a tie too. We know the rules: Rock crushes scissors, scissors cut paper, and paper covers rock.

A function **playRockPaperScissor(rounds)** will take the number of rounds as an argument and return the total scores of the player, total scores of the computer and the winner. Also, inside the function, show the opponent's action after taking each player's action as input.

Note: You can use random.choice() function from Python's random module to randomize the opponent's action.

Let, X= random.choice([a, b, c])
Here, X can be any value among a, b, c randomly.

Sample Input 1:	**Sample Output 1:**
3 rock paper scissor	Computer: paper Computer: rock Computer: rock Your Score: 1 Computer's Score: 2 Computer has won the game!
Sample Input 2:	**Sample Output 2:**
5 rock paper scissor rock paper	Computer: scissor Computer: rock Computer: rock Computer: rock Computer: paper Your Score: 2 Computer's Score: 1 You have won the game!