Inspiring Excellence

Course Title: Programming Language I
Course Code: CSE 110
Assignment no: 5 (Dictionary and Tuples)

This Assignment is to help you develop your concept of Tuples and Dictionary in Python.
[Please follow variable naming conventions including using meaningful variable names for all the tasks]
When you run your code, please make sure your outputs exactly match the sample outputs for each of the sample inputs given. Check if your code works for other valid inputs not given in the samples.
Write the code in Python to do the following tasks:

Part 1: Tuples

Task 1

Assume, you have been given a tuple.

a_tuple = ("The Institute", ("Best Mystery & Thriller", "The Silent Patient", 68821), 75717, [1, 2, 3, 400, 5, 6, 7], ("Best Fiction", "The Testaments", 98291))

Write **one line** of Python code to access and print the value 400.

==========

Output: 400

==========

Task 2

Assume, you have been given a tuple. Write a Python program that creates a **new tuple** excluding the first and last two elements of the given tuple and prints the new tuple.

Hint: You may use tuple slicing.

==========

Sample Input 1:
(10, 20, 24, 25, 26, 35, 70)

Sample Output 1:
(24, 25, 26)

==========

Sample Input 2:
(-10, 20, 30, 40)

Sample Output 2:
()

==========

Sample Input 3:
(-10, 20, 25, 30, 40)

Sample Output 3:
(25,)

==========

Task 3

Assume, you have been given a tuple.

```
book_info = (
("Best Mystery & Thriller","The Silent Patient",68,821),
("Best Horror","The Institute",75,717),
("Best History & Biography","The five",31,783 ),
("Best Fiction","The Testaments",98,291)
)
```

Write a Python program that prints the size of the tuple and all its elements as shown below.

==========

Output:
Size of the tuple is: 4
('Best Mystery & Thriller', 'The Silent Patient', 68, 821)
('Best Horror', 'The Institute', 75, 717)
('Best History & Biography', 'The five', 31, 783)
('Best Fiction', 'The Testaments', 98, 291)

==========

Task 4

Assume, you have been given a tuple with details about books that won the Good Reads Choice Awards.

```
book_info = (
("Best Mystery & Thriller","The Silent Patient",68821),
("Best Horror","The Institute",75717),
("Best History & Biography","The five",31783 ),
("Best Fiction","The Testaments",98291)
)
```

Write a Python program that prints the award category, the book name, and its total votes earned as shown below.

[Must use Tuple unpacking for printing and need to handle the quotation marks as a part of the output]

==========

Output:
The Silent Patient won the 'Best Mystery & Thriller' category with 68821 votes
The Institute won the 'Best Horror' category with 75717 votes

The five won the 'Best History & Biography' category with 31783 votes
The Testaments won the 'Best Fiction' category with 98291 votes

==========

Task 5

Write a python program that takes an input from the user and finds the number of times that the input is present in a given tuple.

==========

Example 1:

Given tuple: (10, 8, 5, 2, 10, 15, 10, 8, 5, 8, 8, 2)

Sample Input 1:

8

Sample Output 1:

8 appears 4 times in the tuple

==========

Example 2

Given tuple: (10, 8, 5, 2, 10, 15, 10, 8, 5, 8, 8, 2)

Sample Input 2:

1

Sample Output 2:

1 appears 0 times in the tuple

==========

Task 6

Write a Python program to reverse a given tuple.

[You are not allowed to use tuple slicing]

==========

Note: Unlike lists, tuples are immutable. So, in order to reverse a tuple, we may need to convert it into a list first, then modify the list, and finally convert it back to a tuple.

==========

Example 1:

Given tuple: ('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h')

Output:

('h', 'g', 'f', 'e', 'd', 'c', 'b', 'a')

==========

Example 2:

Given tuple: (10, 20, 30, 40, 50, 60)

Output:

(60, 50, 40, 30, 20, 10)

==========

Part 2: Dictionary

Task 7

Suppose you are given two dictionaries.
Now create a new dictionary "marks", merging the two dictionaries, **so that the original two dictionaries remain unchanged.**

Note: You can use dictionary functions.

==========

Given:

{'Harry':15, 'Draco':8, 'Nevil':19}

{'Ginie':18, 'Luna': 14}

Output:

{'Harry': 15, 'Draco': 8, 'Nevil': 19, 'Ginie': 18, 'Luna': 14}

==========

Given:

{'A':90, 'B': 0}

{'C':50}

Output:

{'A': 90, 'B': 0, 'C': 50}

==========

Task 8

Write a Python program that takes a dictionary as an input from the user and then prints the average of all the values in the dictionary.

[You are not allowed to use len() and sum()]

==========

Hint (1): For taking dictionary input

Approach (1): For taking dictionary as an input from the user, you may take the whole dictionary as a string using the input() function. Then you can use the split(), strip() functions and conditions to get the keys and values from the string. Finally, you can make the dictionary using the obtained data.

Approach (2): If the first approach seems too difficult you can create an empty dictionary and then just run a simple loop. For each iteration ask the user for a key and a value using the input() function and keep updating the dictionary with the key and value.

Hint (2): After you have a dictionary, you can use dictionary functions to get all the values from it, run loop to calculate the sum and the total number of values in the dictionary in order to find out the average.

==========

Sample Input 1:

{'Jon': 100, 'Dan':200, 'Rob':300}

Sample Output 1:

Average is 200.

==========

Sample Input 2:

{'Jon': 100, 'Dan':200, 'Rob':30, 'Ned':110}

Sample Output 2:

Average is 110.

==========

Task 9

Suppose there is a dictionary named exam_marks as given below.

exam_marks = {'Cierra Vega': 175, 'Alden Cantrell': 200, 'Kierra Gentry': 165, 'Pierre Cox': 190}

Write a Python program that takes an input from the user and creates a new dictionary with only those elements from 'exam_marks' whose keys have values higher than the user input (inclusive).

==========

Sample Input 1:

170

Sample Output 1:

{'Cierra Vega': 175, 'Alden Cantrell': 200, 'Pierre Cox': 190}

==========

Sample Input 2:

190

Sample Output 2:

{'Alden Cantrell': 200, 'Pierre Cox': 190}

==========

Task 10

Write a Python program that finds the largest value with its key from a given dictionary.

[You are not allowed to use the max() function for this task]

Note: You do not need to take the dictionaries as an input from the user but your code should work for any given dictionary. Also, you need to handle the quotation marks as a part of the output.

Hint: Think of membership operators (in and not in). You can use dictionary functions to get the values.

==========

Sample 1:

Given:

{'sci fi': 12, 'mystery': 15, 'horror': 8, 'mythology': 10, 'young_adult': 4, 'adventure':14}

Output:

The highest selling book genre is 'mystery' and the number of books sold are 15.

==========

Sample 2:

Given:

{'sci fi': 5, 'mystery': 3, 'horror': 14, 'young_adult': 2, 'adventure':9}

Output:

The highest selling book genre is 'horror' and the number of books sold are 14.

==========

Task 11

Write a Python program that takes a String as an input from the user and counts the frequency of each character using a dictionary. For solving this problem, you may use each character as a key and its frequency as values. [You are not allowed to use the count() function]

Hint: You can create a new dictionary to store the frequencies. You may ignore case for simplicity (i.e. may consider P and p to be the same).

==========

Sample Input:
"Python programming is fun"

Sample Output:
{'p': 2, 'y': 1, 't': 1, 'h': 1, 'o': 2, 'n': 3, 'r': 2, 'g': 2, 'a': 1, 'm': 2, 'i': 2, 's': 1, 'f': 1, 'u': 1}

==========

Task 12

Suppose you are given the following dictionary where the values are lists.

dict_1 = {'A': [1, 2, 3], 'b': ['1', '2'], "c": [4, 5, 6, 7]}

Write a Python program that counts the total number of items in a dictionary's values and prints it.

[Without using sum(), len(), count() functions]

Note: Make changes to the above dictionary and see if your code works properly for other dictionaries as well.

==========

Output:

9

==========

Task 13

Suppose you have been given the following list of tuples.

list_1 = [("a", 1), ("b", 2), ("a", 3), ("b", 1), ("a", 2), ("c", 1)]

Write a Python program that converts this list of tuples into a dictionary and then prints the dictionary.

[You are not allowed to use set]

Hint: Think of membership operators (in and not in).

==========

Output:

{'a': [1, 3, 2], 'b': [2, 1], 'c': [1]}

==========

Task 14

```
var1 = var2 = var3 = var4 = var5 = var6 = False
result1 = result2 = result3 = result4 = result5 = result6 =
result7 = result8 = result9 = result10 = False
var1 = 4 < 3 - 1
var2 = var1 and False
var3 = False
var4 = True
var5 = False
var6 = var3 and True
result1 = (var1 or var2) and (8 * 10 > 45)
result2 = (var1 or var2) and (result1 and False)
result3 = (var1 and result1) or result2
result4 = (var1 or var2) or ((var3 and var1) and False)
result5 = (var1 and var2) and (result3 or var1)
result6 = ((var3 or var2) and not result5) or True
result7 = (var4 and result1) and ((result1 and False) or True)
result8 = ((var1 and result3) and (var5 or var6)) and True
result9 = ((result2 and var2) or (result7 and var1)) and False
result10 = not(var1 and True)
```

Show the values of the result variables in the above program:

result1	
result2	
result3	
result4	
result5	
result6	
result7	
result8	
result9	
result10	

Task 15

```
var1 = False
var2 = False
var3 = False
var4 = False
var5 = False
var6 = False
result1 = False
result2 = False
result3 = False
result4 = False
result5 = False
result6 = False
result7 = False
result8 = False
result9 = False
result10 = False
var1 =((not True) or True) and False
var2 = var1 and False
var3 = True and not False
var4 = False
var5 = True
var6 = var3 and False
result1 = (var1 and var2) and (40 % 3) > 45 or (var5 and var6)
result2 = (var1 or var2) or (result1 and False)
result3 = (var1 and result1) or result2 or var5
result4 = (var1 or var2) or ((var3 and var1) and False)
result5 = (var1 and var2) and (result3 or var1)
result6 = ((var3 or (not var2)) and (result5)) or True
result7 = (var4 and result1) and ((result1 and False) or True)
result8 = ((var1 and result3) and ((not var5) or var6)) and True
result9 = ((result2 and var2) or ((not result7) and var1)) and not False
result10 = not(var1 and True)
```

Show the values of the result variables of the program:

result1	
result2	
result3	
result4	
result5	
result6	
result7	
result8	
result9	
result10	

Task 16

```
myList = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
index1 = 0
index2 = 0
index1 = 1
while(index1<10):
  myList[index1] = index1+4
  index2 = 1
  while(index2<index1):
      myList[index1] = myList[index1] +myList[index2]-index1
      index2 = index2+1
  print(myList[index1])
  index1 = index1+1
```

Output

Task 17

```
myList = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
index1 = 0
index2 = 0
index1 = 1
while (index1 < 10):
  myList[index1] = index1 + 4
  index2 = 1
  while (index2 < index1):
    myList[index1] = myList[index1] + myList[index2] - index1
    index2 = index2 + 1
  print(myList[index1])
  index1 = index1 + 1
```

Output

Task 18

```
myList = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
b = []
index1 = 0
index2 = 0
index1 = 1
b  = myList
while(index1<10):
  myList[index1] = index1+2;
  index2 = 1;
  while(index2<index1):
    myList[index1] = b[index1]+myList[index2]-index1
    index2 = index2+1
  print(myList[index1])
  index1 = index1+1
```

Output

Task 19

```
myList = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
b = []
index1 = 0
index2 = 0
index1 = 1
b = myList
while (index1 < 10):
  myList[index1] = index1 + 1
  index2 = 1
  while (index2 < index1):
    myList[index1] = b[index2-1] + myList[index2] - index1
    index2 = index2 + 1
  print(myList[index1])
  index1 = index1 + 1
```

Output

Optional Tasks (20-25) [Ungraded]

Task 20

Given a list of tuples, your task is to multiply the elements of the tuple and return a list of multiplied elements as shown below.

==========

Example 1:

Given:

[(2, 3), (4, 5), (6, 7), (2, 8)]

Output:

[6, 20, 42, 16]

==========

Example 2:

Given:

[(11, 22), (33, 55), (55, 77), (11, 44)]

Output:

[242, 1815, 4235, 484]

==========

Task 21

Assume, you have been given a tuple as below.

a_tuple = ([1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12])

Write a Python program that asks the user for an input (can be any data type) and **replaces the last element of each of the inner lists with the user given value.**

==========

Sample Input 1:

abc

Sample Output 1:

([1, 2, 'abc'], [4, 5, 'abc'], [7, 8, 'abc'], [10, 11, 'abc'])

==========

Sample Input 1:

1000

Sample Output 1:

([1, 2, '1000'], [4, 5, '1000'], [7, 8, '1000'], [10, 11, '1000'])

==========

Task 22

Suppose there is a dictionary named my_dictionary.

my_dictionary = {'c1':'Red', 'c2':'Green', 'c3':None, 'd4':'Blue', 'a5':None}.

Now write a Python program to remove empty items from the dictionary. [Empty items means keys without any values (None)].

==========

Output:

{'c1':'Red', 'c2':'Green', 'd4':'Blue'}

==========

Task 23

Suppose you are given a dictionary as shown below. Write a Python program that takes two inputs from the user representing the lower (inclusive) and upper (exclusive) of a range of values. Your task is to extract all the items from the dictionary whose values lie in the range given as input by the user.

dict_1 = {'a' : 6, 'b' : 7, 'c' : 9, 'd' : 8, 'e' : 11, 'f' : 12, 'g' : 13}

==========

Sample Input 1:

9, 12

Sample Output 1:

{'c': 9, 'e': 11}

Explanation: Keys with values within the range of 9 and 11 are extracted.

==========

Sample Input 2:

14, 18

Sample Output 2:

{}

Explanation: No values in range.

==========

Task 24

Given a list of tuples, your task is to group the tuples based on the **second** element in the tuples as shown in the examples below. We can achieve this using **dictionary** by checking the second element in each tuple.

[You do not need to take tuple as input and can assume that it is given as below]

==========

Example 1:

Given:

[(20, 80), (31, 80), (1, 22), (88, 11), (27, 11)]

Output:

{80: [(20, 80), (31, 80)],
11: [(88, 11), (27, 11)],
22: [(1, 22)]}

==========

Example 2:

Given:

[(20, 'Sad'), (31, 'Sad'), (88, 'NotSad'), (27, 'NotSad')]

Output:

{'NotSad': [(88, 'NotSad'), (27, 'NotSad')],
'Sad: [(20, 'Sad'), (31, 'Sad')]}

==========

Task 25

Write a python program that takes 2 inputs from the user.

- Input1: A string containing a single sentence or multiple sentences. Multiple sentences are separated by full stops.
- Input2: A list containing special characters.

After taking the inputs, your task is to create a dictionary where the **keys** will be the corresponding special characters from the given list and the **values** will be the words in the given sentence.

You need to find out which key belongs to which value. To do that, you should calculate the **index number** using the formula given below. **The calculated index is the corresponding key of the word.** Make sure no duplicate values are inserted.

The formula:
Index value of special character list = (ASCII sum of the word) % (length of special character list)

==========

Sample Input 1:
'I love Programming. Python is love.'
['@', '$', '&', '#']

Sample Output 1:
Words in the given String: ['I', 'love', 'Programming', 'Python', 'is', 'love']
Answer: {'$': ['I'], '&': ['love', 'Python'], '#': ['Programming'], '@': ['is']}

Explanation 1:
From the string 'I love Programming. Python is love.' the {key: value} pair is calculated accordingly:

Word	ASCII sum	List length	Calculated index	Key
I	73	4	73%4 = 1	'$'
love	438	4	438%4 = 2	'&'
Programming	1155	4	1155%4 = 3	'#'
Python	642	4	642%4 = 2	'&'
is	220	4	220%4 = 0	'@'
love	438	4	438%4 = 2	'&'

The calculation is shown in the given table. Full stops after ‘Programming.’ and ‘love.’ are avoided. Also, the last word ‘love’ is a duplicate, and therefore, it was not inserted again as a value.

==========

Sample Input 2:
'The secret of getting ahead is getting started.'
['-', '=', '+', '*', '%']

Sample Output 2:
Words in the given String: ['The', 'secret', 'of', 'getting', 'ahead', 'is', 'getting', 'started']
Answer: {'%': ['The', 'getting', 'ahead', 'started'], '=': ['secret'], '*': ['of'], '-': ['is']}

Explanation 2:
From the string 'The secret of getting ahead is getting started.' the {key: value} pair is calculated accordingly:

Word	ASCII sum	List length	Calculated index	Key
The	289	5	289%5 = 4	'%'
secret	646	5	646%5 = 1	'='
of	213	5	213%5 = 3	'*'
getting	754	5	754%5 = 4	'%'
ahead	499	5	499%5 = 4	'%'
is	220	5	220%5 = 0	'-'
getting	754	5	754%5 = 4	'%'
started	759	5	759%5 = 4	'%'

The calculation is shown in the given table. Full stop after ‘started.’ is avoided. Also, the word ‘getting’ is a duplicate and therefore it was not inserted again as a value.

==========