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

Faculty of Sciences and Engineering

Semester: (Spring, Year: 2025), B.Sc. in CSE (Day)

LAB REPORT NO 01

Course Title: Artificial IntelligenceLab

Course Code: CSE-316 Section: 222-D6

Student Details

Name		ID
1.	MD FAHIM SARKER MRIDUL	221902369

Submission Date : 18-02-2025

Course Teacher's Name : Kazi Hasnayeen Emad

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

Lab Report Status

Marks : …… **Signature :.....**

Comments:..... **Date :.....**

Report Title: Basic Python Problems and Solutions.

Objectives:

i.To provide fundamental Python problem-solving practice.
ii.To enhance logical thinking and coding skills.
iii.To cover key Python concepts like loops, functions, lists, and dictionaries.

Implementation:

1. Take an integer input and print it.

   Code:

```
[1]: #Problem-1:Take an integer input and print it.
     A=int(input("give an input"))
     print(f"The taken input is {A}")
```

```
give an input 55
The taken input is 55
```

2. Take two integer inputs and print their sum.

   Code:

```
[2]: #problem-2:Take two integer inputs and print their sum.
     A=int(input())
     B=int(input())
     Sum=A+B
     print(f"thew summation is {Sum}")
```

```
 5
 6
thew summation is 11
```

3. Take a string input and print it.

   Code:

```
[3]: #problem-3:Take a string input and print it.
     Char=input()
     print(f"{Char} is a string")
```

```
 Aam Khabo
Aam Khabo is a string
```

4. Take a float input and print its square.

   Code:

```
[4]: #problem-4:Take a float input and print its square.
     rational=float(input())
     square=rational*rational
     print(square)
```

```
 5.69
32.3761
```

5. Take a name as input and print "Hello, [name]!".

   Code:

```
[5]: #problem-5:Take a name as input and print "Hello, [name]!".
     name=input()
     print(f"hello, {name}")
```

```
 Noyon
hello, Noyon
```

2. Arithmetic Operations

6. Add two numbers.

   Code:

```
#problem-6:Add two numbers.
num1=int(input())
num2=int(input())
print(num1+num2)
```

```
 6
 16
22
```

+ Code + Markdown

7. Subtract two numbers.

   Code:

```
[4]: #problem-7:Substract two numbers.
num1=int(input())
num2=int(input())
print(num1-num2)
```

```
 58
 6
52
```

8. Multiply two numbers.

   Code:

```
[5]: #problem-8:Multiply two numbers.
num1=int(input())
num2=int(input())
print(num1*num2)
```

```
 5
 9
45
```

9. Divide two numbers (return float).

   Code:

```
[6]: #problem-9:Divide two numbers.
     num1=int(input())
     num2=int(input())
     div = num1/num2
     print(float(div))
```

```
15
5
3.0
```

10. Find the remainder when dividing two numbers.

    Code:

```
[1]: #Find the remainder when dividing two numbers.
     num1=int(input())
     num2=int(input())
     Rem = num1%num2
     print(Rem)
```

```
18
5
3
```

3. Conditional Statements

11. Check if a number is even or odd.

    Code:

[2]:
```python
#Check if a number is even or odd.
N=int(input())
if N%2==0:
    print("Even")
else:
    print("Odd")
```

```
5
Odd
```

12. Check if a number is positive, negative, or zero.

    Code:

[5]:
```python
#Check if a number is positive, negative, or zero.
N=int(input())
if N>0:
    print("Positive")
elif N<0:
    print("Negative")
else:
    print("Zero")
```

```
0
Zero
```

13. Find the maximum of two numbers.

    Code:

[6]:
```python
#Find the maximum of two numbers.
A=int(input())
B=int(input())
if A>B:
    print(f"{A} is grater than {B}")
elif B>A:
    print(f"{B} is grater than {A}")
else:
    print("both are equal")
```

```
5
6
6 is grater than 5
```

14. Find a minimum of three numbers.

Code:

[9]:
```python
#Find the minimum of three numbers.
A=int(input())
B=int(input())
C=int(input())
if A>B>C:
    print(f"{A} is grater than {B} and {C}")
elif B>A>C:
    print(f"{B} is grater than {A} and {C}")
elif C>B>A:
    print(f"{C} is grater than {A} and {B}")
```

```
5
4
9
```

15. Check if a number is a multiple of 5.

Code:

[10]:
```
#Check if a number is a multiple of 5.
N=int(input())
if N%5==0:
    print(f"{N} is multiple of 5")
```

```
 50
50 is multiple of 5
```

4. Loops

16. Print numbers from 1 to 10 using a for loop.

    Code:

[12]:
```
#Print numbers from 1 to 10 using a for loop.
for i in range(1,11):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
```

17. Print even numbers from 1 to 20.

    Code:

[18]:
```
#Print even numbers from 1 to 20.
for i in range(1,21):
    if i%2==0:
        print(i)
```

```
2
4
6
8
10
12
14
16
18
20
```

18. Print odd numbers from 1 to 20.

Code:

[15]:
```
#Print odd numbers from 1 to 20.
for i in range(1,21,2):
    print(i)
```

```
1
3
5
7
9
11
13
15
17
19
```

19. Print numbers from 10 to 1 in reverse.

Code:

[23]:
```python
#Print numbers from 10 to 1 in reverse.
for i in range(10,0,-1):
    print(i)
```

```
10
9
8
7
6
5
4
3
2
1
```

20. Print the multiplication table of a given number.

Code:

[24]:
```python
#Print the multiplication table of a given number.
N=int(input())
for i in range(1,11):
    print(f"{N} * {i} = {N*i}")
```

```
 5
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50
```

5. Lists

21. Create a list with 5 elements and print it.

Code:

[25]:
```
#Create a list with 5 elements and print it.
list = ["mango","banana","kamranga","guava","apple"]
print(list)
```

```
['mango', 'banana', 'kamranga', 'guava', 'apple']
```

22. Find the sum of all elements in a list.

Code:

[2]:
```
num1 = [10, 20, 30, 40, 50]
print(sum(num1))
```

```
150
```

23. Find the largest element in a list.

Code:

[3]:
```
num = [10, 25, 32, 45, 8]
print("largest:", max(num))
```

```
largest: 45
```

24. Find the smallest element in a list.

Code:

[4]:
```
num = [10, 25, 32, 45, 8]
print("smallest:", min(num))
```

```
smallest: 8
```

25. Reverse a list without using the reverse() function.

Code:

[6]:
```
num = [1, 2, 3, 4, 5]
print("reversed list:", num[::-1])
```

```
reversed list: [5, 4, 3, 2, 1]
```

6. Strings

26. Find the length of a string.

Code:

[7]:
```
text = input("Enter a string: ")
print("length:", len(text))
```

```
Enter a string:  fahim is a good boy
length: 19
```

27. Convert a string to uppercase.

Code:

[8]:
```
text = input("Enter a string: ")
print("uppercase:", text.upper())
```

```
Enter a string:  awamileague guni na
uppercase: AWAMILEAGUE GUNI NA
```

28. Convert a string to lowercase.

Code:

[9]:
```
text = input("Enter a string: ")
print("lowercase:", text.lower())
```

```
Enter a string:  BANGLADESH IS A SMALL COUNTRY
lowercase: bangladesh is a small country
```

29. Count occurrences of a character in a string.

    Code:

[10]:
```
text = input("Enter a string:")
char = input("Enter a character:")
print(f"occurrences of '{char}':", text.count(char))
```

```
Enter a string: fahim
Enter a character: i
```

30. Reverse a string without using the [::-1] method.

    Code:

[11]:
```
text = input("enter a string: ")
reversed_text = "".join(reversed(text))
print("reversed string:", reversed_text)
```

```
enter a string:  every action has an equal and opposit reaction
reversed string: noitcaer tisoppo dna lauqe na sah noitca yreve
```

7. Functions

31. Write a function to add two numbers.

    Code:

[12]:
```
def add(a, b):
    return a + b

print(add(5, 3))
```

```
8
```

32. Write a function to return the square of a number.

    Code:

[13]:
```
def square(n):
    return n ** 2

print(square(4))
```

```
16
```

33. Write a function to check if a number is even.

    Code:

[15]:
```
def is_even(n):
    return n % 2 == 0

print(is_even(8))
```

```
True
```

34. Write a function to return the factorial of a number.

    Code:

[16]:
```python
def factorial(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

print(factorial(5))
```

```
120
```

35. Write a function to find the largest of three numbers.

    Code:

[17]:
```python
def largest(a, b, c):
    return max(a, b, c)

print(largest(10, 25, 7))
```

```
25
```

8. Tuples

36. Create a tuple and print it.

    Code:

[18]:
```python
tuple = (1, 2, 3, 4, 5)
print(tuple)
```

```
(1, 2, 3, 4, 5)
```

37. Convert a tuple to a list.

    Code:

[19]:
```
t = (1, 2, 3, 4)
lst = list(t)
print(lst)
```

```
[1, 2, 3, 4]
```

38. Access the first and last elements of a tuple.

Code:

[20]:
```
t = (10, 20, 30, 40, 50)
print("First:", t[0], "Last:", t[-1])
```

```
First: 10 Last: 50
```

39. Find the length of a tuple.

Code:

[22]:
```
t = (10, 20, 30, 40)
print("Length:", len(t))
```

```
Length: 4
```

40. Check if an element exists in a tuple.

Code:

[24]:
```
t = (1, 2, 3, 4, 5)
num = int(input("Enter a number: "))
print("Exists" if num in t else "Does not exist")
```

```
Enter a number:  5
Exists
```

9. Dictionaries

41. Create a dictionary with 3 key-value pairs.

    Code:

```
[25]: student = {"name": "fahim", "age": 22, "grade": "D"}
      print(student)
```

```
{'name': 'fahim', 'age': 22, 'grade': 'D'}
```

42. Access a value from a dictionary using its key.

    Code:

```
[26]: print(student["name"])
```

```
fahim
```

43. Add a new key-value pair to a dictionary.

    Code:

```
[27]: student["city"] = "narayanganj"
      print(student)
```

```
{'name': 'fahim', 'age': 22, 'grade': 'D', 'city': 'narayanganj'}
```

44. Remove a key-value pair from a dictionary.

    Code:

```
[28]: del student["age"]
      print(student)
```

```
{'name': 'fahim', 'grade': 'D', 'city': 'narayanganj'}
```

45. Print all keys and values of a dictionary.

    Code:

```
[29]: for key, value in student.items():
          print(key, ":", value)
```

```
name : fahim
grade : D
city : narayanganj
```

10. Sets

46. Create a set and print it.

    Code:

```
[30]: s = {1, 2, 3, 4, 5}
      print(s)
```

```
{1, 2, 3, 4, 5}
```

47. Find the union of two sets.

    Code:

```
[31]: s1 = {1, 2, 3}
      s2 = {3, 4, 5}
      print("Union:", s1 | s2)

Union: {1, 2, 3, 4, 5}
```

48. Find the intersection of two sets.

    Code:

```
[32]: print("Intersection:", s1 & s2)

Intersection: {3}
```

49. Find the difference between two sets.

    Code:

```
[33]: print("Difference:", s1 - s2)

Difference: {1, 2}
```

50. Check if an element exists in a set.

    Code:

```
[34]: print(2 in s1)

True
```

Conclution: This set of 75 problems helps beginners strengthen their Python basics, improving problem-solving skills and logical reasoning.