NAME:KUMUD MOHAN ZIMAL CLASS:SE COMPS-3 ROLL NO-65

| Experiment No. 3 |
| --- |
| To explore basic data types of python like strings, list,  dictionaries and tuples |
| Date of Performance: 23/1/2024 |
| Date of Submission: 30/1/2024 |

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 a

non-homogeneous

data structure which

stores key value

pairs

List can be

represented by [ ]

Tuple can be

represented by

( ) Set can be

represented by { }

Dictionary can be

represented by { }

List allows

duplicate elements

Tuple allows

duplicate elements

Set will not allow

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, 5]

Example: (1, 2, 3, 4,

5)

Example: {1, 2, 3,

4, 5}

Example: {1, 2, 3, 4,

5}

List can be created

using list() function

Tuple can be created

using tuple() function.

Set can be created

using set() function

Dictionary can be

created

using dict() function.

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

Set is mutable i.e

we can make any

changes in set. But

elements are not

duplicated.

Dictionary is

mutable. But Keys

are not duplicated.

List is ordered Tuple is ordered Set is unordered Dictionary is ordered

Creating an empty

list

l=[]

Creating an empty

Tuple

t=()

Creating a set

a=set()

b=set(a)

CODE:

list=[1,2,3,4,5]

list.append(6)

print(list)

list.remove(1)

print(list)

list.pop() # index se accesible hai

print(list)

list.append("kumud")

print(list)

tuple1=("hi",1,2,3,"bye")

print(tuple1)

print(tuple1[2])

sets={1,3,5,"hii",9.8,3,5,5,1} #error nahi dega par element ko ek baar hi print karega

print(sets)

sets.remove(3) #append nahi chal raha sets me

sets.pop()

sets.pop()

print(sets)

DICTIONARY={"kumud":"zimal","aparna":"shelke"}

print(DICTIONARY)

DICTIONARY={1:"zimal",2:"shelke"}

print(DICTIONARY)

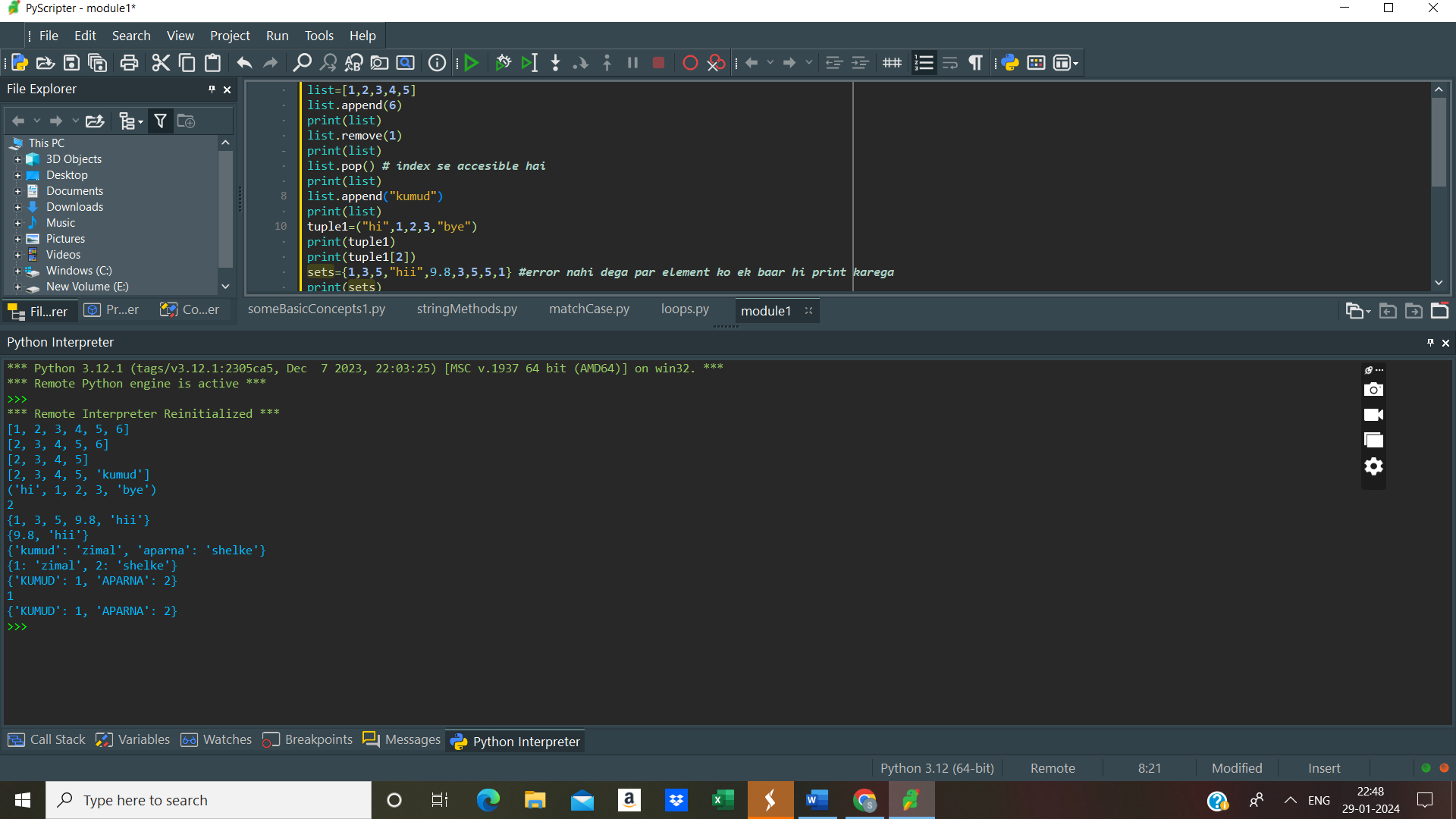
DICTIONARY={"KUMUD":1,"APARNA":2}

print(DICTIONARY)

print(DICTIONARY["KUMUD"])

print(DICTIONARY)

OUTPUT:



Conclusion:

- Understanding the basic data types in Python—strings, lists, dictionaries, and tuples—is essential for effective programming.

- Each data type has its unique properties, behaviors, and use cases, making them suitable for different scenarios.

- Choosing the appropriate data type based on the requirements of your program can lead to more efficient and readable code.

- Mastery of these basic data types is fundamental to becoming proficient in Python programming and building more complex data structures and algorithms.