Experiment No. 3
To explore basic data types of python like strings, list, dictionaries and tuples
Date of Performance: 31/01/2024
Date of Submission: 14/02/2024

Experiment No. 3

Title: To explore basic data types of python like strings, list, dictionaries and tuples.

Aim: To study and explore basic data types of python like strings, list, dictionaries and tuples.

Objective: To introduce basic data types of python

Theory:

Lists: are just like dynamic sized arrays, declared in other languages (vector in C++ and ArrayList in Java). Lists need not be homogeneous always which makes it a most powerful tool in Python.

Tuple: A Tuple is a collection of Python objects separated by commas. In someways a tuple is similar to a list in terms of indexing, nested objects and repetition but a tuple is immutable unlike lists that are mutable.

Set: A Set is an unordered collection data type that is iterable, mutable and has no duplicate elements. Python's set class represents the mathematical notion of a set.

Dictionary: in Python is an unordered collection of data values, used to store data values like a map, which unlike other Data Types that hold only single value as an element, Dictionary holds key:value pair. Key value is provided in the dictionary to make it more optimized.

List, Tuple, Set, and Dictionary are the data structures in python that are used to store and organize the data in an efficient manner.

List	Tuple	Set	Dictionary
List is a non-homogeneous data structure which stores the elements in single row and multiple rows and columns	Tuple is also a non-homogeneous data structure which stores single row and multiple rows and columns	Set data structure is also non-homogeneous data structure but stores in single row	Dictionary is also non-homogeneous data structure which stores key pairs
Tuple can be represented by			

List can be represented by [ ]	( )	Set can be represented by { }	Dictionary can be represented by { }
Set will not allow duplicate elements			
List allows duplicate elements	Tuple allows duplicate elements	Set will not allow duplicate elements	but keys are not duplicated
List can use nested among all	Tuple can use nested among all	Set can use nested among all	Dictionary can use nested among all
Example: [1, 2, 3, 4, 4, 5]	Example: (1, 2, 3, 4, 5)	Example: {1, 2, 3, 4, 5}	Example: {1, 2, 3, 5}
Dictionary can be created using **dict()** function.			
List can be created using **list()** function	Tuple can be created using **tuple()** function.	Set can be created using **set()** function	
Set is mutable i.e we can make any changes in set. But elements are not duplicated.			
List is mutable i.e we can make any changes in list.	Tuple is immutable i.e we can not make any changes in tuple		Dictionary is mutable. But Keys are not duplicated.
List is ordered	Tuple is ordered	Set is unordered	Dictionary is ordered
Creating a set a=set()			
Creating an empty list l=[]	Creating an empty Tuple t=()		
		b=set(a)	

Code:

```
# List in Python

# List is a collection which is ordered and changeable. Allows duplicate members.

fruits = ["apple", "mango", "banana", "orange"] print("The initial list is : ")

print(fruits) fruits.append("pineapple") fruits.remove("apple")

fruits.insert(2,"cherry") print("The updated list is : ") print(fruits) print("\n")


# Tuple in Python

# Tuple is a collection which is ordered and unchangeable. Allows duplicate members.


vehicles = ("car", "bike", "truck", "tracktor")

print("The initial tuple is : ") print(vehicles)

print("\n")



# Dictionary
# Dictionary is a collection which is ordered** and changeable. No duplicate members.

students = {29:"Prathamesh", 30:"MDG", 31:"KP", 32:"Devil"} print("The initial

dictionary is ") print(students) students[28] = "OM" students.pop(32) print("The

updated dictionary is ") print(students) print("\n")


# Set
```

Set is a collection which is unordered, unchangeable*, and unindexed. No duplicate members.

flowers = {"lotus", "gulab", "sunflower"}

print("The initial set is : ") print(flowers)

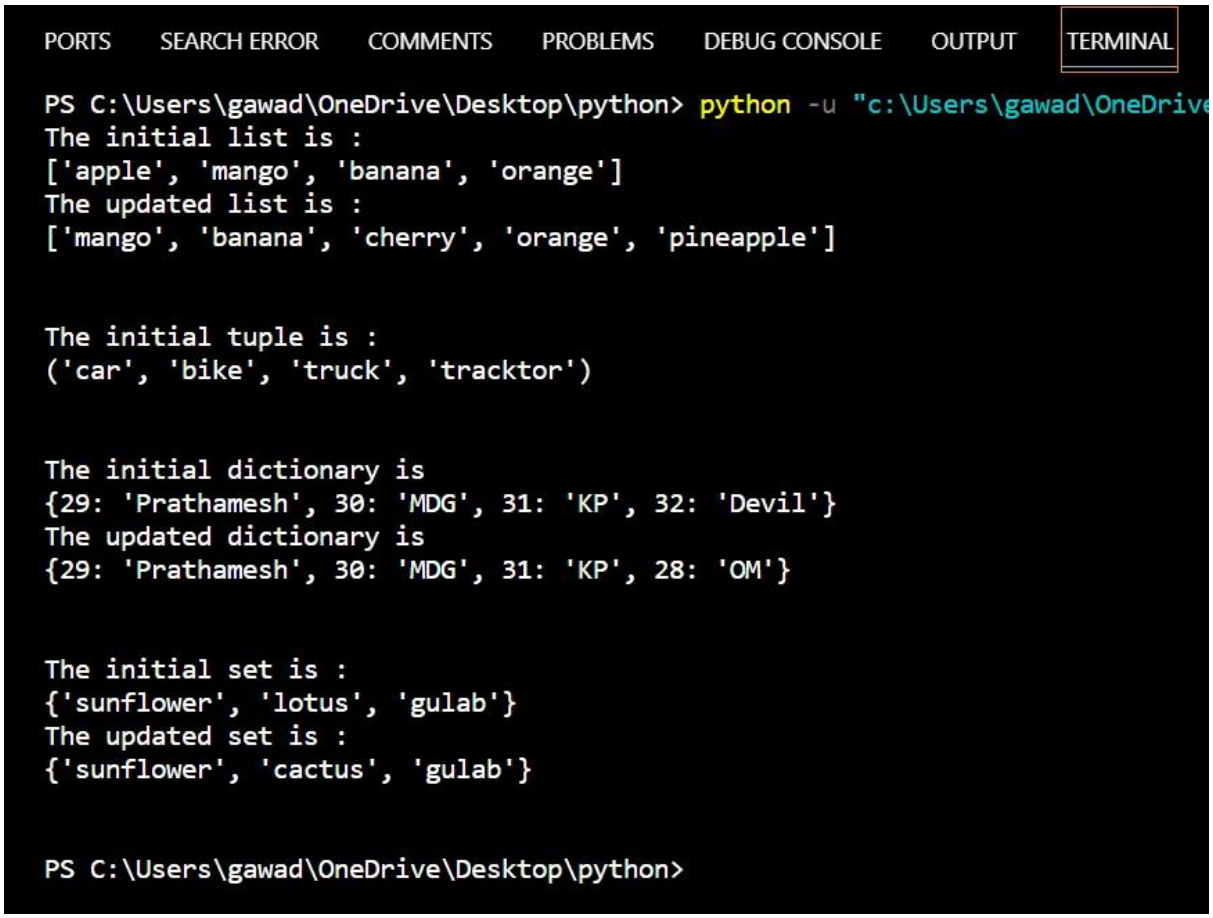
flowers.add("cactus")

flowers.remove("lotus") print("The

updated set is : ") print(flowers)

print("\n")

Output:


PORTS SEARCH ERROR COMMENTS PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL
PS C:\Users\gawad\OneDrive\Desktop\python> python -u "c:\Users\gawad\OneDrive
The initial list is :
['apple', 'mango', 'banana', 'orange']
The updated list is :
['mango', 'banana', 'cherry', 'orange', 'pineapple']
The initial tuple is :
('car', 'bike', 'truck', 'tracktor')
The initial dictionary is
{29: 'Prathamesh', 30: 'MDG', 31: 'KP', 32: 'Devil'}
The updated dictionary is
{29: 'Prathamesh', 30: 'MDG', 31: 'KP', 28: 'OM'}
The initial set is :
{'sunflower', 'lotus', 'gulab'}
The updated set is :
{'sunflower', 'cactus', 'gulab'}
PS C:\Users\gawad\OneDrive\Desktop\python>

Conclusion:

The provided Python code demonstrates the usage of various data structures: lists, tuples, dictionaries, and sets. Lists are ordered collections that allow duplicates and support operations like appending, removing, and inserting elements. Tuples, on the other hand, are similar to lists but immutable, meaning they cannot be changed after creation. Dictionaries are key-value pairs where keys are unique and mutable, allowing for easy access and modification. Sets are unordered collections of unique elements, useful for operations like adding and removing items efficiently. Overall, these data structures offer versatile ways to organize and manipulate data in Python, catering to different needs and scenarios with their distinct characteristics and functionalities.