Chapter 11

- **Polymorphism:**

  -different form of different functions in different class but All of them have same name

  EX.

  BASE* ptr; **Drive : public BASE**

  Drive d1;

  Ptr =& d1;

  Note : Drived class always has address to Base class .

  . if there function in Drived and Base class with same name The compiler always execute function in Base class as Drived class point on

  To solve this problem,

- **Virtual Function :**

  Form ---> Base class

  Virtual Return value func.name { }

  So if we call function from Drived class and there class with same name in Base ---> compiler will execute function in Base class

  EX.

  Base ---has function--> show{ }

  Drived ---has function--> show{ }

  Base* ptr;

  Drived d1; ptr=d1;

  Ptr -> show( ); show of Drived will execute

  -In General : if you point to any function found in Base and Drive

---> poiner of type Base will point to Drive function As this function is virtual int it

- **Pure virtual function . #p.g 512 in book**
  - **You need only pure function in class to make it abstract (can't instantiate object from it)**

Form---> virtual Returnvalue function( )=0;

. in pure virtual function never write a definition & it should declare in All drived class

Explain ---> pure virtual function As we make form for all Drive class and call all func. Of drive calss from Base class

- **Scenario for that Base* ptr[100];**

  Ptr[1]=new Drive1;

  Ptr[2]=new Drive2;

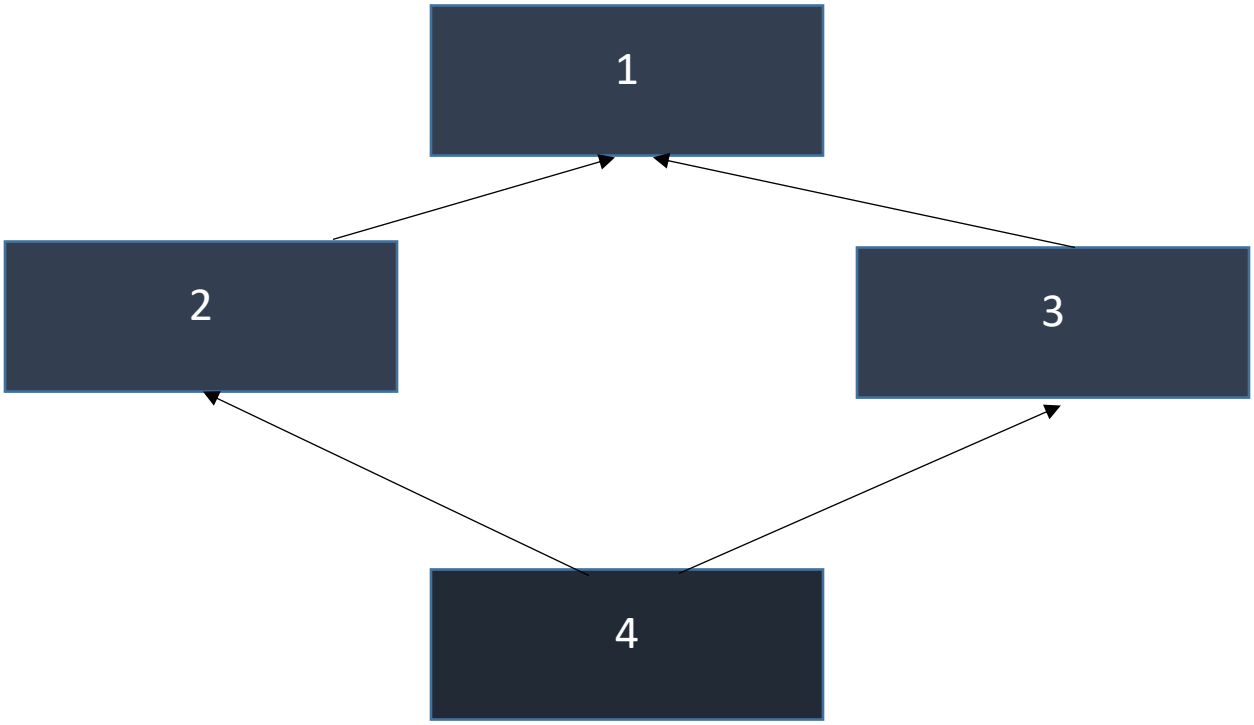
  Ptr[1]-> show( ) Excute show of D1

  Ptr[2]->show( ) Excute show of D2

- **Virtual Distructor**
  Always Distructor is virtual to allow Distructor excuse it's class
  With out virtual Distructor compiler will execute Base Distructor

Virtual Function	Pure Virtual Function
Member function declared within Base class & redefined by derived class when call by pointer of Base class to Drive it will execute function in drived class	**Virtual function (abstraction function),don't have implement, we only declare it and implement with suitable implement in every drived class**

- **Virtual Base class:**



// compiler error

-Bec.2,3 drive from1 so each of them have unique copy of 1 so 4 can't detect which copy it should be use

Solve this problem:

- Virtual Base class

  So 2,3 have same copy of class1 so 4 Refer to one copy only of class1

  Form--> class2: virtual public 1

  Class3: virtual public1

- Friend function:

  When we want to access data from another class to function of our class we work in it.

  -we define function as friend in which class we want to access this data.

  Class B;

  Class A

  { friend func( ) }

  Class B

  Func( ) ---> can access data of A

  Note: we should define class B above A to can define it's func in class A

  Note : friend ---> write in decleration only .

- Friend class:

  Class Alpha

  {

  Friend class Beta;--->Beta can access All private data of Alpha

- Static function = static Data member
  - Can't call by obj --> so it call with class name in main
    Class name :: static function_name
    In class----> static return type func ( ) { }

  - It make operation with class in General not to object of class

- Overloading Assignment operator

  Form---> class name operator = (class name& a)
  {
  Return class_name( )

  Obj1=obj2 ----> Assignment
  Obj1(obj2)----> copy intalization

  -Note : Assignment operator Not inherited

لتجنب اننا نعمل نسخة من ال

Obj

ال بدورها تكلف وقت ومساحة

- Copy constructor:
  Form---> class A {
  A(A& x) ----> copy constructor
  Data of A = data of x
  }

الهدف منها انك ممكن تحتاج تعمل نسخة من الأوبجكت ل أوبجكت تانى ولكن بشكل معين عن الأفتراضى الخاص بالكوبيلر

-Note ---> we should pass by Reference to avoid constructor from calling itself that will make crash in memory

- This pointer :

  -point to object itself (carry pointer of obj)

  - help use in Return by Reference that is better than Return temprory object which will cost memory & time

  EX.

  Return A(x,y) Return *this;

  Before After