OOP Course

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

Day _ 6

is_a >> Inheritance << kind of

الوراثة

State Vs Identity.

Point p1 (20, 30)

Point P2(40, 50)

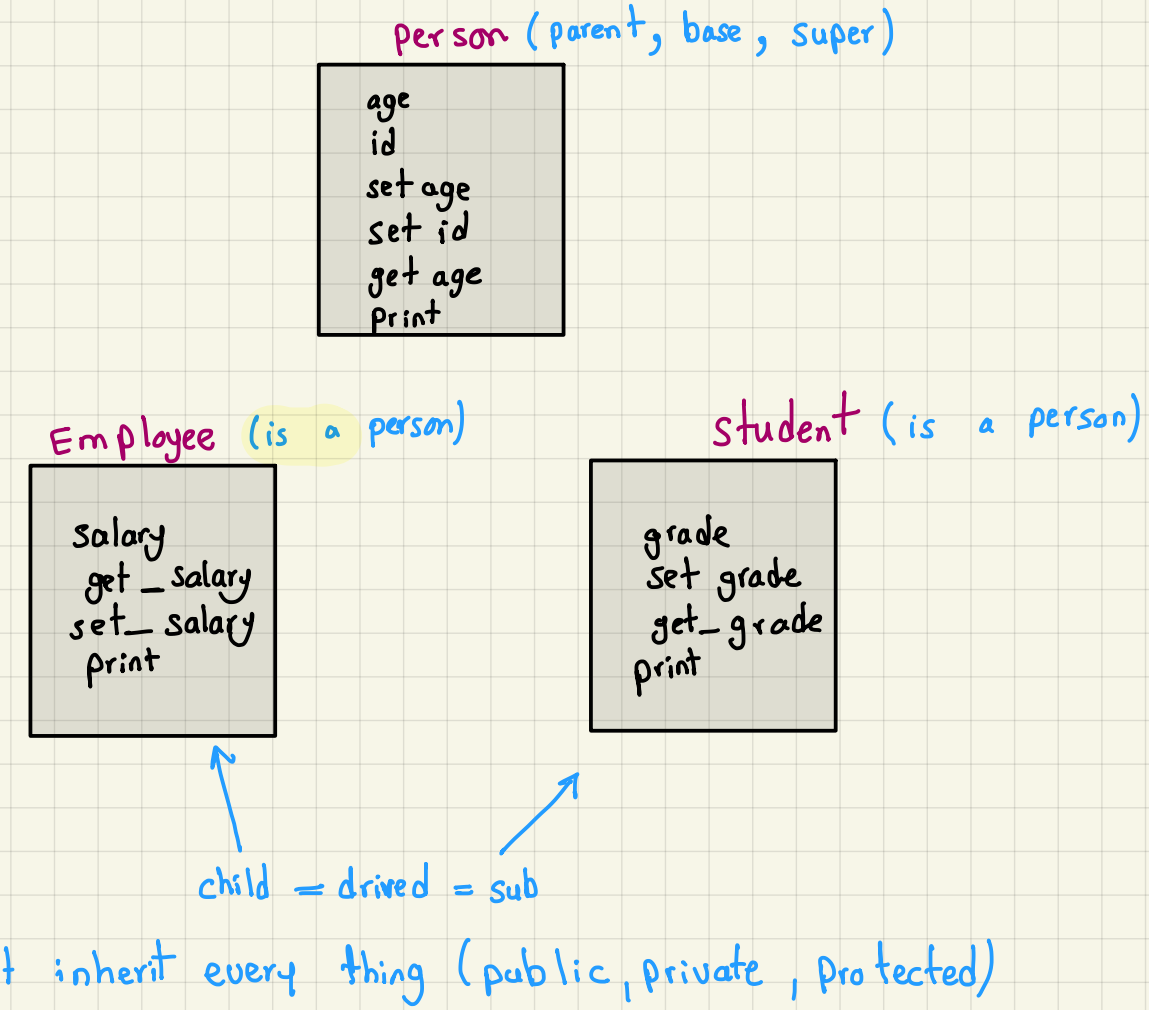
State (values)

Location in the memory is the Identity

Point * p = &p1;

Point * p3 = &p1;

} reffer to same address so they have same identity



it inherit every thing (public, private, protected)

- if I have a dog class I can not inherit it from Person (logical error)
- private members of the parent are not accessable by childs but it can be accessible if protected instead of private
- Point *p is not an object
- if member is protected I can access it in the child using this ⟹ cout << this -> ID << endl ;

```
// the problem here is that the parent should be created before the child so when I call this constructor
// it calls the parent constructor(parameterless at first then call this constructor)
// so it is reassigning the values for the same object twice
//  and if the parent do not have a parameterless constructor it will give compilation error
   emp(int _ID , int _Age , char *_name )
   {
       setID( _ID ) ;
       setAge( _Age ) ;
       setName( _name ) ;
       salary = 3000;
   }
```

TABLE 9.1 Inheritance and Accessibility

Access Specifier	*Accessible from Own Class*	*Accessible from Derived Class*	*Accessible from Objects Outside Class*
public	yes	yes	yes
protected	yes	yes	no
private	yes	no	no

when to use Inheritance.

- if I don't have the code only object file I can create a child of class of the obj
- if I have good class that I want to add new feature it is preffered to create a child of it to save

adv :- . reusability
- take data from code and build on it in another class
- generalization myfun (Person s) ⇒ can take emp or student too
- create child for the object file and can override later

protected member let child access the data of the parent.

private members can be توريث

constructor يورث ✗ ⇒ compilation error

access modifiers:-

public
private
protected

* if parameterless constructor of the parent is not availble ⟹ Compilation error

* parent is called at first then child

* Composition relation ⟹ part then whole.

* Constructor & destructor ⇒ opposite to construction child then parent

Code example :-

```
Class Person
{
    int id;
    char name[20];
    int age;
  Public:
    void setId(int _id) {id = _id;}
    void set name(char *_name) { strcpy(name, _name);}
    void setAge (int _Age) { age = _Age;}
    int get id(){ return id;}
    char* get name () { return name;}
    int getage() { return Age;}
    print
```