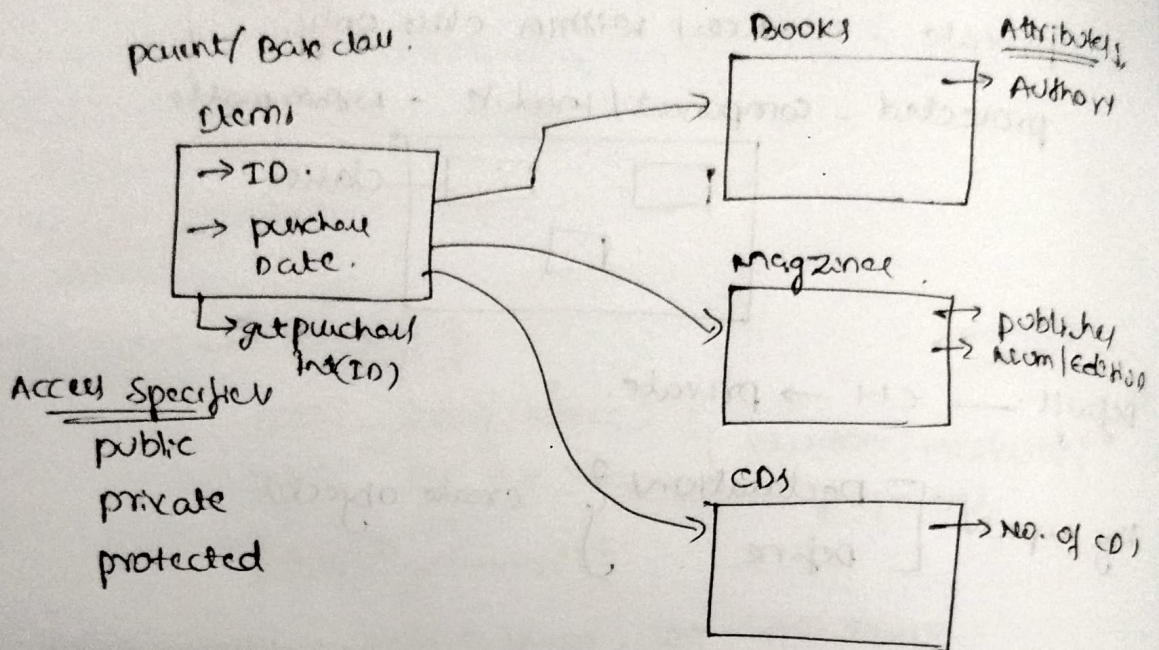Advance OOPs   12-09-2025

Inheritance — one class (child) to inherit the properties and
behaviour of another class (parent).

parent/ Base class.   Books   Attributes:
                                                          → Authors
    Items
   ┌→ ID ·   ┐
   ├→ purchase   │
   │ Date.   Magzinee.
   └→ get purchase   ┌→ publisher
      ht(ID)   └→ num/edition
Access Specifier
   public
   private   CDs
   protected   ┌→ No. of CDs

public — accessed to all · by all class

private — private variables or methods can be accessed only from
          the class in which they are declared and defined.
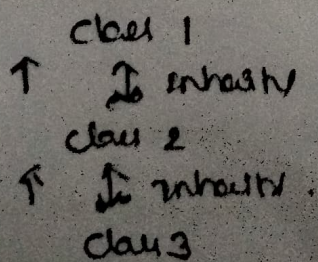
protected — only in derived class.
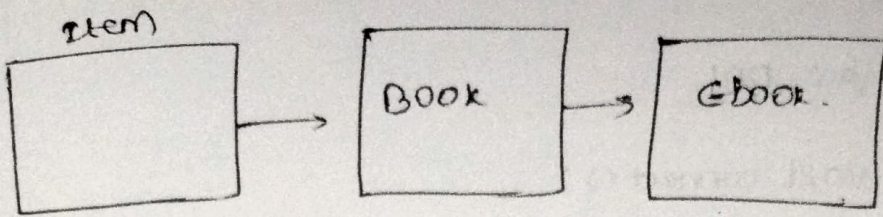
inheritance used "extends" key work.

@override = override the parent class function in child class.

method overloading
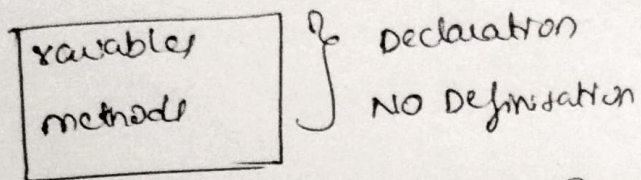   same function name with different size of inputs.
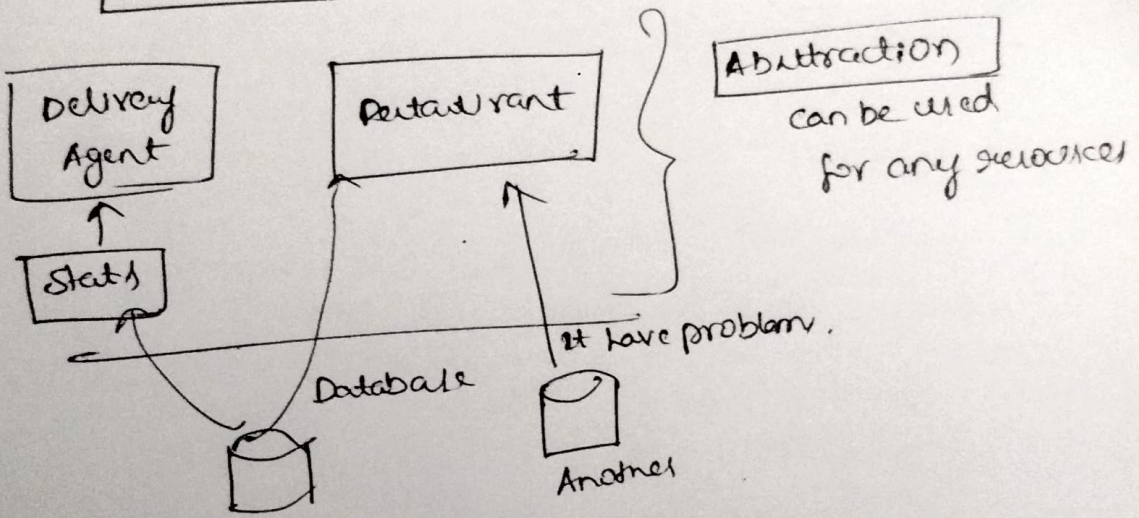
multilevel inheritance

        class 1
    ↑   Is inherits
        class 2
    ↑   Is inherits.
        class 3

Item

```
┌──────────┐         ┌──────────┐        ┌──────────┐
│          │ ──────> │  BOOK    │ ─────> │  GBOOK.  │
│          │         │          │        │          │
└──────────┘         └──────────┘        └──────────┘
```

multilevel inheritance,

Abstract

```
┌──────────────┐   }  Declaration
│  variables   │
│  methods     │      No Definidation
└──────────────┘
```

GI!

```
┌──────────┐        ┌──────────────┐    }   ┌─────────────┐
│ Delivery │        │  Restaurant  │    }   │ Abstraction │
│ Agent    │        │              │    }   └─────────────┘
└──────────┘        └──────────────┘    }     can be used
     ↑                   ↑    ↑         }        for any resources
┌────────┐              /      \
│ Stats  │             /        \
└────────┘            /          \
     ↑               /            │ It have problems.
                    /        ┌────┴───┐
         Database  /         │        │
              ┌────┴──┐      │        │
              │       │      └────────┘
              │       │      Another
              └───────┘
```
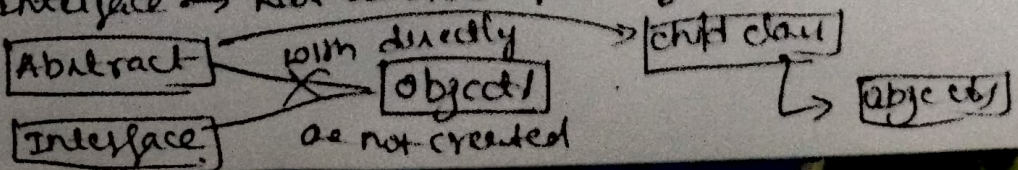
Syntax!

Abstract class

{

　　private ID;
　　public Name;   } Declaration
　　void search (string Name);

　　void printDetails ();
　　void print Details() → Defined.
　　{

　　}

}

interface → Not allowed for definition.

```
┌──────────┐     with directly    ┌───────────┐
│ Abstract │  ⟍               ──> │ child class│
└──────────┘    ⟋ ┌─────────┐     └───────────┘
┌──────────┐  ⟋   │ Objects │          ↓  ┌─────────┐
│ Interface│      └─────────┘             │ objects │
└──────────┘   are not-created            └─────────┘
```

interface DAL
{
    void connect();

    void getTable();

    void insert();

inheritance /   Sqlite extends DAL
by usage   {
to use he
clou   }
                }