Green University of Bangladesh
Department of Computer Science and Engineering(CSE)

Faculty of Sciences and Engineering

Semester: (Spring , Year:2024), B.Sc. in CSE (Day)

LAB REPORT NO #01

Course Title: Artificial Intelligence Lab

Course Code: CSE - 316 **Section: 213_D5**

Lab Experiment Name: Introduction to Basic Operations on Python.

Student Details

Name	ID
MD Dulal Hossain	213902116

Lab Date : **27– 02 - 2024**
Submission Date : **05– 03 - 2023**

Course Teacher's Name : **Sagufta Sabah Nakshi**

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: …… **Signature:.....**
Comments:..... **Date:.....**

1. TITLE OF THE LAB EXPERIMENT

- To acquire knowledge about python.To learn variables , operators , conditional statements, loops & functions in python also to learn Basic Operations on Python such as Lists, Tuple, Dictionary, Numpy, Pandas, Matplotlib.

2. OBJECTIVES/AIM

- **Code 1** : Write a python program to find the largest num between two num using function
- **Code 2** : Write a python program to find the sum of the numbers passed as parameters.
- **Code 3** : Write a Python program to get the 4th element from the beginning and the 4th element from the last of a tuple.
- **Code 4** : Write a Python Program to Count Even and Odd Numbers in a list.

3. PROCEDURE / ANALYSIS / DESIGN

Algorithm For Code 1 :

1. Define a function find_largest that compares two numbers and returns the larger one.
2. Take user input for two numbers.
3. Call find_largest with the user-input numbers.
4. Print the largest number between the two inputs.
5. End.

Algorithm For Code 2 :

1. Define a function sum_of_numbers that calculates the sum of any number of arguments.
2. Initialize a variable total to 0.
3. Iterate through each argument and add it to total.
4. Return the value of total.
5. Call sum_of_numbers with arguments (10, 20, 30, 40).
6. Print the sum of the numbers.
7. End.

Algorithm For Code 3 :

1. Define a function count_even_odd to count even and odd numbers in a list.
2. Initialize even_count and odd_count to 0.
3. Iterate through the list, incrementing either even_count or odd_count based on the number's parity.
4. Return the counts.
5. Provide a list of numbers.
6. Call the function with the list.
7. Print the counts of even and odd numbers.
8. End.

Algorithm For Code 4 :
1. Provide a tuple named tuplex. 2. Access the fourth element from the beginning of the tuple and store it in the variable fourth_from_beginning. 3. Access the fourth element from the end of the tuple and store it in the variable fourth_from_end. 4. Print both fourth_from_beginning and fourth_from_end. 5. End.

4. IMPLEMENTATION

Code 1 :

```
def find_largest(num1, num2):

    if num1 > num2:
        return num1
    else:
        return num2

num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number:
"))

largest_number = find_largest(num1, num2)

print("The largest number between", num1,
"and", num2, "is", largest_number)
```

Code 2 :

```
def sum_of_numbers(*args):

    total = 0
    for num in args:
        total += num
    return total
result = sum_of_numbers(21, 39, 21, 16)

print("Sum of the numbers:", result)
```

Code 3 :

```
def count_even_odd(numbers):
    even_count = 0
    odd_count = 0
    for num in numbers:
        if num % 2 == 0:
            even_count += 1
        else:
            odd_count += 1
    return even_count, odd_count
numbers_list = [2, 1, 3, 9, 0, 2, 1, 1, 6]
even_count, odd_count =
count_even_odd(numbers_list)
print("Number of even numbers:",
even_count)
print("Number of odd numbers:", odd_count)
```

Code 4 :

```
tuplex = ("w", 3, "r", "e", "s", "o", "u", "r",
"c", "e")

fourth_from_beginning = tuplex[3]

fourth_from_end = tuplex[-4]

print(fourth_from_beginning,
fourth_from_end)
```

5. TEST RESULT / OUTPUT

Code 1 :

```
PS C:\Users\USER> & C:/Users/USER/AppData
/Local/Microsoft/WindowsApps/python3.12.e
xe "e:/07 Seven Semester/CSE 316-Artifici
al Intelligence Lab_/lab report 1/Write a
 python program to find the largest numbe
r between two numbers using function.py"
Enter the first number: 2139
Enter the second number: 2116
The largest number between 2139 and 2116
is 2139
```

Figure 1 : Output Successfully of Code 1.

Code 3 :

```
PS C:\Users\USER> & C:/Users/USER/AppData
/Local/Microsoft/WindowsApps/python3.12.e
xe "e:/07 Seven Semester/CSE 316-Artifici
al Intelligence Lab_/lab report 1/Write a
 Python Program to Count Even and Odd Num
bers in a list.py"
Number of even numbers: 4
Number of odd numbers: 5
```

Figure 3 : Output Successfully of Code 3.

Code 2 :

```
PS C:\Users\USER> & C:/Users/USER/AppData
/Local/Microsoft/WindowsApps/python3.12.e
xe "e:/07 Seven Semester/CSE 316-Artifici
al Intelligence Lab_/lab report 1/Write a
 python program to find the sum of the nu
mbers passed as parameters.py"
Sum of the numbers: 97
```

Figure 2 : Output Successfully of Code 2.

Code 4 :

```
PS C:\Users\USER> & C:/Users/USER/AppData
/Local/Microsoft/WindowsApps/python3.12.e
xe "e:/07 Seven Semester/CSE 316-Artifici
al Intelligence Lab_/lab report 1/Write a
 Python program to get the 4th element fr
om the beginning and the 4th element from
 the last of.py"
e u
```

Figure 4 : Output Successfully of Code 4.

6. ANALYSIS AND DISCUSSION

For code 1 :

This code efficiently determines the larger of two user-inputted numbers using a conditional statement within a function. It enables dynamic comparison of any two numbers provided by the user, offering a simple yet effective way to find the maximum value between them.

For code 2 :

This code defines a flexible function, sum_of_numbers, which computes the sum of any number of input arguments. It effectively employs iteration to accumulate the total and returns it. The example call demonstrates its usage with specific values, showcasing its versatility in handling varying input sizes.

For code 3 :

This code efficiently defines a function, count_even_odd, to tally even and odd numbers in a list using iteration. It accurately tracks counts based on the number's parity and returns them. The example demonstrates its practical usage, enhancing understanding and utility in counting even and odd occurrences.

For code 4 :

This code instantiates a tuple named tuplex, accessing its fourth element from both the beginning and the end. This demonstrates the flexibility of tuple indexing in Python. Printing both elements illustrates successful retrieval, highlighting tuple's bidirectional indexing capability.