Day 4 (28/8/25): Floating Point Types and Strings in Python

Part 1: Floating Point Types

1. What are Floating Point Types?

Floating point types are used to represent **numbers with a decimal point**.
In Python, this is handled by the built-in **float data type**.

Example:

```
pi = 3.14159
temperature = -7.5
```

2. Characteristics of Floats

1. **Decimal Numbers** – Can store numbers like 0.5, -2.75, 100.0.
2. **Scientific Notation** – Python supports e notation for very large or small numbers.
   - Example: 1.2e3 → 1200.0, 3.5e-2 → 0.035
3. **Immutable** – Float values cannot be changed; operations return new float values.
4. **Precision** – Python floats are implemented using **64-bit double precision**.

3. Operations on Floats

- Arithmetic: +, -, *, /, // (floor division), %, **
- Comparison: <, >, ==, !=

4. Example Program – Floating Point Numbers

```
# Floating Point Example
a = 5.5
b = 2.0

sum_val = a + b
difference = a - b
product = a * b
division = a / b
power = a ** b

print("a + b =", sum_val)
print("a - b =", difference)
print("a * b =", product)
print("a / b =", division)
print("a ** b =", power)
```

Output:

```
a + b = 7.5
a - b = 3.5
a * b = 11.0
a / b = 2.75
a ** b = 30.25
```

Part 2: Strings in Python

1. What is a String?

A string is a **sequence of characters** enclosed in **single (' ')** or **double (" ")** quotes.
Strings can contain letters, numbers, symbols, or spaces.

Examples:

```
name = "Alice"
greeting = 'Hello, World!'
```

2. Characteristics of Strings

1. **Immutable** – Once created, the string cannot be changed.
2. **Indexing** – Characters in a string can be accessed using **indices** (0-based).
    - Example: name[0] → 'A'
3. **Slicing** – Substrings can be extracted.

   - Example: name[1:4] → 'lic'

4. **Concatenation** – Strings can be joined using +.
   - Example: "Hello " + "Alice" → "Hello Alice"
5. **Repetition** – Strings can be repeated using *.
   - Example: "Hi! " * 3 → "Hi! Hi! Hi! "

3. Example Program – Strings

```
# String Example
greeting = "Hello"
name = "Alice"

# Concatenation
message = greeting + ", " + name + "!"
print(message)

# Indexing
print("First character:", name[0])

# Slicing
print("Substring:", name[1:4])

# Repetition
print("Repeat greeting:", greeting * 3)
```

Output:

```
Hello, Alice!
First character: A
Substring: lic
Repeat greeting: HelloHelloHello
```