TechSaksham Training - Day 3 Notes

🔹 Introduction to Python

- **Founder:** Guido Van Rossum (1991)
- **Key Features:**
  - Open Source
  - Platform Independent
  - Object-Oriented Programming (OOPs)
  - Interpreted Language
  - Dynamically Typed
  - General Purpose

🔹 Python OOPs Concepts

- **Class & Object**: Python follows an object-oriented approach where everything is an object.
- **Encapsulation**: Wrapping data and methods into a single unit (class) to restrict direct access.
- **Inheritance**: One class can inherit attributes and methods from another class to promote code reusability.
- **Polymorphism**: Methods can have the same name but behave differently depending on the object.
- **Abstraction**: Hiding implementation details and showing only the necessary functionalities.

Example:

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def display(self):
        print(f"Name: {self.name}, Age: {self.age}")

p1 = Person("Tanisha", 20)
p1.display()
```

🔹 OOPs Concepts in Detail

- Discussed the differences between **Compiler vs. Interpreter**
- **Python Memory Management:**
    - **Memory Storage:**
        - RAM → Python Memory → Two parts: **Stack & Heap**
    - **Variable Storage:**
        - Variable stored in Stack, Value stored in Heap
        - Accessed using `id()`
    - **Memory Optimization:**
        - If variables have the same value, only one value is stored in Heap

Example:

```
 a = 1
b = 1
print(id(a))
print(id(b))
```

Output:

```
 140736423868856
140736423868856
```

-

🔹 Print Methods in Python

Printing in one line:

```
 print("*", end="")
print("*", end="")
print("*", end="")
```

Output:

```
 ***
```

-

Printing multiple characters:

```
 print(100 * "*")
```

-

Printing multiple variables:

```
 name = "Tanisha"
```

```
age = 20
print("My name is", name, "and my age is", age)
```

Output:

My name is Tanisha and my age is 20

-

🔹 Data Types in Python

List

- Stores multiple values in a single variable
- Allows duplicate values
- Supports positive & negative indexing
- Mutable (can be modified)
- Can be **multi-dimensional**

Ways to define a list:

1. **Using built-in methods**
2. **Using constructors**

Tuple

- Stores multiple values
- Allows duplicate values
- **Immutable** (cannot be changed)
- Supports indexing
- Can add two tuples together

Dictionary

- Stores values in **key-value pairs**
- Keys must be unique, values can be duplicated
- Ordered data type
- Mutable (values can be changed)

Set

- **Unordered** multi-collection data type
- **No duplicate values allowed**
- Defined using `{}`
- Elements get shuffled in output

🔹 Conditional Statements in Python

If-Else Statement

```
if condition:
    expression
else:
    expression
```

Elif Statement

```
if condition:
    expression
elif condition:
    expression
else:
    expression
```

🔹 Loops in Python

While Loop

```
initialization
while condition:
    expression
    increment/decrement
```

For Loop

```
for variable in range(start, end, step):
    expression
```

🔹 Hands-on Problems Solved

✅ Print even numbers from a list
✅ Sort a given list **without using built-in functions**
✅ Find the **second lowest number** from a list with duplicates
✅ Pattern Printing Example:

```
a = ["A", "B", "C", "D"]
for i in range(len(a)):
```

```
    print((a[i] + " ") * (i + 1))
```

Output:

A
B B
C C C
D D D D

✅ Display student results using **roll number & name**
✅ Code for **ATM Transaction System**

💡 Summary:

Day 3 focused on **Python basics, OOPs, memory management, data types, conditional statements, and loops**. Looking forward to applying Python in **databases and analytics** in the coming sessions! 🚀