Assignment-1

Python Fundamentals

1. Introduction to Python and its Features (simple, high-level, interpreted language).

Python is a high level, interpreted programming language.
It was created by Guido van Rossum and first released in 1991.

Features of Python

1. Simple and Easy to Learn

- Python has a clean and straightforward syntax.
- Writing Python code is often similar to writing english

2. High-Level Language

- Python handles complex details like memory management automatically.
- You can focus on problem-solving instead of worrying about system-level programming.

3. Interpreted Language

- Python code is executed line-by-line by the interpreter.
- This makes it easy to test and debug your programs.

4.object -oriented

- Python is object-oriented but supports both functional and object-oriented programming.

5. Platform-independent

- Python is platform-independent.
- You don't need to write a them separately for each platform.

6. Open-Source

- It is free and its source code is freely available to the public.

- You can download python from the official python website.

7. Large Standard library

- The standard library is large and has many packages and modules with common and important functionality.

8. Dynamically-Typed language

- You don't need to declare data type while defining a variable
- This is easy for programmers but can everything in python is an object.

9. GUI support

- You can use python to create GUI (Graphical user interfaces)

2.History of Python.

⭕Python is a high-level programming language developed by Guido van Rossum in the late 1980s.

⭕He started working on it in 1989 during his time at the Centrum
Wiskunde & Informatica (CWI) in the Netherlands. The first official version, Python 0.9.0, was released in 1991.

⭕Guido Van Rossum created Python as a successor to the ABC programming language, aiming to fix some of its issues while adding new features like exception handling and extensibility.

⭕The language was designed to be easy to read, simple to write,
and powerful for both beginners and experienced developers.

3. Advantages of using Python over other programming languages.

1. Easy to Learn and Use

- Python's syntax is simple and readable, making it an excellent choice for beginners
- Its structure resembles plain English, reducing complexity compared to languages like C++ or Java.

2. Cross-Platform Compatibility

- Python runs seamlessly on Windows, Mac, and Linux without modification.

3. Extensive Libraries and Frameworks

- Python has powerful libraries like NumPy, Pandas, TensorFlow, and Django, speeding up development.
- Whether it's AI, machine learning, web development, or automation, Python has a solution.

4. Strong Community Support

- Python boasts a large, active developer community for support and troubleshooting.
- Continuous updates ensure Python stays relevant and secure for modern applications

.

5. High Demand and career Growth

- Python is widely used in industries like finance, healthcare, education, and cybersecurity.
- Many companies, including Google, Facebook, and Netflix, rely on Python for various tasks.

4. Installing Python and setting up the development environment (Anaconda, PyCharm, or VS Code).

Installing Python

- Visit the official Python website and download the latest version. Run the installer and check the box for "Add Python to PATH" before clicking Install Now. Once installed, verify by running python --version in the Command Prompt (Windows) or Terminal (Mac/Linux).

Setting Up Development Environments

VS Code (Lightweight & Versatile)

- Get VS Code . Install the Python extension from the Marketplace. Set up the Python interpreter under Settings >

Python: Interpreter. Open a new .py file and start coding!

5.Writing and executing your first Python program.

Step 1: Write a Simple Python Program

Let's start with a classic **"Hello, World!"** program:
Code:
Print ("Hello, World!")
This program simply prints "Hello, World!" to the screen.

Step 2: Save the File

- Open any text editor (like Notepad, VS Code, or PyCharm). Type

the code above. Save the file with a .py extension, like hello.py.

Step 3: Running in an IDE (PyCharm or VS Code)

- Open PyCharm or VS Code and create a new Python file. Write the code inside the editor. Click Run or press Ctrl + Shift + F10 (PyCharm) / F5 (VS Code).

Programming Style

1.Understanding Python's PEP 8 Guidelines

- PEP 8 (Python Enhancement Proposal 8) is the official style guide for writing Python code. It helps ensure that Python code is readable, consistent, and maintainable, especially when working in teams or on large projects.

2. Indentation, comments, and naming conventions in Python.

Indentation in Python:

Indentation defines code blocks (like in loops, functions, classes). Python does not use **{}** brackets like other languages—indentation is mandatory.

- Use 4 spaces per indentation level .

⭕Comments in Python

- Comments explain why something is done, not what—that should be clear from the code.

⭕Naming Conventions in Python

Type
Convention
Example
Variable lower_case_with_underscores user_name
Function lower_case_with_underscores
def_total()
Class
CapWords (PascalCase)
StudentDetails
Constant
ALL_CAPS
MAX_SIZE = 100

3. Writing readable and maintainable code.

- Readable Code: Code that is easy to read and understand.
- Maintainable Code: Code that is easy to fix, improve, or update.

⭕ Simple Rules to Write Good Code

Use good names Easy to understand
Use 4 spaces for indentation Keeps code clean and structured
Write comments Explains your thinking
Use functions Reuse code, avoid repeating
Don’t hardcode numbers Easy to update
Rule Why?

Conditional Statements

1.Introduction to conditional statements: if, else, elif.

- Conditional statements allows your Python program to make decisions based on certain conditions.

1) If statement:

The if statement checks a condition. If it is True, the block of code under it runs.

2) Else statement:

The else statement runs when the if condition is False.

3) Elif statement:

The elif allows you to check multiple conditions.

- Basic rules:

- Use colon: after if , elif and else.
- You can have multiple elif, but only one else.

2. Nested if-else conditions.

- Nested if…else condition means using one if…else inside another if…else. This is useful when decisions depend on multiple levels of conditions.
- You can have **if** inside **if** or **if** inside **else**.

Looping (For, While)

1.Introduction to for and while loops

- **For Loop**

A **for** loop is used to iterate over a sequence (such as a list, tuple,
dictionary, string, or range). It executes a block of code for each
item in the sequence.

Example : Using range()

```
for i in range (5):
print(i)
```

This prints numbers from 0 to 4.

- **While Loop**

A **while** loop continues to execute a block of code **as long as** a specified condition is **true**.

Example :

i = 0
while count < 5:
print(i)
i += 1
This prints numbers from 0 to 4.

2. How loops work in Python.

- Loops in Python allow you to execute a block of code multiple times without writing it repeatedly. They help automate repetitive tasks, making programs more efficient.

How Loops Work

Python has two main types of loops: **for loops** and **while loops**.

For Loop

A **for loop** iterates over a sequence (like a list, tuple, string, or range) and executes the loop body for each element.

# Example of a for loop:
for i in range (5):
print(i)

- For each number i in the sequence 0,1,2,3,4 Do the following block of code.
- The print(i) line runs once per loop cycle, printing the current
value of i .

Output:
0
1
2
3
4
5

While Loop

A while loop executes as long as a condition remains True.

Example:

```
count = 0
while count < 3:
print(count)
count += 1
```

- Check if count < 3 → yes, so print 0.
- Add 1 to count → now it's 1.
- Check again → still less than 3, so print 1.
- Repeat until count is no longer less than 3.

Output:

```
0
1
2
```