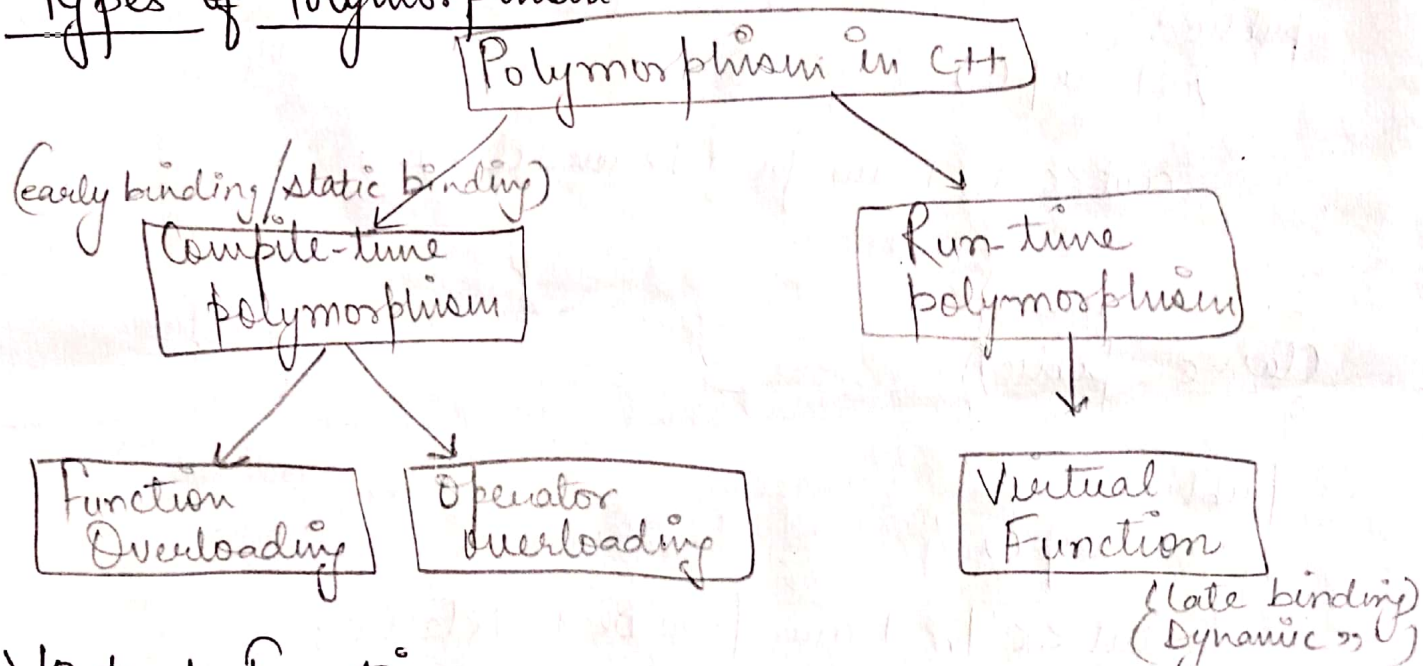
Polymorphism

It is the ability to use an operator or method in diff. ways.

Polymorphism is a mechanism that allows you to implement a function in different ways.

Polymorphism plays an important role in allowing ~~you to implement~~ objects having diff. internal structures to share the same external interface.

Types of Polymorphism



Virtual Function If there are member functions with same name in base class and derived class, virtual functions gives programmer capability to call member function of diff. class by a same function call depending upon different context. It supports runtime-polymorphism. OR

A function declare with Virtual keyword in base class redefine in derive class. Virtual function must be call by base class pointer.

```
# include <iostream>
Class A
{
public:
Virtual void disp() // Parent Class Function
    {
     cout << "\n I am from Base class A";
    }
};
Class B: public A
    {
  public:
     void disp() // Child class function
      {
       cout << "\n I am from Derived Class B";
      }
};
Class C : public A
  {
  public:
     void disp() // Child Class Function
      {
       cout << "\n I am from Derived Class C";
      }
};
int main()
{
  A *pa;    // Base class pointer
  B objb;   // B class object
  C objc;   // C class object
  pa = &objb; // address of derived Class B
  pa -> disp(); // calling disp of class B
  pa = &objc;  // address of derived class C
  pa -> disp(); // calling disp of class C
  return 0;
}
```

means base ptr. points

Pure Virtual Func^n.

```
virtual void disp() = 0;
```

PURE VIRTUAL FUNCTION (Do nothing function)

→ It is a specific type of virtual functions.

→ A virtual func? with no function body is called Pure Virtual Function.

→ A pure virtual func? purely exists in base class only to be overriden by the derived class functions.

PURE is not a keyword.

Example

```
CLASS A
public :
virtual void display ( ) = 0;
```

← for pure virtual fun

→ It is also called "Do notting function."

→ You cannot define this function in this class, therfore it is called Pure Virtual.

But

You have to define that class in all derived classes (it is compulsory)

ABSTRACT CLASSES

A class that contains Pure Virtual Function is called as Abstract class.

An abstract class is one that is not used to create any object of its own but it solely exists to acts as a base class for other classes.

Imp: WE CANNOT CREATE OBJECT OF ABSTRACT CLASS

But

WE CAN CREATE POINTER OF ABSTRACT CLASS

Example is same as Virtual & Pure Virtual Func?

Function Overriding –

We know that the functionality of virtual functions can be overridden in its derived classes.

To cause late binding to occur for a particular function, c++ requires that you use the virtual keyword when declaring the function in the base class.

To create a member function as virtual, you simply precede the declaration of the function with the keyword virtual.

If a function is declared as virtual in the base class, it is virtual in all the derived classes. The redefinition of a virtual function in a derived class is usually called FUNCTION OVERRIDING.

RULES FOR VIRTUAL FUNCTION

1. Virtual func. must be member of same class.
2. They cannot be static member.
3. They are accessed by using object pointers.
4. Virtual func. can be friend of other function.
5. Virtual func. in a base class must be defined, Even though it may not be used.
6. The corresponding functions in the derived class must agree with the virtual function's name and signature that means both must have same name and signature.