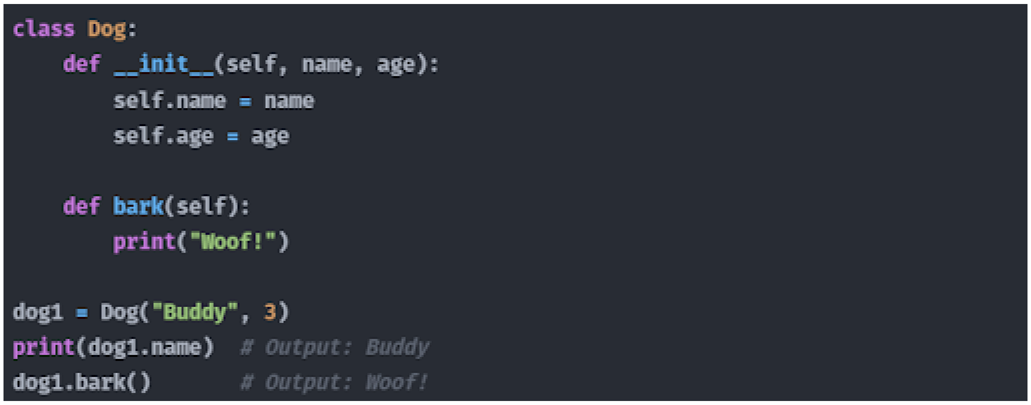
What is Python

Python is a high-level, interpreted, dynamically-typed, object-oriented Computer language.



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

    def bark(self):
        print("Woof!")

dog1 = Dog("Buddy", 3)
print(dog1.name)  # Output: Buddy
dog1.bark()       # Output: Woof!
```

What Is Language

Language is a way to share or express messages.

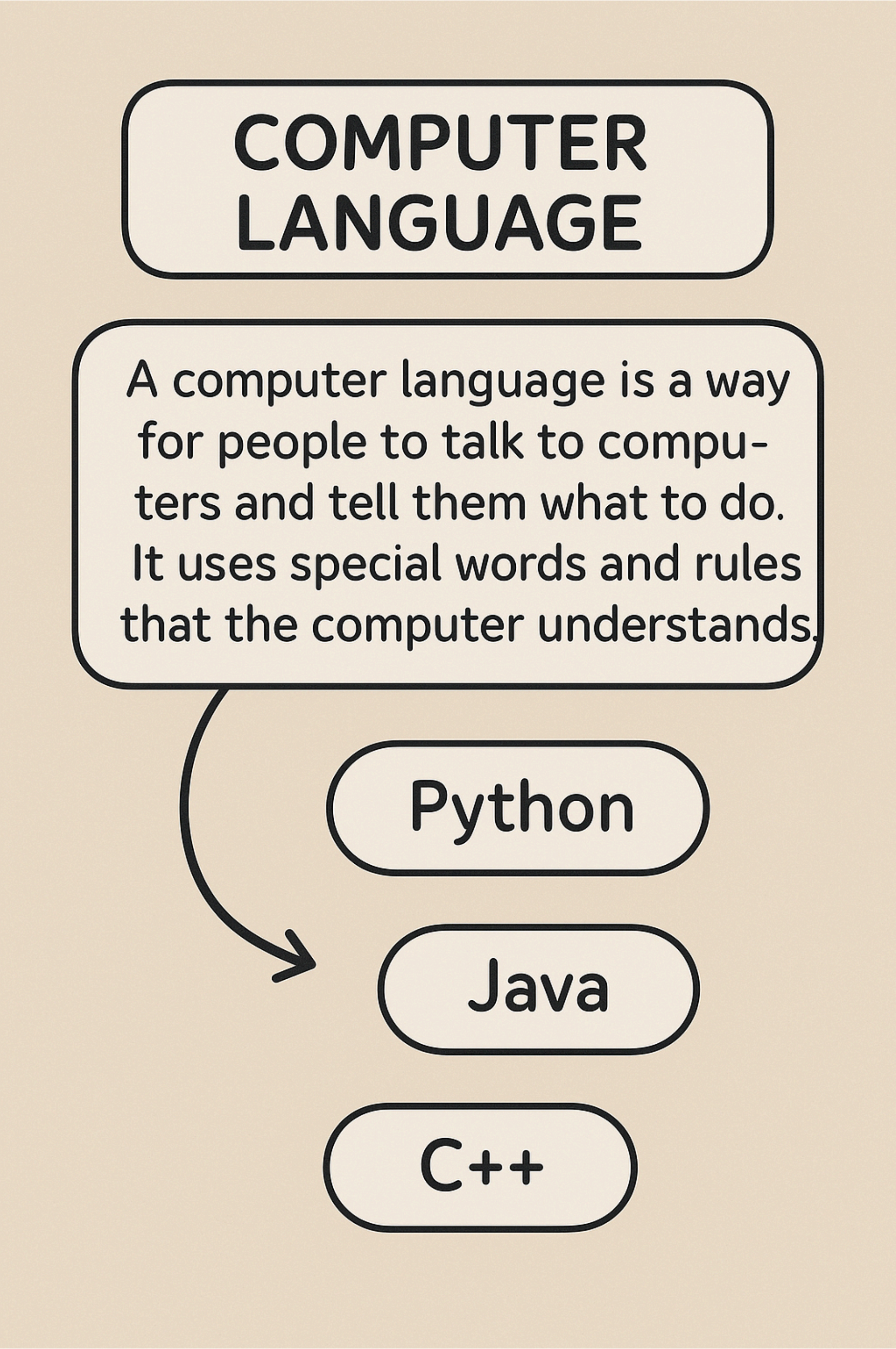
Language is how we share messages with others.

💻 **What is a Computer Language?**

A computer language is a way to talk to computers. We use it to give instructions, so the computer can do things like:

🧮 Doing calculations
💬 Showing messages
🕹️ Running apps or games

Just like we use Urdu or English to talk to people, we use Python, C++, or Java to talk to computers.



COMPUTER LANGUAGE

A computer language is a way for people to talk to computers and tell them what to do. It uses special words and rules that the computer understands.

Python

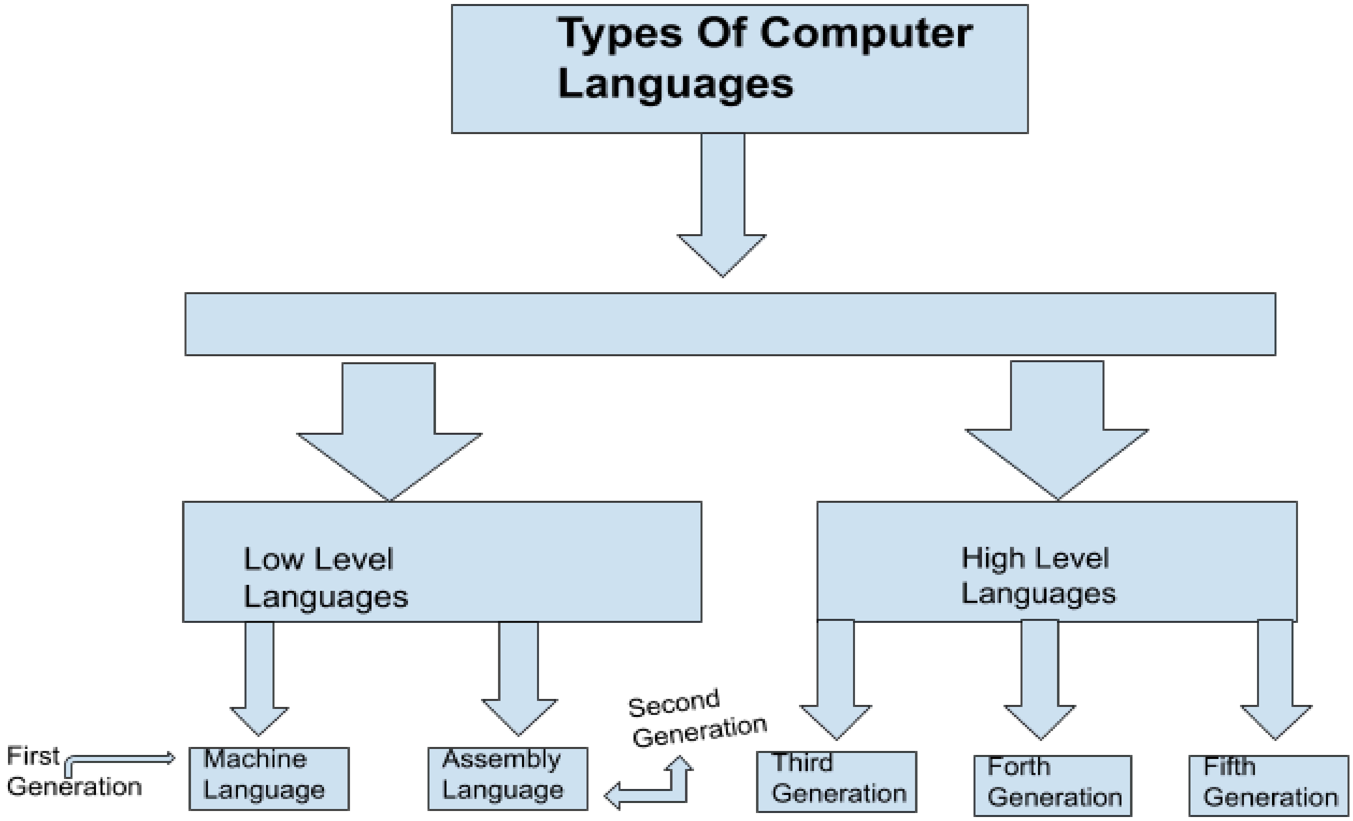
Java

C++

Types of Computer Languages

Whether you're a beginner or have some experience, learning about these language types helps you understand how computers work and how we use code to communicate with them.

🧩 Types of Computer Languages



🟢 1. Machine Language (Low-Level Language)

🧬 1st Generation Language

- The only language a computer truly understands
- Written in **binary code — 0s and 1s**

- Very difficult for humans to read or write

Example:

```
10110000 01100001
```

✅ **Pros:**

- Very fast
- Directly executed by the computer

❌ **Cons:**

- Not human-friendly
- Hardware-specific

🟡 2. Assembly Language

🧬 2nd Generation Language

- A step above machine language
- Uses **short mnemonics** like `MOV`, `ADD`, `SUB`
- Needs an **assembler** to convert into machine language

Example:

```
MOV A, B
```

✅ **Pros:**

- Easier to write than machine code
- Gives more control over hardware

❌ **Cons:**

- Still complex
- Not portable across different systems

🔵 3. High-Level Language

🧬 3rd Generation Language

- Easy to read, write, and understand
- Uses English-like syntax
- Requires a **compiler or interpreter**

Examples:
`Python, Java, C++, JavaScript`

✅ **Pros:**

- Great for beginners
- Widely used in software development
- Portable across platforms

❌ **Cons:**

- Not as fast as low-level languages
- Less control over hardware

🟣 4. Very High-Level Language / 4GL

🧬 4th Generation Language

- Even more abstract than high-level languages
- Often used for **databases**, **reporting**, **analytics**

Examples:
`SQL, MATLAB, Oracle Reports`

✅ Pros:

- Fast development
- Minimal code needed

❌ Cons:

- Limited flexibility
- Mostly domain-specific

🔴 5. Scripting Languages

- Used for **automating tasks**, **web scripting**, and **small programs**
- Usually interpreted

Examples:
`Python, JavaScript, PHP, Bash`

✅ **Pros:**

- Quick to write
- Great for web and automation tasks

❌ **Cons:**

- Slower execution
- Sometimes less structured

🟠 Bonus: Domain-Specific Languages (DSLs)

These languages are **tailored for specific domains** — not for general-purpose programming.

Examples:

- `HTML` – For webpage structure
- `CSS` – For styling
- `SQL` – For databases

📊 Summary Table

Type	Easy to Use	Speed	Example
Machine Language	❌ No	✅ Very Fast	`0s and 1s`
Assembly Language	⚠️ Medium	✅ Fast	`MOV A, B`
High-Level Language	✅ Yes	⚠️ Slower	`Python, Java, C++`
Very High-Level Language (4GL)	✅ Yes	⚠️ Medium	`SQL, MATLAB`
Scripting Language	✅ Yes	⚠️ Medium	`Python, JavaScript, PHP`
Domain-Specific Language (DSL)	✅ Yes (Specific)	⚠️ Depends	`HTML, CSS, SQL`

🎓 Conclusion

Understanding the types of computer languages is the **first step** toward becoming a great programmer.

- Most modern software is written in **high-level or scripting languages**,

- But having knowledge of **low-level** ones like **machine** and **assembly** gives you a deeper understanding of how computers really work.

This foundation helps you make better decisions as a **developer**, **engineer**, or even a **tech leader**.