Name - Shantanu Deshpande

Class - SYCSE

Roll no. - 7

Batch - S1

Practical No. 5

Aim:- Apply dictionary data type in python to create solution for Problem statement #5 with lists and Tuples.

Objective:- The objective of this lab work is :

1. Students will be able to use list and set data types.
2. Students will be able to apply dictionary.
3. Students will be to use dictionary data type and its functions.
4. Students will be able to choose appropriate data structure to come up with solution of given problem statement.

Outcomes:- After completing the lab work #5 students will be able to :

1. Use dictionary data type to come up with programming solution for given data in the programming statement.
2. Apply basics list data type and its functions to solve programming problems.

Pre-Requisite:-

1. List and tuple data type.
2. Dictionary data type.

Input-Output÷

1. Input will be as specified in problem statement #5.
2. Output shall be as desired in problem statement #5.

Theory÷

A dictionary in python works similar to the dictionary in the real world. keys of a dictionary must be unique and of immutable data types such as strings, Integers and tuples, but the key values can be repeated and be of any type.

Nested dictionary÷ Nesting dictionary means putting a dictionary inside another dictionary. Nesting is of great use as the kind of information we can model in programs is enpanded greatly.

```
nested_dict={'dict1': {'key_A': 'value-A'},
             'dict2' : {'key_B': 'value _ B'}}
```

262.

A nested dict is a dictionary whithin a dictionary. A very sample thing.

```
>>> d={}
>>> d['dict1']={}
>>> d['dict']['innerkey']='value'
>>> d

{'dict1': {'innekey': 'value'}}
```

You can also use a default dict from the collections package to facilitate creating nested dictionaries.

```
>>> import collections
>>>d  = collections. defaultdict (dict)
>>> d ['dict 1'] ['innerkey'] = 'value'
>>> d # currently a defaultdict type
  default dict (<type 'dict' >, {'dict1' : {'innerkey' : 'value' }})
>>> dict (d) # but is enactly like a normal dictionary.
  {'dict1': {'innerkey': 'value' }}
```

Source code:-

```
def rainaverage (l):
    data_dict = {}
    out = {}
    fortup in l
        if tup[0] in data-dict :
            data _dict [tup [0] ] . append (tup[1])
        else :
            data_dict (tup [0] ] = [tup[1]]

  for c ign data-dict :
      ar= sum(data _dict [c]) || en (data-dict [c])
      out . append (tuple [c, "%.2f " %, ar ]])

  return out

print (sorted (rainaverage ( [('Bombay', 848), ('Madras', 103),
      ('Bombay', 92), ('Banglore', 201 ), ('madras',
      128 )])))
```

MIT™
A group of Academic & Research Institutions

7-Shantanu Deshpande

Quest for Excellence

Output:-

```
[('Banglore', '201.00'), ('Bombay', '885.50'), ('Madaras',
'115.50')]
```

Assessment questions :-

1. Compare lists and tuples

Solⁿ:-

	List	Tuples
1.	Lists are mutable.	Tuples are immutablo.
2.	Implication of iterations is Time-consuming.	The implication of iterations is comparatively faster.
3.	The list is better for performing operations, such as insertion and deletion.	Tuple data type is appropriate for accessing the elements.
4.	Lists consume more memory.	Tuple consume less memory as compared to the list.
5.	Lists have several built-in methods.	Tuple does not have many built-in methods.

TEJAS