Don Bosco Institute of Technology

Department of Electronics and Telecommunication Engg.
SUB: Skill Lab (ECL404)

Python Data Types and Data Structures

Expt No:	**01**
Aim:	To work with data types and data structures in Python.
Tool used:	Anaconda Navigator+ Jupiter Notebook
Theory:	Answer in brief (not more than 6 to7 sentences.) Resource: https://www.w3schools.com/python/

1. What are strings and how is it declared in python?

A: Strings are sequences of characters, and they are used to represent text in Python.

Strings can be declared using single (' '), double (" ") quotes.

2. What are lists? Discuss the functions of methods append, pop, sort, delete methods.

A: Lists are ordered, mutable (changeable), and can contain elements of different data types.

Functions of list methods:

append(element): Adds the specified element to the end of the list.

pop(index): Removes and returns the element at the specified index. If no index is provided, it removes and returns the last element.

sort(): It sorts the elements of the list in ascending order.

remove(element): Removes the first occurrence of the specified element from the list

3. Elaborate on what is a dictionary.

A: A dictionary is an unordered collection of key-value pairs.

Each key must be unique, and it maps to a specific value.

Ex: My_dict = {"IT": "100", "EXTC": "120", "COMP": "200"}

4. What is a tuple? Elaborate on index and count methods.

A: A tuple is an ordered, immutable collection of elements.

index(element): Returns the index of the first occurrence of the specified element.

count(element): Returns the number of occurrences of the specified element in the tuple.

5. How is set different than a normal lists?

A: A set is an unordered and unindexed collection of unique elements.

- Sets do not allow duplicate elements.
- Unlike lists, sets do not support indexing or slicing since they are unordered.
- Sets are useful for performing mathematical set operations like union, intersection, and difference

Tasks and outputs:

1. Declare values of two variables 'income' and 'rate'. Compute $taxes = income * rate$. Display the result.

```
income = int(input("Enter your income: "))
print(income)
rate = float(input("Enter rate: "))
print(rate)
taxes = int(income * rate)
print(taxes)
```

```
Enter your income: 50000
50000
Enter rate: 0.05
0.05
2500
```

2. Declare your Sem I SGPA value and Sem II SGPA value. Compute $CGPA = (SGPA1 * SGPA2)/2$. Display the result.

```
SGPA1 = float(input("Enter your SGPA of SEM 1: "))
print(SGPA1)
SGPA2 = float(input("Enter your SGPA of SEM 2: "))
print(SGPA2)
CGPA = int(SGPA1 + SGPA2)/2
print(CGPA)
```

```
Enter your SGPA of SEM 1: 8.05
8.05
Enter your SGPA of SEM 2: 7.95
7.95
8.0
```

3. Declare a string s as your full name.
   a. Demonstrate how to select and display single letters from the string.
   b. Display the last and third-last letter using reverse indexing.
   c. Reverse your name.
   d. Demonstrate the use of upper, lower and split method

```
NAME="Aditya Punekar"
print(NAME[0])
print(NAME[-9])
print(NAME[::-1])
print(NAME.upper())
print(NAME.lower())
print(NAME.split())
```

```
A
a
rakenuP aytidA
ADITYA PUNEKAR
aditya punekar
['Aditya', 'Punekar']
```

4. Declare a string consisting of 12-15 characters.
   a. Use slicing to display the middle, subsection, first and last subsection separately.

```
NAME="Aditya Varsha Ramesh Punekar"
slice_obj1 = slice(11,17)
print(NAME[slice_obj1])
slice_obj2 = slice(10,18)
print(NAME[slice_obj2])
slice_obj3 = slice(6,12)
print(NAME[slice_obj3])
slice_obj4 = slice(0,-5)
print(NAME[slice_obj4])
```

```
ha Ram
sha Rame
 Varsh
Aditya Varsha Ramesh Pu
```

5. Declare two lists and concatenate them. Append a few values to this new list. Use pop() method to remove some values. Sort the lists.

```
List1 = [15, 13, 12, 14, 11]
print("\nList of Numbers: ")
print(List)
List2 = [7, 8, 9, 10, 6]
Total_List = List1 + List2
print ("Concatenated list using: " + str(Total_List))
List3 = [1, 2, 4, 5, 3]
New_Total_List = Total_List + List3
print ("New Concatenated list using: " + str(New_Total_List))
Popped_Element = int(New_Total_List.pop(1))
print("Popped Element:", Popped_Element )
List1.sort()
List2.sort()
List3.sort()
print(List1)
print(List2)
print(List3)
```

```
List of Numbers:
[1, 2, 3, 4, 5, 6, 7, 8, 9]
Concatenated list using: [15, 13, 12, 14, 11, 7, 8, 9, 10, 6]
New Concatenated list using: [15, 13, 12, 14, 11, 7, 8, 9, 10, 6, 1, 2, 4, 5
Popped Element: 13
[11, 12, 13, 14, 15]
[6, 7, 8, 9, 10]
[1, 2, 3, 4, 5]
```

6. Create a dictionary of fruits and its corresponding rates. Display the keys, values and items separately.
   a. Add two new values.
   b. Delete the most recent entered value.

```
Dict1 = {'Apple':25, 'Banana':35, 'Coconut':15}
print(Dict1)
Dict2 = {'Mango':50}
Dict1.update(Dict2)
print(Dict1)
Dict1.keys()
Dict1.values()
print(Dict1)
Popped_Name = int(Dict1.pop('Banana'))
print(Popped_Name)
```

```
{'Apple': 25, 'Banana': 35, 'Coconut': 15}
{'Apple': 25, 'Banana': 35, 'Coconut': 15, 'Mango': 50}
{'Apple': 25, 'Banana': 35, 'Coconut': 15, 'Mango': 50}
35
```

‘

7. Declare a tuple of alphabets.
   a. Find the count of repeated occurring alphabets.
   b. Demonstrate use of index() method.

```
tuple = ('a','b','c','d','f','f','g','h','i','j')
count = tuple.count('f')
print(count)
print(tuple[6])
```

```
2
g
```

8. a. Declare an empty set.

   b. Add two new values to this set.

```
my_set = set()
print(my_set)
my_set.add("ramesh")
my_set.add("punekar")
print(my_set)
```

```
set()
{'punekar', 'ramesh'}
```

Observation and Conclusion:	Write your understanding out all methods that you have tried. • In the above experiment, various methods were employed to manipulate and interact with the chosen elements. • For strings, we focused on declaration methods using single, double. • Lists, being mutable, were enriched by methods like append for adding elements, pop for removal, sort for ordering, and remove for deleting specific items. • Dictionaries, as key-value pairs, demonstrated their utility in efficient data organization, with quick retrieval and modification capabilities. • Tuples, being immutable, featured index and count methods for locating elements and determining their occurrences within the sequence. • Sets, distinct for their uniqueness and unordered nature, offered methods such as union, intersection, and difference for performing mathematical operations on sets.