***PYTHON QUESTIONS***

1.What is python?

Ans: **Python** is a high-level, interpreted programming language known for its **simplicity**, **readability**, and **versatility**.

2.Tell me the areas where python is being used?

Ans: Python is used in a **wide variety of fields** due to its flexibility, ease of use, and powerful libraries. The **main areas where Python is commonly used are**:

* **Data Science & Analytics**
* **Machine Learning & Artificial Intelligence**
* **Web Development**

3. What is high level and low-level language?

Ans: Programming languages are classified based on how **close they are to human language or machine code**.

**High-Level Language**

A **high-level language** is a programming language that is **closer to human language** and abstracts away most hardware details.

**Characteristics:**

* Easy to read, write, and understand
* Portable (can run on different machines)

**Examples:**

* Python
* Java

**Low-Level Language**

A **low-level language** is closer to **machine code (binary)** and gives direct control over hardware.

**Characteristics:**

* Harder to read and write
* Fast and efficient

**EXAMPLES:**

* **Machine Language** (binary code)
* **Assembly Language** (uses mnemonics like MOV, ADD, etc.)

4. What is interpreted language?

Ans: An **interpreted language** is a programming language that is **executed line-by-line** by an **interpreter**, rather than being fully compiled into machine code before running.

5. What is compiled language?

Ans: A **compiled language** is one where the source code is **translated into machine code** (binary) **before execution** using a **compiler**. The resulting file (called an executable) can be run directly by the computer.

6. What is statically typed language?

Ans: A **statically typed language** is one in which **variable types are known at compile-time**. This means you must **declare the type of a variable before using it**, and the compiler **checks types for correctness** before the program runs.

7. What is dynamically typed language?

Ans: A **dynamically typed language** is one where **variable types are determined at runtime**, not in advance. You **don’t need to declare the type of a variable** before using it — the language figures it out while the program is running.

8. What is weekly typed language?

Ans: A **weakly typed language** is a programming language that **allows implicit type conversions (type coercion)** between different data types, sometimes leading to unexpected results.

9. What is strongly typed language?

Ans: A **strongly typed language** is a programming language that **enforces strict rules on how types can be used and combined**. It **does not allow implicit or automatic type conversions** (or very limited), which helps prevent unexpected behaviour and bugs.

10. What is .PYC file (Byte code)?

Ans: A **.pyc file** is a **compiled Python bytecode file**. When you run a Python program, the interpreter **compiles your .py source code into bytecode**, which is a low-level, platform-independent representation of your code.

11. What is PVM?

Ans: The **Python Virtual Machine (PVM)** is the part of the Python runtime environment that **executes Python bytecode**.

12. How Python internally works?

1. Ans: **Source Code (.py file)**
   * You write Python code in a human-readable form (e.g., print("Hello")).
2. **Lexical Analysis (Tokenizer)**
   * Python breaks the source code into tokens — the smallest meaningful units like keywords, identifiers, operators, literals.
3. **Parsing**
   * The tokens are parsed to build a **syntax tree** (also called Abstract Syntax Tree or AST) representing the program’s structure.
4. **Compilation to Bytecode**
   * The AST is compiled into **bytecode**, a low-level, platform-independent representation.
   * This bytecode is stored in .pyc files (inside \_\_pycache\_\_) for faster loading next time.
5. **Execution by Python Virtual Machine (PVM)**
   * The **PVM** reads and executes the bytecode instructions.
   * PVM is a stack-based virtual machine that performs operations like arithmetic, function calls, variable handling, etc.
6. **Memory Management**
   * Python handles memory allocation and garbage collection automatically.
   * It keeps track of objects, references, and frees unused memory.

13. What is PEP 8?

Ans: **PEP 8** is the **official style guide for writing Python code**. It stands for:

**P**ython **E**nhancement **P**roposal **8**

**Purpose of PEP 8:**

* To provide **coding conventions** for Python code.
* Makes code **more readable**, **consistent**, and **maintainable**—especially in large teams or open-source projects.

14. What is PIP? What is the use of PIP?

Ans: **PIP** stands for **“Pip Installs Packages”** or **“Preferred Installer Program.”**  
It is the **official package manager for Python**.

**What is the use of PIP?**

PIP is used to:

* **Install** Python libraries and frameworks.
* **Uninstall** packages you no longer need.
* **Upgrade** existing packages.
* **List** all installed packages.
* **Download** packages from the **Python Package Index (PyPI)**.

15. What is Byte code and when is it created?

Ans: **Bytecode** is a **low-level, platform-independent representation** of your Python code. It’s not human-readable and is meant to be executed by the **Python Virtual Machine (PVM)**.

Bytecode is created **automatically when you run a Python program**. Here's the full process:

1. You write Python source code (.py file).
2. Python **compiles** the source code into **bytecode**.
3. This bytecode is stored in a .pyc file (in a \_\_pycache\_\_ directory).
4. The **PVM executes** the bytecode.

16. What is indentation in python? Does python relay on indentation?

Ans: **Indentation** in Python refers to the **spaces or tabs** used at the beginning of a line of code to **define blocks** (such as inside if, for, while, functions, etc.).

Unlike many other languages that use **brackets {}** to mark code blocks, **Python uses indentation**. This makes indentation **not optional** — it's part of the **syntax**.

17. What is variable?

Ans: A **variable** is a **name** that refers to a value stored in memory.  
It acts like a **container** or **label** that holds data you can use and change later in your program.

18. What are variable naming rules?

Ans: When naming variables in Python, there are a few important rules and best practices to follow to ensure your code works properly and remains readable.

**✅ Official Rules (Must Follow)**

1. **Start with a letter or underscore**
   * A variable **must begin** with a **letter (A–Z or a–z)** or an **underscore (\_).**
2. **Can contain letters, digits, and underscores**
3. **Case-sensitive**
   * Python treats uppercase and lowercase names as**different**.

4.**Cannot be a reserved keyword**

* + You **cannot use Python keywords** as variable names.

19. What is datatype and name of datatypes.

Ans: A **data type** defines the **type of value** a variable can hold and what **operations** can be performed on that value.

| **Category** | **Data Type** | **Example** |
| --- | --- | --- |
| **Numeric** | int | 10, -3, 1000 |
|  | float | 3.14, -0.5, 2.0 |
|  |
| **Text** | str | "Hello", 'Python' |
| **Boolean** | bool | True, False |
| **Sequence** | list | [1, 2, 3], ["a", "b"] |
|  | tuple | (1, 2), ("x", "y") |
|  | range | range(5) |
| **Set Types** | set | {1, 2, 3} |
|  |
| **Mapping** | Dict | {"name": "Alice", "age": 25} |
| **None Type** | NoneType | None |

20. What is keyword in Python?

Ans: A **keyword** in Python is a **reserved word** that has a **special meaning** in the language.  
You **cannot** use keywords as **variable names**, **function names**, or **identifiers**.