OOP Course

Created by : Israa Abdelghany
LinkedIn : www.linkedin.com/in/israa-abdelghany-4872b0222
GitHub : https://github.com/IsraaAbdelghany9

Lecture
Day - 7

* very important
* assign by value ⇒ normal =
* in intialization ⇒ e1=2 ⇒ copy Constructor
* Constructor can be private
* e1 = e2 ⇒ Copy Constructor
* function returning value has a temp object deleted after after the return. and the objects are deleted after exit the Function
* sending object to function ⇒ Composion relation

base = child ⇒ No error ⇒ object of parent = object of child
will take a copy of parent from the child and assign it in the <base>

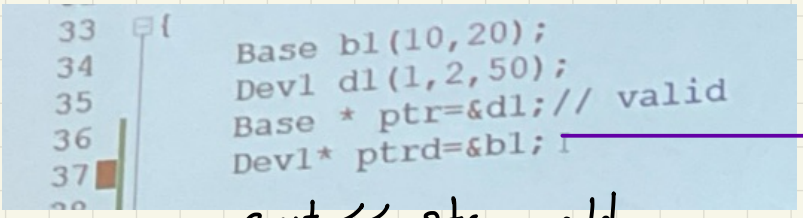
child = base ⇒ will give error <variable will be missed>
even if there's no missed variables

pointer of base point to object of base

```
Base * ptr = &b1;
cout << ptr->add();
```

pointer of child point to object of child

```
Base * ptr = &d1;
cout << ptr->add();
```



→ Compilation error

step of pointer is as child not parent

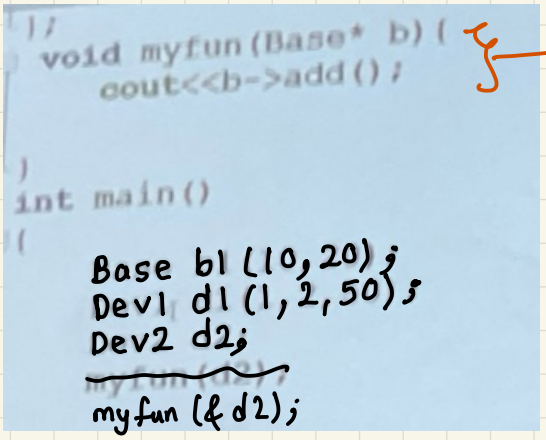
cout << ptr -> add

call add of the pointer type

★ Virtual :- الباطن مش الظاهر

- will look at what is the type of the object the pointer is pointing to and call its function
  - this is a type of polymorphism
  - virtual for the parent.
  - b1 = d1

    cout << b1.add() ⟹ will look at the drieved value not actual one which is the child here
  - if I deleted virtual no error but will call parent function each time.
  - if the child don't have function override it will call parent function ⟹ no error



→ general function

→ function of pointer

```
Base* arr[3] = {&b1, &d1, &d2};
for (int i=0; i<3; i++)
{ cout << arr[i]->add(); }
```

OR

```
Base& arr[3] = {b1, d1, d2};
for (int i=0; i<3; i++)
{ cout << arr[i].add(); }
```

error arr decleration

- virtuality can't be stoped in C++ but other higher languages can
- abstract class must have at least one pure virtual class

  void add() =0
- if the child class don't have override of the parent virtual function it will be considered as abstract class ⇒ No objects can be created from it

- Base * ptr ; (valid)
  ptr = new Dev2();

  cout << ptr → add()

Base* ptr = new Base() ⟹ Compilation error Because of abstract

```
// without virtual
//static binding - early binding
Base* ptr;//=new Base();
ptr->add();
ptr=&d1;
// with virtual , pointer of parent -> child
//dynamic binding , late binding
```

during runtime

interview question what are the diffrences between static and dynamic binding?

Static binding (VS)	dynamic binding
early binding	late binding
before	during run time
calling parent function	child function

virtual function

virtual int fun() { return 0; }

1- not pure but will do nothing

2- Can create object from its class

→ since it is virtual if pointer or object will not be diffrent

→ if not virtual will return value zero.

virsual int fun() = 0;

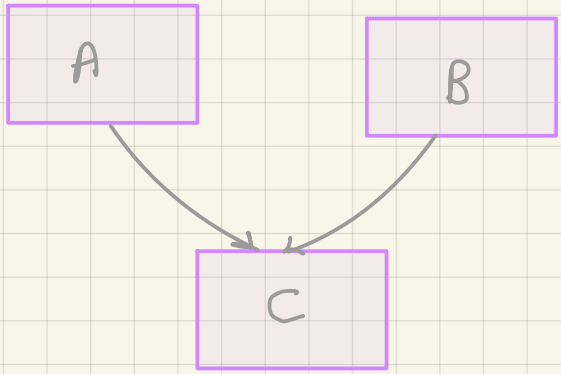
1 pure virtual function

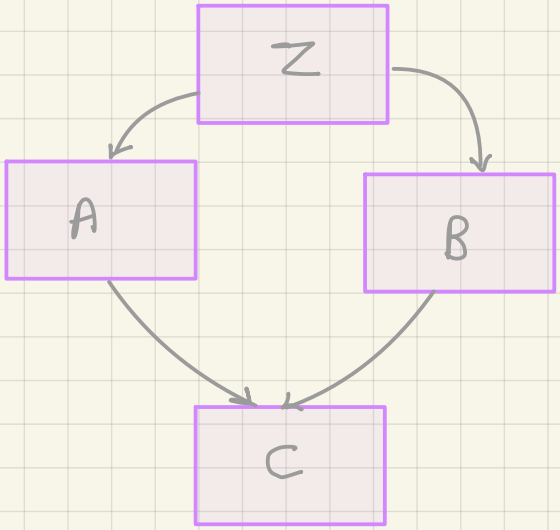
2 can not create objects from its function

↳ Compilation error

So to solve that create array pointer containing its childeren ⇒ shape *arr [3] = {c1, r1, t1};

multi level inheritance:- class C : Public A, Public B





multiple Inheritance had problems
So removed from other higher languages
(will get values of z once by A and
, z, y, c, another by B

what happen if virtual function is in one of the children not the parent

```
class Base
{
    int x;
    int y;
  Public:
    void print() {}
};
```

```
class A : public Base
{

}
```

A will inherit Both Private and public members as they are and inherit them to any child of them and main can see and access the public members

```
class B : protected Base
{

}
```

B will inherit Both Private and public members but Public members as protected which means inherit them to any child of them and main can not access the public members (now they are protected)

```
class C : private Base
{

}
```

B will inherit Both Private and public members but Public members as private which means can't inherit them to any child of them and main can not access the public members (now they are protected)

these access modifiers works mainly on Public members of the parent which affect children & main function (outside the inherited classes)

Compilation errors:-

this -> id = 10; in stand alone functim

int this; ⟹ key word

complex ⟹ key word so we use Complex instead

scanf (%d , *_ptr) ⟹ should be without *