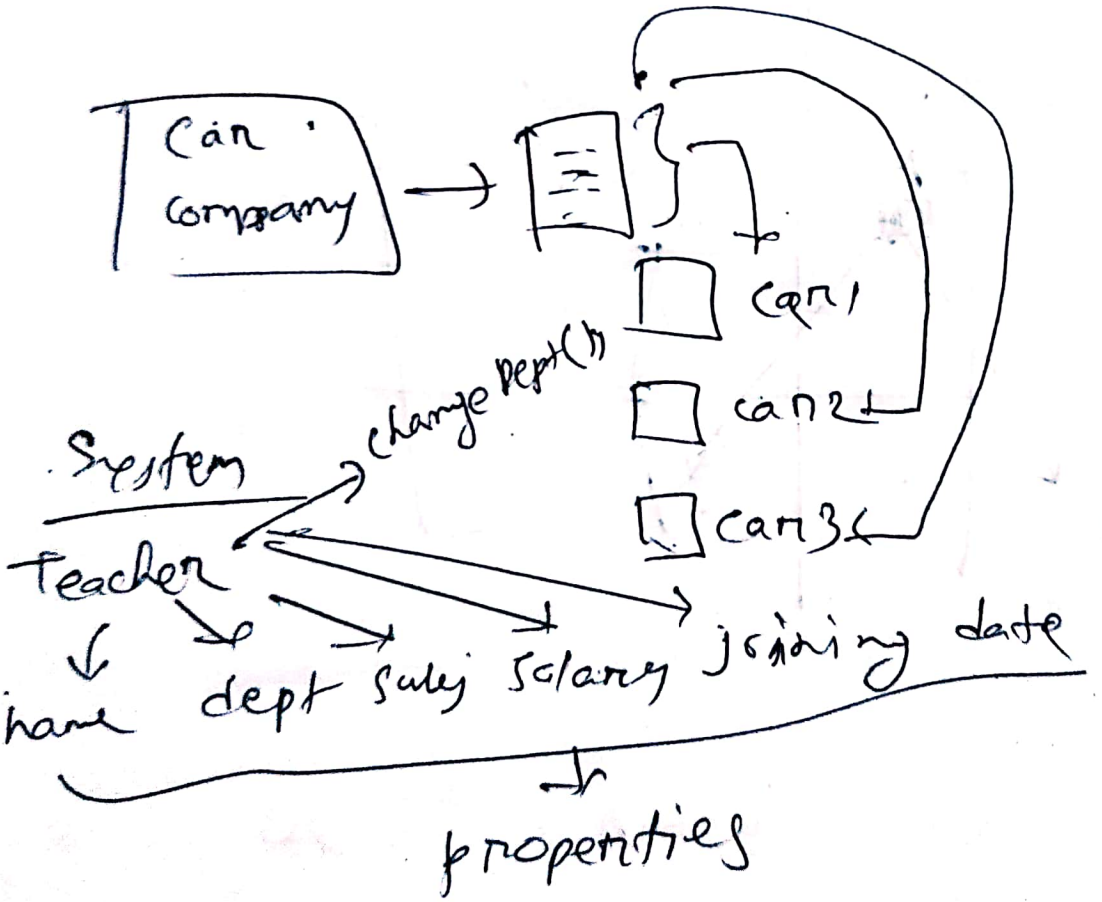
Object Oriented programming

Definition → eg

C++ → vector, string, stack } STL → created by OOPS.

* Objects are entities in real world.

* Class is like a blueprint of those entities



To use repeated fractor, we also use OOP

Access modifiers

private → data and methods accessible inside class

public → data and methods accessible to everyone

protected → data and methods accessible inside class and to its derived classes.

by default → attributes and methods are private.

Encapsulation

OOPS → Encapsulation
→ Abstraction
→ Inheritence
→ Polymorphism

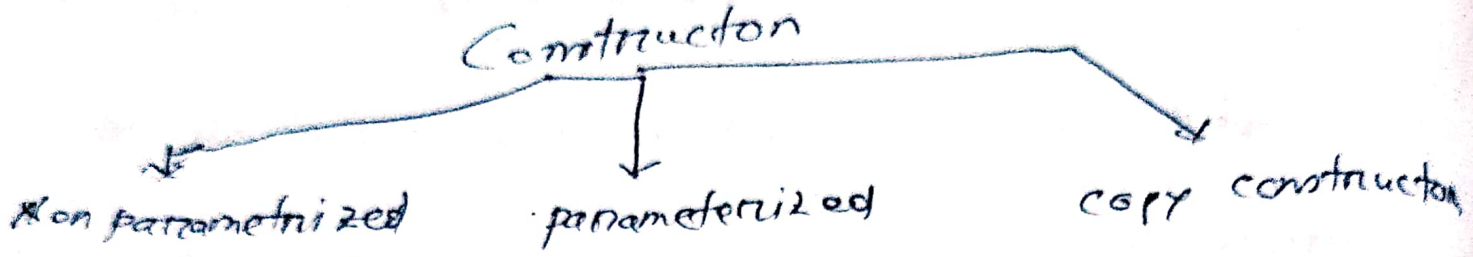
Encapsulation is wrapping up of data and member function in a single unit called class. It helps hiding data. (by using private modifier)

Constructor

special method invoked automatically at the time of object creation. Used for initialisation.

* Same name as class.
* Constructor doesn't have a return type.

* Only called once (automatically), at object creation
* Memory allocation happens when constructor is called.



* Construction overloading: A class can have differe
constructing by putting distinct parameter to each of it. I
is also an example of polymorphism.

This

this is a special pointer in c++ that points to the
current object.

this→prop is same as *(this).prop

* Copy constructor creates two type of ~~object~~.cop

Shallow and Deep copy:

A shallow copy of an object copies all the memb
values from one object to another.

A deep copy, on the other hand, not only copies the member values but also makes copies of any ~~dynamical~~ dynamically allocated memory that ~~too~~ the member points to.

Destructure → Destructuring/clearing the memory.

~ className ( ){ }

Inheritance

When properties and member functions of base class are passed on to the derived class.

* Used for code ~~reusability~~ reusability.

Mode of inheritance

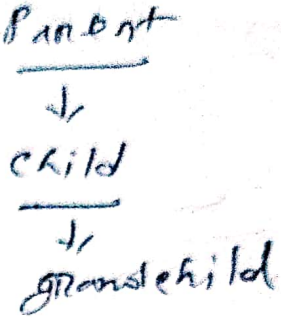
	Derived Class	Derived Class	Derived Class
Base Class	Private	Protected	Public
Private	Not Inherited	Not Inherited	Not inherited
Protected	Private	Protected	Protected
Public	Private	Protected	Public

* Protected use to give permission derived class to use.

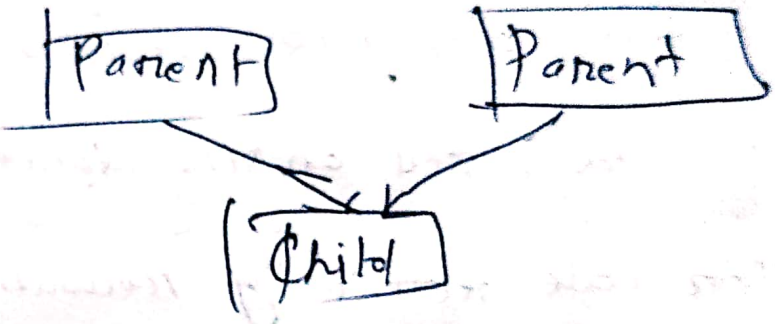
Types of Inheritance

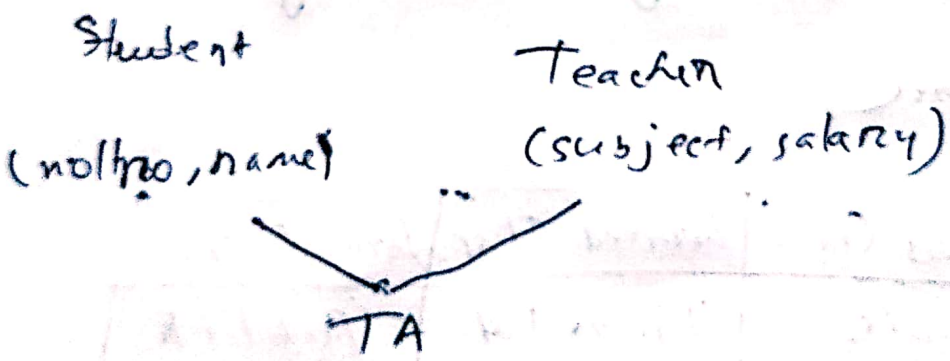
* Single Inheritance

* Multilevel Inheritance

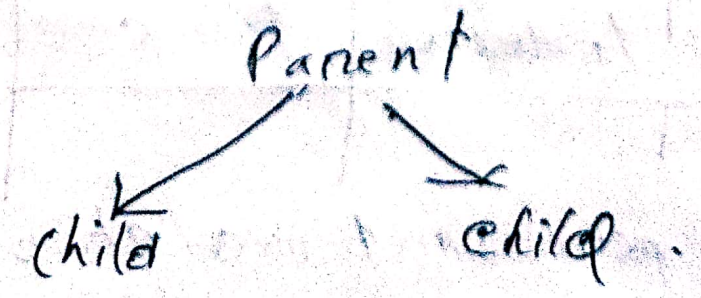


Multilevel Inheritance

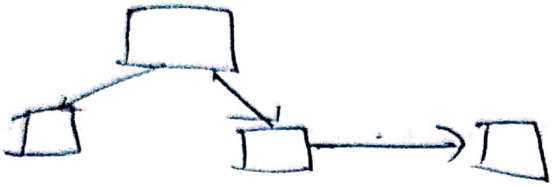


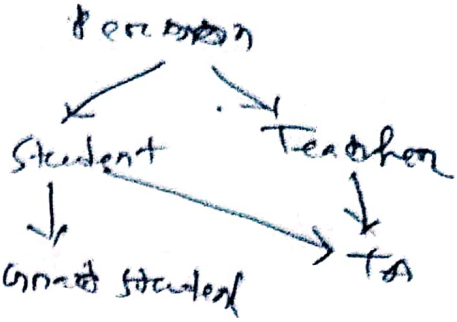


* Hierarchial Inheritance



Hybrid Inheritance





Polymorphism

Polymorphism is the ability of objects to take on different forms or behave in different ways depending on the context in which they are use

* Compile time (Static) polymorphism → construction overloading
* Runtime (Dynamic) polymorphism.

Compile time polymorphism:

Function overloading

class { * no. of parameter * type of parame

func (param)

func (param)

}

Operator overloading

C++ allows you to specify more than one definition for an operator in the same scope, which is called operator overloading.

Runtime polymorphism

function overriding

Parent and child both contain the same function with different implementation. The parent class function is said to be overridden.

Virtual functions

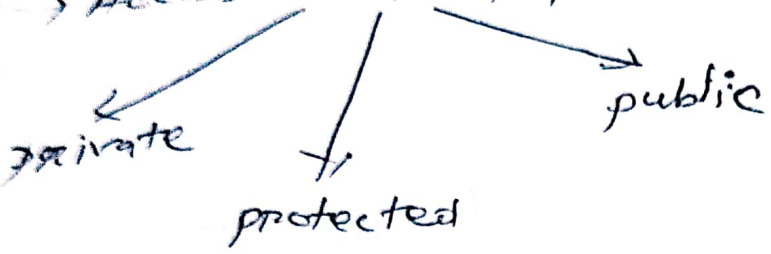
Virtual function is a member function that expect to be redefined in derived class

* Virtual functions are dynamic in nature.
* Defined by keyword "virtual" inside a base class and are always declared with a base class and overridden in child c
* A virtual function is called during runtime.

Abstraction

hiding all unnecessary details and showing only the important parts

3 Access modifier/specifier



Using abstract classes

* Abstract classes are used to provide a base class from which classes can be derived.

* They cannot be instantiated and are meant to be inherited.

* Abstract classes are typically used to define an interface for derived classes.

*** If any class has pure virtual function, it automatically gets converted to abstract class.

***pure virtual function is a virtual function which is defined 6

virtual void a() = 0;//pure virtual function

Static keywoted

Static variables

Variable declared as static in a functi
are created and initialised once for the lifetime of
the program.

Static variables in a class are created and initialised once. They are shared by all the objects of the class.

static objects

It remains till program's lifetime.