# **🌀 Python Polymorphism - Complete Guide**

## **✅ What is Polymorphism?**

**Polymorphism** means **"many forms"**. It allows the **same method name** to perform **different behaviors** depending on the object that is calling it.

## **🔑 Real-Life Analogy**

Imagine both your **father** and **mother** say "Good morning."

* Father says it formally (strict tone)
* Mother says it softly (warm tone)

Even though the **action is same**, the **behavior is different**.  
 That is exactly what **polymorphism** is in programming.

## **💪 Types of Polymorphism in Python**

| **Type** | **Description** | **Python Support?** |
| --- | --- | --- |
| Method Overriding | Child class redefines parent's method | ✅ Yes |
| Method Overloading | Same method name with different parameters | ❌ Not natively |
| Duck Typing | Type doesn't matter, behavior does | ✅ Yes |

## **📆 Example 1: Method Overriding (Runtime Polymorphism)**

class Animal:

def sound(self):

print("Some generic sound")

class Dog(Animal):

def sound(self):

print("Bark")

class Cat(Animal):

def sound(self):

print("Meow")

# Polymorphic behavior

for animal in [Dog(), Cat()]:

animal.sound()

### **🖊️ Output:**

Bark

Meow

✅ Same method name, different behavior based on the object.

## **🦆 Example 2: Duck Typing (Dynamic Polymorphism)**

class Pycharm:

def execute(self):

print("Compiling + Running")

class VSCode:

def execute(self):

print("Running + Linting")

def code(editor):

editor.execute()

code(Pycharm())

code(VSCode())

Python doesn't care about the **type** of editor. As long as it has an execute() method, it works!

✅ This is called **Duck Typing**:

"If it walks like a duck and quacks like a duck, treat it like a duck."

## **❌ What About Method Overloading?**

Python does **not support method overloading directly**:

class Demo:

def show(self, a):

print("One argument")

def show(self, a, b): # This will override the previous method

print("Two arguments")

obj = Demo()

obj.show(1) # Error: missing 1 required positional argument

### **✅ Workaround: Using \*args**

class Demo:

def show(self, \*args):

if len(args) == 1:

print("One argument")

elif len(args) == 2:

print("Two arguments")

obj = Demo()

obj.show(1)

obj.show(1, 2)

## **🚀 Benefits of Polymorphism**

1. **Flexible Code** – Functions work with any object with correct method
2. **Reusable** – Same method name reused across classes
3. **Cleaner Design** – Encourages abstraction and loose coupling
4. **Supports OOP** – Fundamental to dynamic typing and late binding

## **❓ Common Interview Questions**

1. What is polymorphism in Python?
2. What is the difference between overloading and overriding?
3. How does Python handle method overloading?
4. What is duck typing?
5. Can Python achieve compile-time polymorphism?
6. How is polymorphism different from inheritance?

## **💼 Resume Tip**

**Project Line Example:**

"Designed a plugin-based editor using polymorphism by defining a base Plugin interface and dynamically loading SyntaxHighlighter, AutoFormatter, and Linter classes implementing run() differently."

✅ Highlights code flexibility and extensibility.

### **About the Author**

**Gowtham SB** is a **Data Engineering expert, educator,** **and content creator** with a passion for **big data technologies, as well as cloud and Gen AI** . With years of experience in the field, he has worked extensively with **cloud platforms, distributed systems, and data pipelines**, helping professionals and aspiring engineers master the art of data engineering.

Beyond his technical expertise, Gowtham is a **renowned mentor and speaker**, sharing his insights through engaging content on **YouTube and LinkedIn**. He has built one of the **largest Tamil Data Engineering communities**, guiding thousands of learners to excel in their careers.

Through his deep industry knowledge and hands-on approach, Gowtham continues to **bridge the gap between learning and real-world implementation**, empowering individuals to build **scalable, high-performance data solutions**.

𝐒𝐨𝐜𝐢𝐚𝐥𝐬

🎥𝐘𝐨𝐮𝐓𝐮𝐛𝐞 - https://www.youtube.com/@dataengineeringvideos

📸𝐈𝐧𝐬𝐭𝐚𝐠𝐫𝐚𝐦 - <https://instagram.com/dataengineeringtamil>

📸𝐈𝐧𝐬𝐭𝐚𝐠𝐫𝐚𝐦 - [https://instagram.com/](https://instagram.com/dataengineeringtamil)thedatatech.in

🤝𝐂𝐨𝐧𝐧𝐞𝐜𝐭 𝐟𝐨𝐫 𝟏:𝟏 - https://topmate.io/dataengineering/

💼𝐋𝐢𝐧𝐤𝐞𝐝𝐈𝐧 - https://www.linkedin.com/in/sbgowtham/

🌐𝐖𝐞𝐛𝐬𝐢𝐭𝐞 - https://codewithgowtham.blogspot.com

💻𝐆𝐢𝐭𝐇𝐮𝐛 - http://github.com/Gowthamdataengineer

💬𝐖𝐡𝐚𝐭𝐬 𝐀𝐩𝐩 - https://lnkd.in/g5JrHw8q

📧𝐄𝐦𝐚𝐢𝐥 - atozknowledge.com@gmail.com

📱𝐀𝐥𝐥 𝐌𝐲 𝐒𝐨𝐜𝐢𝐚𝐥𝐬 - <https://lnkd.in/gf8k3aCH>