Experiment No : 1

Dated : 13th Jan 2022

Aim

Exploring basics of Python - Data types and Control Statements.

Theory

- Basic Data Types in Python (enlist 5 functions for each)
- Numbers, Boolean, Strings
- Built-in Containers in Python (enlist 5 functions for each)
- Lists, Dictionaries, Sets, Tuples.
- Looping & Control Statements in Python (enlist them)
- Looping statements - While, For, nested loops
- Control statements - continue, break & pass
- Arrays in Python (enlist 5 functions)

Handwritten Theory:

EXPERIMENT 1

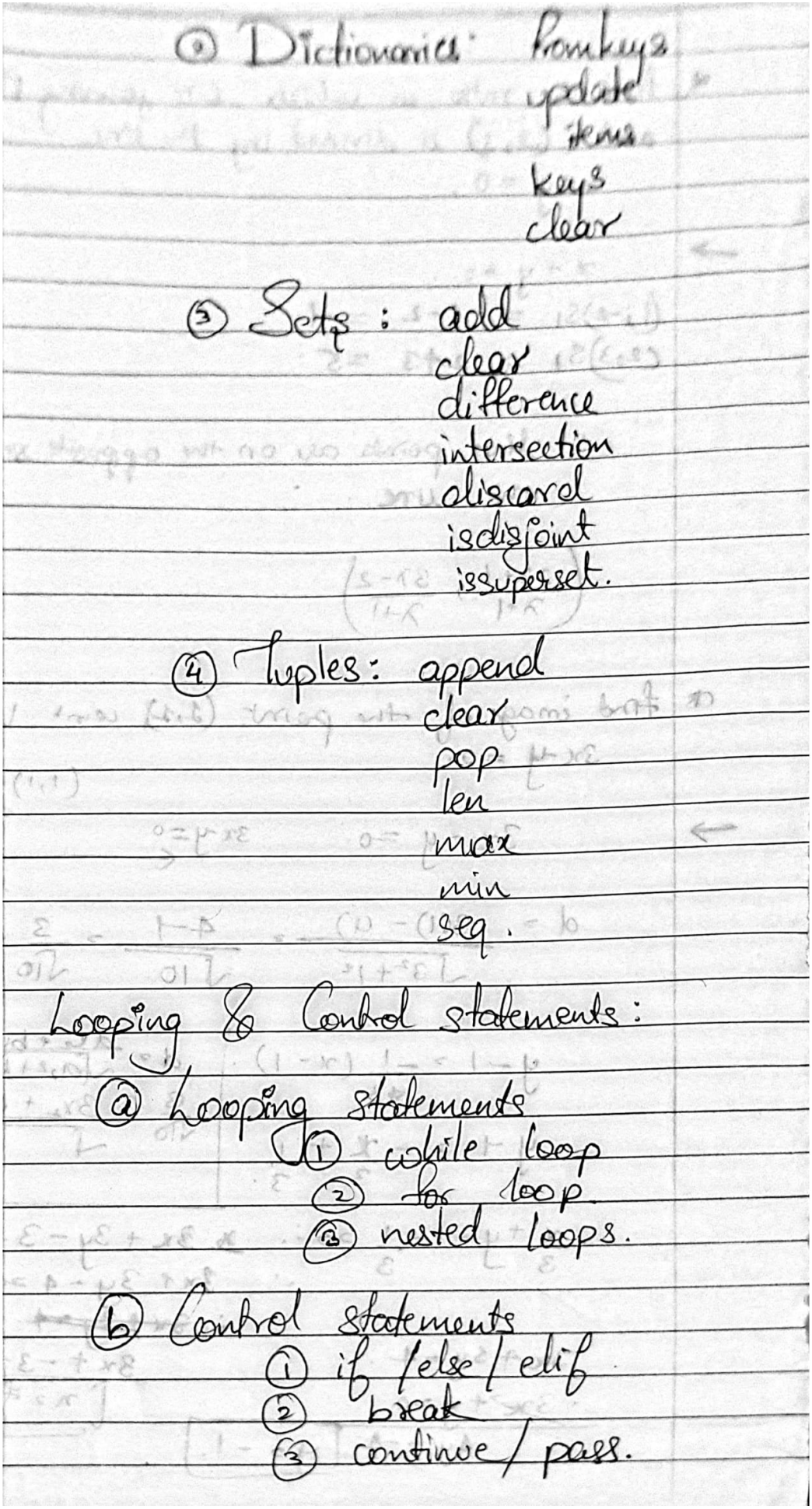
Aim: Explaining basics of Python -Data types & Control statements.

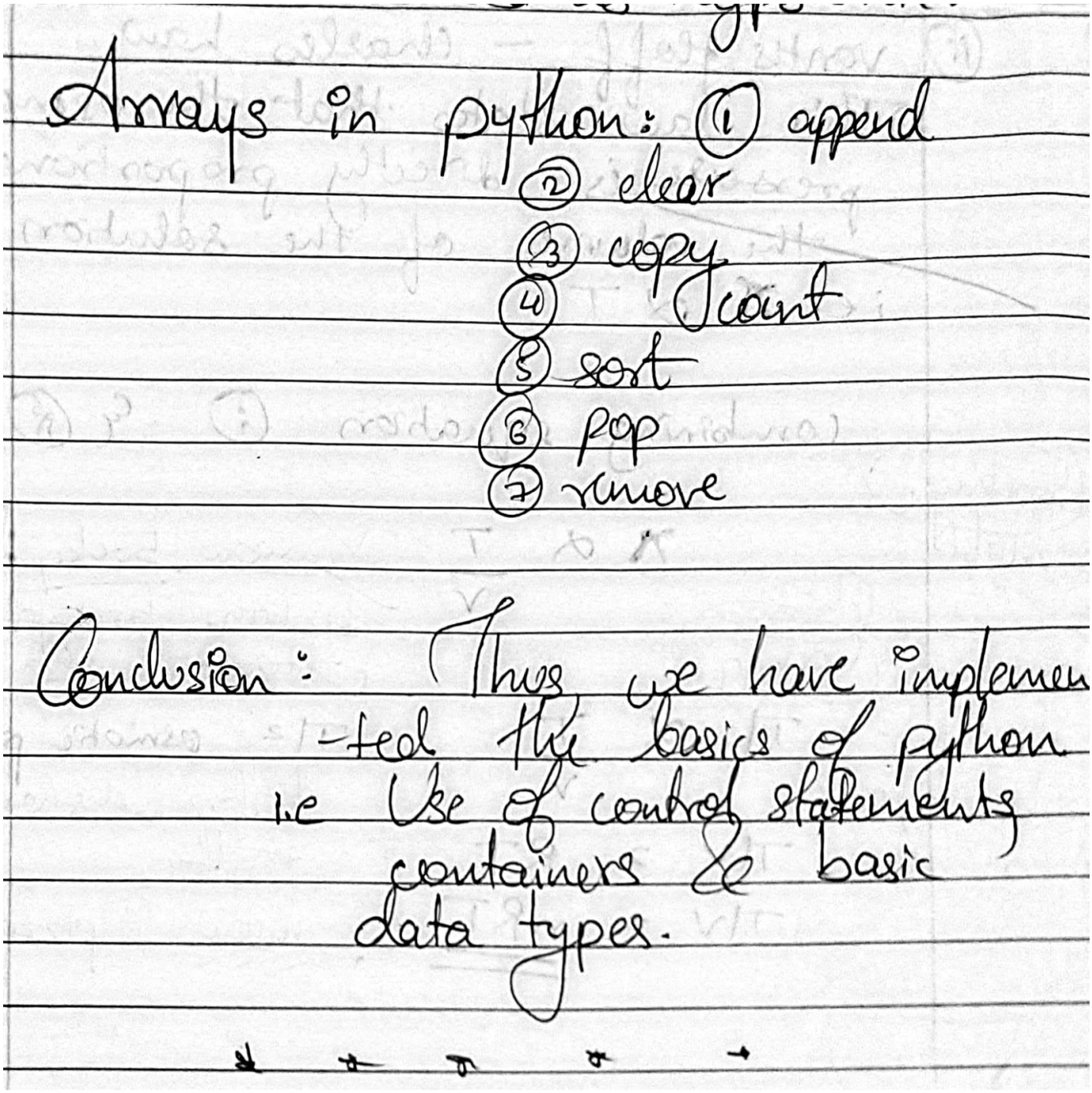
Theory: Basic data types in Python:

1. Numeric: int, float, complex.
2. Boolean: bool
3. Strings: functions- capitalize casefold center count endswith.

Built in containers in Python.

1. Lists: append pop insert remove clear reverse sort.


② Dictionaries: fromkeys
update
items
keys
clear
③ Sets: add
clear
difference
intersection
discard
isdisjoint
issuperset.
④ Tuples: append
clear
pop
len
max
min
seq.
Looping & Control statements:
ⓐ Looping statements
① while loop
② for loop.
③ nested loops.
ⓑ Control statements
① if / else / elif
② break
③ continue / pass.


Arrays in python: ① append
② clear
③ copy
④ count
⑤ sort
⑥ pop
⑦ remove
Conclusion: Thus we have implemen
-ted the basics of python
i.e Use of control statements
containers & basic
data types.

Programs to be performed

1.

- a) To swap two numbers
- b) check if the first number is positive or negative or zero.

```
#1. a) To swap two numbers
num1 = input("Input the first number: ")
num2 = input("Input the second number: ")
temp = num1
num1 = num2
num2 = temp
print(num1 + ' ' + num2)
```

```
Input the first number: 69
Input the second number: 420
420 69
```

```
#1. b) check if the first number is positive or negative or zero.
def int_nature(any_number):
  if int(any_number) == 0:
    return 'zero'
  elif int(any_number) < 0:
    return 'negative'
  elif int(any_number) > 0:
    return 'positive'
  else:
    return 'Please pass an integer variable to check'

print(int_nature(num1))
```

```
positive
```

2.

- a) To check whether the entered string is palindrome
- b) Find the factorial of the input number.

```
#2. a) To check whether the entered string is palindrome.
random_string = input("Input the string to check for palindrome: ")
def is_palindrome(any_string):
  return_string = any_string.lower()
  temp = list(reversed(return_string))
  if temp == list(return_string):
    return 'is a palindrome'
  else :
    return 'is not a palindrome'

print("The string input given " + is_palindrome(random_string))
```

```
Input the string to check for palindrome: racecar
The string input given is a palindrome
```

```
#2. b) Find the factorial of the input number.
import math
#help(math)
num_for_fact = input("Input the number to find it's factorial: ")
fact_of_num = math.factorial(int(num_for_fact))
print("Factorial of the input number " + num_for_fact + " is " , fact_of_num)
```

```
Input the number to find it's factorial: 8
Factorial of the input number 8 is  40320
```

3. Perform the following operations using Lists:

- a) Separate even and odd nos from the list
- b) Merge and sort the two lists
- c) Update the first element with x value and delete the middle element of the list.
- d) Find the minimum and maximum element from the list.
- e) Add n names into the existing list and check if the word "python" is present in the list

```
#3. a) Separate even and odd nos from the list

my_list = [x for x in range(10)]
my_odd_list, my_even_list = [],[]
print(my_list)

for num in my_list:
  if num % 2 ==0:
    my_even_list.append(num)
  else:
    my_odd_list.append(num)

print(my_even_list, my_odd_list)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 2, 4, 6, 8] [1, 3, 5, 7, 9]
```

```
#b) Merge and sort the two lists

merged_list = my_even_list + my_odd_list #Merging
print(merged_list)

merged_list.sort()                       #Sorting
print(merged_list)
```

```
[0, 2, 4, 6, 8, 1, 3, 5, 7, 9]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
#c) Update the first element with x value and delete the middle element of the list.

merged_list[0] = 69                          #Updating first element to 69

merged_list.pop(int((len(merged_list))/2))  #Deleting(Popping) the middle element
print(merged_list)
```

```
[69, 1, 2, 3, 4, 6, 7, 8, 9]
```

```
#d) Find the minimum and maximum element from the list.
def min_max_list(any_list):
  min = max = any_list[0]
  for item in any_list:
    if item > max : max = item
    if item < min : min = item
  return(min , max)

print("The minimum and maximum element from the list ", merged_list, " are ", min_max_l
```

```
The minimum and maximum element from the list  [69, 1, 2, 3, 4, 6, 7, 8, 9]  are
```

```
#e) Add n names into the existing list and check if the word "python" is present in the
new_list = merged_list
new_list.append('java')
new_list.append('c++')
new_list.append('python')
new_list.append('javascript')
print(new_list)

print(('python' in new_list))
```

```
[69, 1, 2, 3, 4, 6, 7, 8, 9, 'java', 'c++', 'python', 'javascript']
True
```

4. Perform the following operations using Tuples:

- a) Create a Tuple to store Student details (rno, name , subjects marks, total)
- b) Display the details
- c) Sort the tuples wrt to total

```
# 4. Perform the following operations using Tuples:
# a) Create a Tuple to store Student details (rno, name , subjects marks, total)
student1=("Roll No. 47", "Name: Yash Sarang", 69 , 420)
student2=("Roll No. 48", "Name: IDK", 70 , 411)
student3=("Roll No. 49", "Name: Hmm", 68 , 414)
student4=("Roll No. 45", "Name: Hehe", 96 , 96)

students=[student1,student2,student3,student4]

#b) Display the details
```

```
print(students)

#c) Sort the tuples wrt to total
def sort_tuple_list(any_tuple,index_no):
  for x_times in any_tuple :
    base_tuple = any_tuple[0]
    for idx,tpl in enumerate(any_tuple):
      if (any_tuple[idx])[index_no] < base_tuple[index_no] :
        temp = any_tuple[idx-1]
        any_tuple[idx-1] = any_tuple[idx]
        any_tuple[idx] = temp
      base_tuple = any_tuple[idx]
  return any_tuple


print(sort_tuple_list(students,3))
print((students[1])[3])                #accessing a tuple item in a list of tuples
```

```
[('Roll No. 47', 'Name: Yash Sarang', 69, 420), ('Roll No. 48', 'Name: IDK', 70, 
[('Roll No. 45', 'Name: Hehe', 96, 96), ('Roll No. 48', 'Name: IDK', 70, 411), ('
411
```

5. Perform the following operations using Sets:

- a) Accept two strings using variable declarations.
- b) Display the common letters.
- c) Display letters present only in the first string
- d) Display all letters of both string
- e) Display letters which are not common in both string. (Symmetric Difference)

```
# a) Accept two strings using variable declarations.
first_name, last_name = (input("Enter your Full Name : ").lower().split())
```

```
Enter your Full Name : Yash Sarang
```

```
# b) Display the common letters.
print("Common letters between the two strings are:")
for letter in set(first_name):
  if letter in set(last_name):
    print(letter)
```

```
Common letters between the two strings are:
s
a
```

```
# c) Display letters present only in the first string
print("Letters not present between the two strings are (only in the first string):")
for letter in set(first_name):
```

```
  if letter not in set(last_name):
    print(letter)
```

```
Letters not present between the two strings are (only in the first string):
h
y
```

```
# d) Display all letters of both string
for item in set(first_name + last_name):
  print(item)
```

```
h
s
g
n
y
r
a
```

```
# e) Display letters which are not common in both string. (Symmetric Difference)
print("Letters which are not common in both string. (Symmetric Difference):")
for item in set(first_name)^set(last_name):
  print(item)
```

```
Letters which are not common in both string. (Symmetric Difference):
h
n
y
r
g
```

6. Create a dictionary to perform the following operations:

- a) Update, concatenate, delete
- b) Search a key
- c) Mapping two list into dictionary

```
#Create
data = {'Sarang' : 47, 'Manav' : 37, 'Om' : 58}
print(data)
```

```
{'Sarang': 47, 'Manav': 37, 'Om': 58}
```

```
#Update
data['Om'] = 15
print(data)
```

```
{'Sarang': 47, 'Manav': 37, 'Om': 15}
```

```
#Concatenate
data['Madhu'] = 36
print(data)
```

```
{'Sarang': 47, 'Manav': 37, 'Om': 15, 'Madhu': 36}
```

```
#Delete
data.pop('Madhu')
print(data)
```

```
{'Sarang': 47, 'Manav': 37, 'Om': 15}
```

```
#b) Search a key
if 'Om' in data.keys():
    print("Key exists.")
else:
    print("Key doesn't exist.")
```

```
Key exists.
```

```
#c) Mapping two list into dictionary
alphabets = ['A', 'B', 'C', 'D']
numbers = [1,2,3,4]
myDict = {k: v for k, v in zip(alphabets,numbers)}
print("Mapping two list into dictionary: ",myDict)
```

```
Mapping two list into dictionary:  {'A': 1, 'B': 2, 'C': 3, 'D': 4}
```

7. To search a given element in the Array.

```
int_array = [0,1,2,3,4,5]
print(type(int_array), int_array)
print(int_array.index(3))
```

```
<class 'list'> [0, 1, 2, 3, 4, 5]
3
```