# **🧬 Python Inheritance - Complete Guide**

## **✅ What is Inheritance in Python?**

Inheritance is an OOP concept where **one class (child)** gets access to the **properties and methods of another class (parent)**. It helps with **code reusability**, **extensibility**, and **modularity**.

### **🔑 Real-Life Analogy:**

* **Father → Son**: Son gets some qualities and behavior from the father.
* **Mother → Daughter**: Daughter may inherit traits from her mother.
* Each child may also have their **own unique abilities**.

## **🧱 Syntax of Inheritance**

class Parent:

def greet(self):

print("Hello from Parent")

class Child(Parent): # Inherits from Parent

def play(self):

print("Child is playing")

c = Child()

c.greet() # Inherited

c.play() # Child's own method

## **🔀 Types of Inheritance in Python**

### **1. Single Inheritance**

class Father:

def drive(self):

print("Father can drive")

class Son(Father):

def play(self):

print("Son can play")

s = Son()

s.drive()

s.play()

### **2. Multi-Level Inheritance**

class Grandfather:

def wisdom(self):

print("Grandfather shares wisdom")

class Father(Grandfather):

def drive(self):

print("Father can drive")

class Son(Father):

def play(self):

print("Son can play")

s = Son()

s.wisdom()

s.drive()

s.play()

### **3. Hierarchical Inheritance**

class Mother:

def cook(self):

print("Mother cooks well")

class Daughter(Mother):

def dance(self):

print("Daughter can dance")

class Son(Mother):

def play(self):

print("Son can play")

d = Daughter()

d.cook()

d.dance()

s = Son()

s.cook()

s.play()

### **4. Multiple Inheritance**

class Father:

def drive(self):

print("Father drives")

class Mother:

def cook(self):

print("Mother cooks")

class Child(Father, Mother):

def play(self):

print("Child plays")

c = Child()

c.drive()

c.cook()

c.play()

### **5. Hybrid Inheritance**

class A:

def method\_a(self):

print("A")

class B(A):

def method\_b(self):

print("B")

class C(A):

def method\_c(self):

print("C")

class D(B, C):

def method\_d(self):

print("D")

obj = D()

obj.method\_a() # Comes from MRO

obj.method\_b()

obj.method\_c()

obj.method\_d()

## **❓ Interview Questions on Inheritance**

1. What is inheritance?
2. What are the types of inheritance in Python?
3. What is method overriding?
4. Can you inherit private members?
5. How does Python handle multiple inheritance?
6. Is multiple inheritance good practice?
7. What is the difference between inheritance and composition?

## **🧳 How to Add Inheritance to Your Resume (Project-Level)**

**Project Line Example:** "Built a modular School Management System using OOP principles like inheritance and encapsulation; implemented base User class and extended Student, Teacher, and Admin with role-specific behavior."

### **✅ Tip:**

* Show that you created **base classes** and reused logic using **inheritance**
* Mention **real-world models** (e.g., vehicles, users, products)
* Highlight **code reuse** and **extensibility**

## **🧠 Summary Table**

| **Inheritance Type** | **Description** | **Example Classes** |
| --- | --- | --- |
| Single | One parent, one child | Father → Son |
| Multi-Level | Chain of inheritance | Grandfather → Father → Son |
| Hierarchical | One parent, many children | Mother → Daughter, Son |
| Multiple | One child, many parents | Father + Mother → Child |
| Hybrid | Combo of above | Mix of all |

### **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>