Lab no.01

Name: Nehal Ali

Student's ID: BIT-24S-041

Introduction to python

Python is a versatile, high-level programming language known for its readability and ease of use, making it suitable for various applications, including web development, data analysis, and machine learning. Created by Guido van Rossum and first released in 1991, Python's syntax is designed to be simple and intuitive, resembling natural language.

It is used for:

- Web development (server-side)
- Software development
- Mathematics
- System scripting

Task no.1

Q) Make 2-2 programs of each datatype.

1) Integer datatype:

Program 1

```
# Program 1: Sum of two integers
a = 5
b = 10
sums = a + b
print("Sum is:", sums)
```

```
Sum is: 15
```

Program 2

```
# Program 2: Check if number is even or odd
num = 7
if num % 2 == 0:
    print("Even")
else:
    print("Odd")
```

```
Odd
```

2) Float datatype:

Program 1:

```
# Program 1: Area of a circle
radius = 4.5
area = 3.14 * radius * radius
print("Area of circle:", area)
```

```
Area of circle: 63.585
```

Program 2:

```
# Program 2: Celsius to Fahrenheit
celsius = 37.0
fahrenheit = (celsius * 9/5) + 32
print("Fahrenheit:", fahrenheit)
```

```
Fahrenheit: 98.6
```

3) String datatype:

Program 1:

```
# Program 1: Concatenate two strings
first_name = "Nehal"
last_name = "Ali"
full_name = first_name + " " + last_name
print("Full Name:", full_name)
```

```
Full Name: Nehal Ali
```

Program 2:

```
# Program 2: Count characters in a string
text = "Hello World"
print("Length:", len(text))
```

```
Length: 11
```

4) List datatype:

Program 1:

```
# Program 1: Sum of all elements in a list
numbers = [1, 2, 3, 4, 5]
if all(isinstance(x, (int, float)) for x in numbers):
    print("Sum:", sum(numbers))
else:
    print("Error: The list contains non-numeric values.")
```

```
Sum: 15
```

Program 2:

```
# Program 2: Append an item to list
fruits = ["apple", "banana"]
fruits.append("mango")
print(fruits)
```

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

5) Tuple datatype:

Program 1:

```
# Program 1: Accessing tuple elements
person = ("Nehal Ali", 25, "Karachi")
print(person[0])
```

```
Nehal Ali
```

Program 2:

```
# Program 2: Tuple unpacking
a, b = (10, 20)
print("a:", a, "b:", b)
```

```
a: 10 b: 20
```

6) Dictionary datatype:

Program 1:

```
# Program 1: Access value using key
student = {"name": "Nehal Ali", "age": 22}
print("Name:", student["name"])
```

```
Name: Nehal Ali
```

Program 2:

```
# Program 2: Add new key-value pair.
car = {"brand": "Toyota", "model": "Corolla"}
car["year"] = 2020
print(car)
```

```
{'brand': 'Toyota', 'model': 'Corolla', 'year': 2020}
```

7) Boolean datatype:

Program 1:

```
# Program 1: Check if value is greater
a = 5
b = 3
print(a > b)  # Output: True
```

```
True
```

Program 2:

```
# Program 2: Using boolean in condition
is_logged_in = True
if is_logged_in:
    print("Welcome!")
else:
    print("Please login.")
```

```
Welcome!
```

8) Complex datatype:

Program 1:

```
# Program 1: Add two complex numbers
a = 3 + 4j
b = 1 + 2j
print("Sum:", a + b)
```

```
Sum: (4+6j)
```

Program 2:

```
# Program 2: Multiply complex numbers
x = 2 + 3j
y = 1 - 1j
print("Product:", x * y)
```

```
Product: (5+1j)
```

9) Range datatype:

Program 1:

```
# Program 1: Print numbers from 1 to 5 using range
for i in range(1, 6):
    print(i)
```

```
1
2
3
4
5
```

Program 2:

```
# Program 2: Generate even numbers
even_numbers = list(range(2, 11, 2))
print(even_numbers)
```

```
[2, 4, 6, 8, 10]
```

10) Set datatype:

Program 1:

```
# Program 1: Create a set and add elements
my_set = {1, 2, 3}
my_set.add(4)
print(my_set)
```

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

Program 2:

```
# Program 2: Set operations (union)
a = {1, 2, 3}
b = {3, 4, 5}
print("Union:", a | b)
```

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

11) Frozenset datatype:

Program 1:

```
# Program 1: Create a frozenset
f_set = frozenset([1, 2, 3, 4])
print(f_set)
```

```
frozenset({1, 2, 3, 4})
```

Program 2:

```
# Program 2: Froenset intersection
a = frozenset([1, 2, 3])
b = frozenset([2, 3, 4])
print("Intersection:", a & b)
```

```
Intersection: frozenset({2, 3})
```

Task no.2

Q) Make up to 5 Shape programs using '*'.

Shape 01:

```
# 01 Make a diamond shape using ' * ' by logic method
rows = 5
for i in range(1, rows + 1):
    for j in range(i):
        print('*', end='')
    print()
```

```
*
**
***
****
*****
```

Shape 02:

```
# 02 Make a square shape using ' * ' by logic method
rows = 5
for i in range(rows):
    print('*' * rows)
```

```
*****
*****
*****
*****
*****
```

Shape 03:

```
# 03 Make a rectangle shape using ' * ' by logic method
rows = 5
columns = 10
for i in range(rows):
    print('*' * columns)
```

```
**********
**********
**********
**********
**********
```

Shape 04:

```
# 04 Make a Triangle shape using ' * ' by logic method
rows = 5
for i in range(1, rows + 1):
    print(' ' * (rows - i), end='')
    print('* ' * i)
```

```
    *
   * *
  * * *
 * * * *
* * * * *
```

Shape 05:

```
# 05 Make a diamond shape using ' * ' by logic method
rows = 5
for i in range(1, rows + 1, 2):
    spaces = (rows - i) // 2
    print(' ' * spaces + '*' * i)
for i in range(rows - 2, 0, -2):
    spaces = (rows - i) // 2
    print(' ' * spaces + '*'*i)
```

```
  *
 ***
*****
 ***
  *
```

Task 03:

Q) Make same shapes you have made in task 2, using * multiply by number.

Shape 01:

```
# 01 Make a diamond shape using 'numbers' by logic method
rows = 5
for i in range(1, rows + 1):
    for j in range(1, i + 1):
        print(j, end=' ')
    print()
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Shape 02:

```
# 02 Make a square shape using 'numbers' by logic method
rows = 5
for i in range(rows):
    for j in range(1, rows + 1):
        print(j, end='')
    print()
```

```
12345
12345
12345
12345
12345
```

Shape 03:

```
# 03 Make a rectangle shape using ' * ' by logic method
rows = 5
columns = 10
for i in range(rows):
    for j in range(1, columns + 1):
        print(j, end='')
    print()
```

```
12345678910
12345678910
12345678910
12345678910
12345678910
```

Shape 04:

```
# 04 Make a Triangle shape using ' * ' by logic method
rows = 5
for i in range(1, rows + 1):
    print(' ' * (rows - i), end='')
    print((str(i) + ' ') * i)
```

```
    1
   2 2
  3 3 3
 4 4 4 4
5 5 5 5 5
```

Shape 05:

```
# 05 Make a diamond shape using ' * ' by logic method
rows = 5
rows = 5
for i in range(1, rows + 1, 2):
    spaces = (rows - i) // 2
    print(' ' * spaces + (str(i) * i))
for i in range(rows - 2, 0, -2):
    spaces = (rows - i) // 2
    print(' ' * spaces + (str(i) * i))
```

```
  1
 333
55555
 333
  1
```