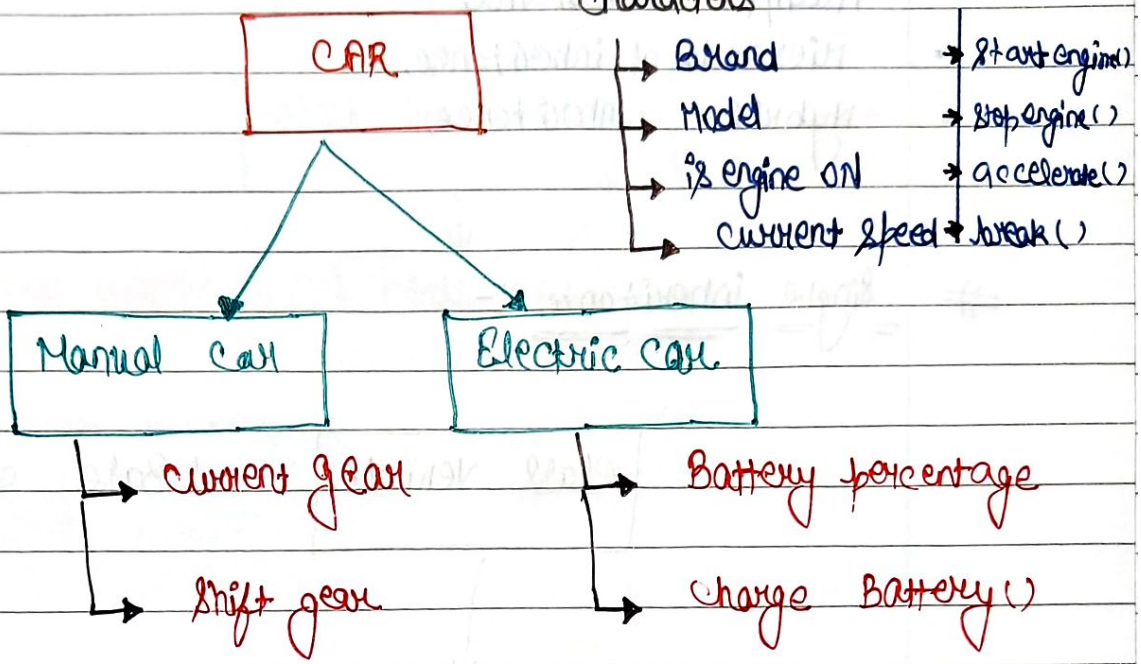
14-May-2025 Lecture-3

Inheritance :- The capacity of a class to derive properties and character from another class is called inheritance.

Example :-



* Parent class

```
class CAR {
    model
    Brand
    is engine ON
    ⋮
    stop engine()
    start engine()
    accelarate()
    ⋮
};
```

* Child1 class

```
class CAR : Pubic CAR{
    // shift Gear.() { }
    current Gear = 0;
};
```

* Child2 class

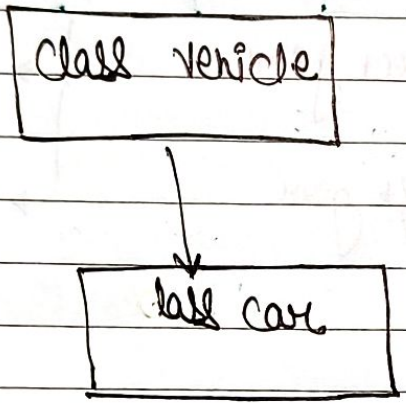
```
Electric CAR : Pubic CAR{
    //charge Battery
    // Battery Percentage.
};
```

__/__/____

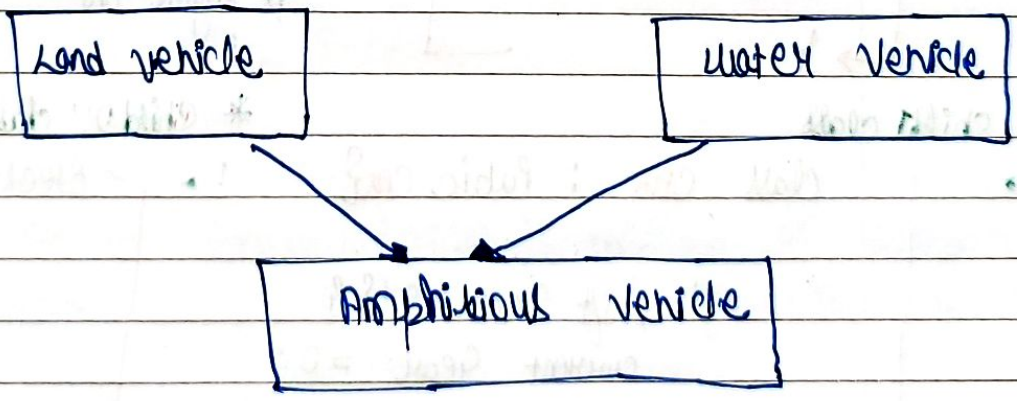
Type of Inheritance in c++ :-

- Single inheritance
- Multilevel inheritance
- Multiple inheritance
- Hierarchical inheritance
- Hybrid inheritance

Single inheritance :-

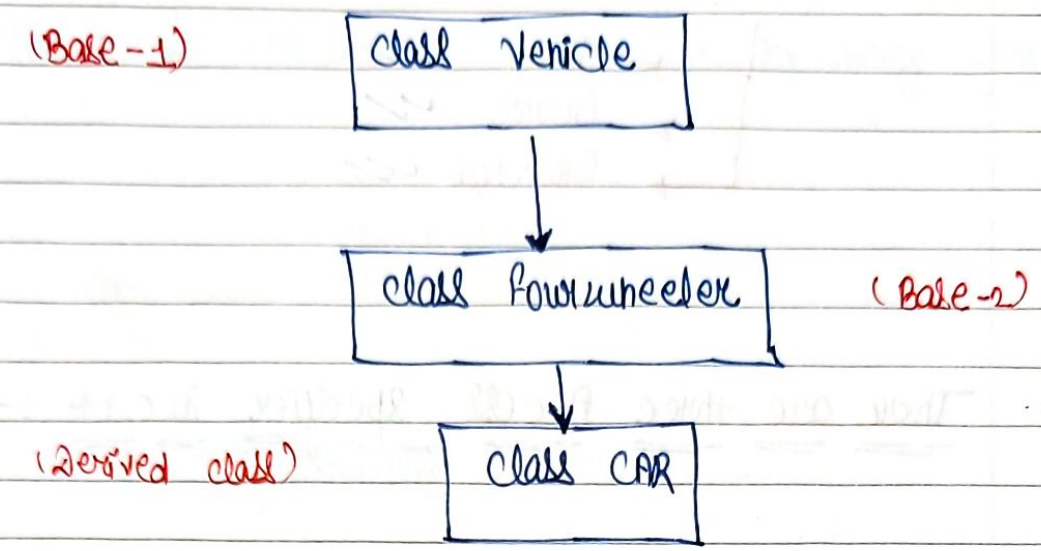


Multiple inheritance :-

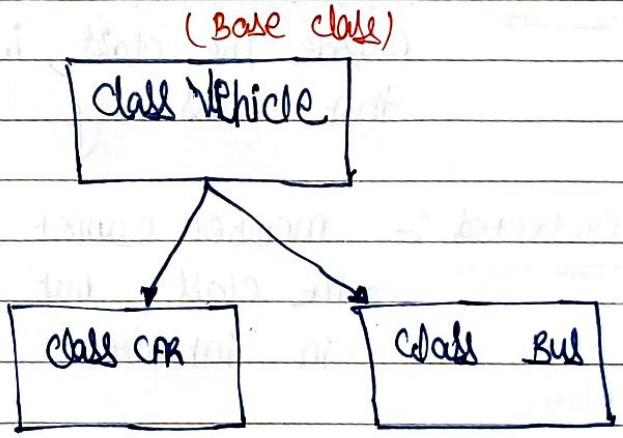


__/__/____

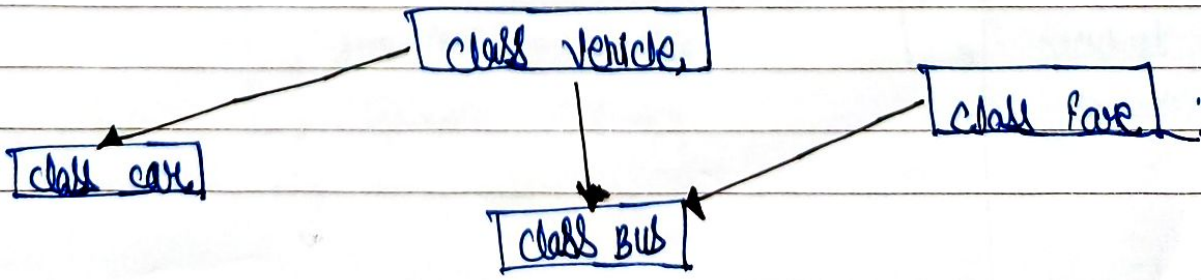
Multilevel Inheritance :-



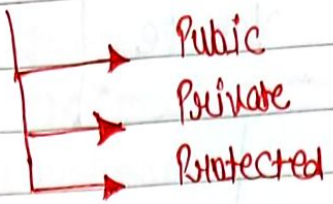
Hierarchical inheritance :-



Hybrid inheritance :-



Access Modifier :-



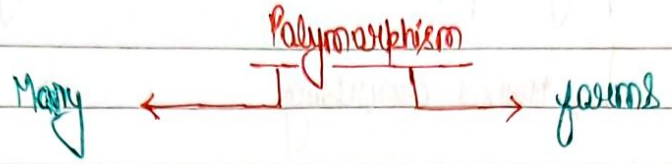
There are three Access Specifier in C++ :-

- **Public :-** member are accessible from outside the class, and member can be accessed from anywhere.
- **Private :-** Members cannot be accessed (or viewed) from outside the class, i.e member are private to that class only.
- **Protected :-** member cannot be accessed from outside the class, but they can be accessed in inherited class or derived class.

__/__/____

Polymorphism :-

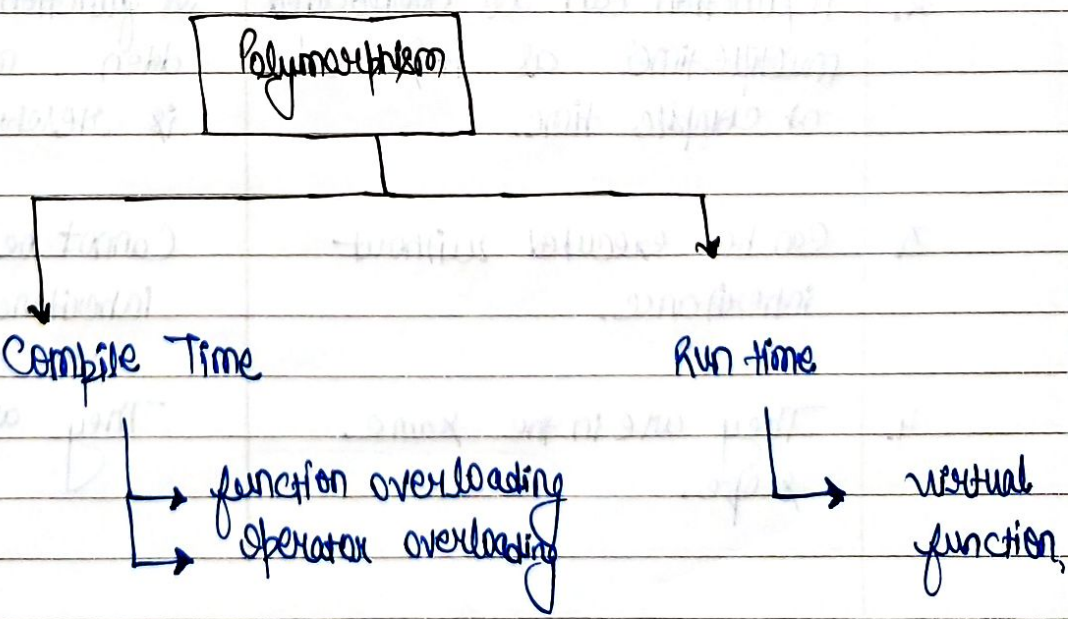
The word Polymorphism means having many forms.



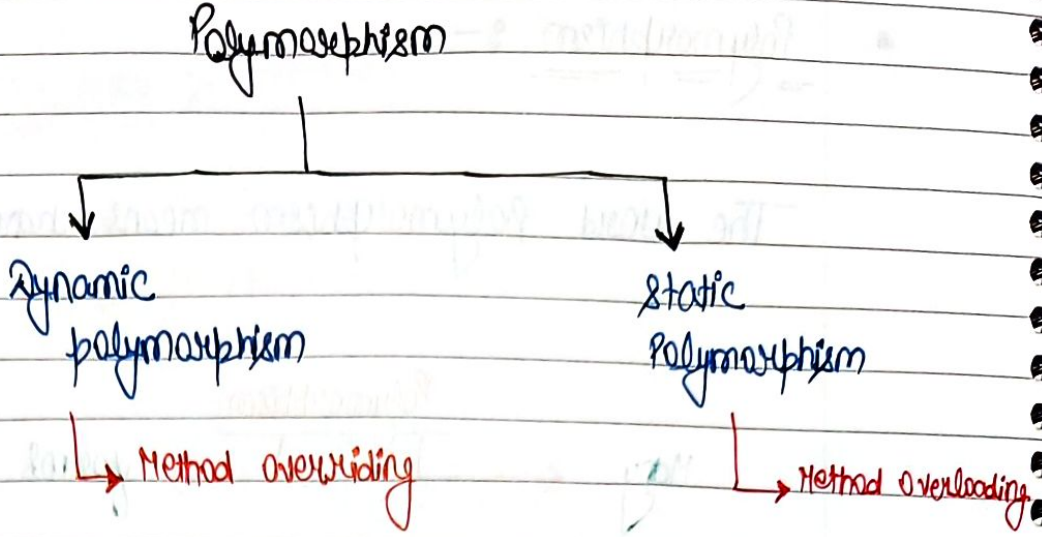
Type of Polymorphism :-

Polymorphism in C++ can be classified into two types :-

1. Compile-time polymorphism
2. Runtime polymorphism.



__/__/____



Function Overloading Vs Function Overriding

	Function Overloading	Function Overriding
1.	It falls under compile time polymorphism.	It can both compile Time or runtime polymorphism.
2.	A function can be overloaded multiple times as it is resolved at compile time.	A function cannot be overri-dden multiple times as it is resolved at runtime.
3.	Can be executed without inheritance.	Cannot be executed without inheritance.
4.	They are in the same scope.	They are of different scope