Task-4: use various data types, List, Tuples and dictionary in Python Programming.

25/08

Aim: To use various data types List, Tuples and Dictionary in Python Programming.

a) You are working on a Python Project that requires you to manage and manipulate a list of members. Your task is to create a Python Program that demonstrate the following list operations.

1. Add Elements: Add elements to the list
2. Remove elements: Remove specific elements from the list.
3. Sort elements: sort the list in ascending and descending order.
4. Find minimum and maximum: Find the maximum and minimum elements in the list
5. Calculate sum and Average: Calculate the sum and average of the element in the list.

Algorithm:

1. Start
2. For adding elements to a list first create a list with name "list" and assign values with [ ] brackets, inorder to add a new value use function append().
3. For removing a specific element use "pop" or "remove".
4. for sorting elements use "Sorted (list)" function.
5. for finding minimum value use "min (list)" and for maximum use "max (list)".
6. for sum use function "sum (list)" and for average use the formula "sum (list) / len (list)".
7. Print output
8. End.

Program:

```
# Add elements: Add element to the list.
list = [10,20]
a=30
list.append(a)
Print(list)
#Remove elements: Remove specific elements from the list.
list.pop(1) # by index value.
Print(list)
list.remove(10) # by itemname
Print(list)
# sort elements: sort the list in ascending and descending order.
l = [5,8,9,15,30,89]
Print(sorted(l))
#find minimum and maximum: Find the minimum and maximum elements in the list.
Print("The minimum value is:", min(l))
Print("the maximum value is:", max(l))
#calculate sum and Average.
Print("The sum is:", sum(l))
Print("The average is:", (sum(l)/len(l)))
```

output:

```
=== Restart: C:/Users/student.
[10, 20, 30]
[10, 30]
[30]
[5, 8, 9, 15, 30, 89]
The minimum value is: 5
The maximum value is: 89
The sum is: 156
The average is: 26.0
```

b) You are tasked with creating a python program that showcase operations on tuples. Tuples are immutable sequences, similar to lists but with the key differences that they cannot be changed after creation. Your program should illustrate the following tuple operations:

1. Create a tuple: Define a tuple with elements of different data types (10, 'hello', 3.14, 'world')
2. Access elements: Access individual elements and slices of the tuple.
3. Concatenate Tuples: combine two tuples to create new tuple.
4. Immutable Nature: Attempt to madify elements of the tuple and handle the resulting error.

Algorithm:

1. start
2. To create a tuple use tuple_name = (values)
3. To access the elements of a tuple either use the index values. or the tuple slicing.
4. To concatenate tuples use the operator "+"
5. Try to modify the tuple elements elements by assigning the values directly like:
   tuple(index) = new_value, will result in an error as it is immutable.
6. Print the output.
7. End

Program:

```
# create a Tuple: Define a tuple with elements of different data types (10, 'hello', 3.14, 'world')
tuple = (10, 'hello', 3.14, 'world')
Print (tuple)
# Acces Elements: Access individual elements and slices of the tuple
for i in tuple:
Print (i)
Print (tuple[1:3])
Print (tuple[:-1])
# concatenate Tuples: combine two tuples to create a new Tuple:
t2 = (5, 0.5)
t3 = tuple + t2
Print (t3)
# Immutable Nature: Attempt to modify elements of tuple and handle the resulting error.
tuple (3) = "PI" # ERROR.
```

output:

```
= = Restart: C:/Users/student.D.
(10, 'hello', 3.14, 'world')
10
hello
3.14
world
('hello', 3.14)
(10, 'hello', 3.14)
```

Program:

```
# create a Dictionary: Define a dictionary with key-value Pairs of different data types. ({'name': 'Alice', 'age': 30, 'city': 'New York'}
dictionary = {'name': 'Alice', 'age': 30, 'city': 'New York'}
Print (dictionary)
# Acess values: Access values using keys.
Print (dictionary ['name'])
Print (dictionary ['age'])
# Modify Dictionary: Update values, add new key-value Pairs, and remove existing Pairs.
dictionary ['name'] = "James"
Print (dictionary)
dictionary. Pop ('city')
Print (dictionary)
# Iterate over Dictionary: Use loops to iterate over keys or values.
for k in dictionary:
Print ("KEY:", K)
Print (dictionary. items())
```

output:

```
= Restart: C:/Users/student. DesktOP-45NH6EA/Desktop/
{'name': 'Alice', 'age': 30, 'city': 'New York'}
Alice
30
{'name': 'James', 'age': 30, 'city': 'New York'}
{'name': 'James', 'age': 30}
key: name
key: age
dict_items ([('name', 'James'), ('age', 30)])
```

c) You are tasked with creating a Python Program that showcases operations on dictionaries. Dictionaries in Python are unordered collections of items. Each item is a pair consisting of a key and a value. Your program should illustrate the following dictionary operations.

1. Create a dictionary: Define a dictionary with key-value pairs of different data types. {'name': 'Alice', 'age': 30, 'city': 'New York'}
2. Access values: Access the values using keys.
3. Modify Dictionary: Update values, add new key-value pairs and remove existing pairs.
4. Iterate over Dictionary: Use loops to iterate over keys or values.

Algorithm:

1. Start the Program
2. Define a dictionary with key-value pairs of different data types.
3. Retrive values from the dictionary using their corresponding keys
4. Modify Dictionary.
5. Iterate over dictionary.
6. Stop

VEL TECH - CSE	
EX NO.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	
TOTAL (20)	
SIGN WITH DATE	15

Result:

Thus, various data types, List, Tuples and Dictionary in Python Programming was used and verified successfully.