OOP Course

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

Day 5 (online)

to day schedule :-

- static members , object members
- array of objects
- array of pointers to objects
- object relations (Composition , aggregation , association , inheritance)

- static → make the member → class member instead of object member

---

- static function inside class → can call it without object
  → cout << Complex :: getCounter();
  and if you tried to print any other member of object
  └ compilation error because static functim doesn't have this pointer
  Ex:- static int getCounter() return Counter;
  if trying to add cout <<real ⟹ Compilation error

- Even if counter is static it's still private so you can't access it directly from main function ⟹ Compilation Error
  you need getCounter functim. → this function need to be static to be able to call it without object

```
int main ()
{
    {
        Complex S1 (30);   ⟶ this object will be created here
    }   ⟶ deleted when exit this {
    return 0;
}
Public:
  static getCounter ()
  {
      real = 60;   ⟶ Compilation error (static fun)
      return Counter;
  }
```

- any function can
- :: ⇒ scope operator (scope resolution)

  it will not be available in higher languages

- if C4.getCounter ⇒ Compilation error

because the higher languages consider it as compilation error

- Array of objects in stack :-

Complex arr[3];

Coding Example :-

```
int main()
{

    Complex arr[3]={Complex(10,20)};
    for(int i=0;i<3;i++)
        arr[i].print();

    cout<<Complex::getCounter();

    return 0;
}
```

⟹ initialized the first obj.

OR →

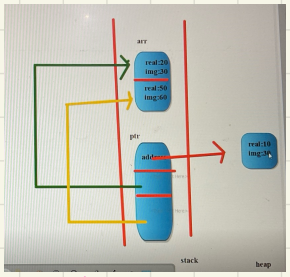
```
Complex arr[3];
arr[0].setComplex(20,30);
arr[1].setComplex(50,60);
arr[2].setComplex(70,80);

for(int i=0;i<3;i++)
    arr[i].print();

cout<<Complex::getCounter();
```

- Note Complex * ptr ; it's not an object it's pointer of type Complex (Counter still 0)
- Same if Complex * ptr[3]; it is array of pointers No objects (Counter = 0)

```
int main()
{
    Complex * ptr[3];
    ptr[0]=new Complex(10,30);
    ptr[0]->print();
```

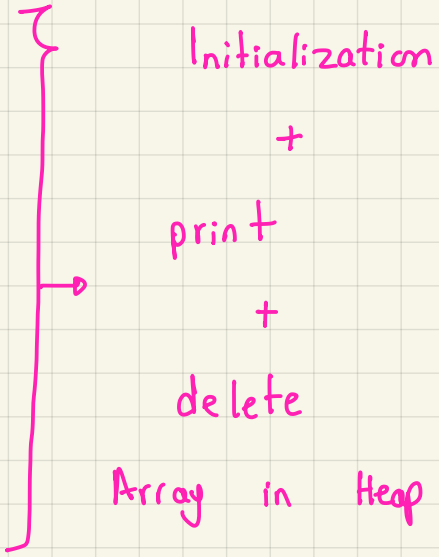
```
int main()
{
    Complex * ptr[3];
    ptr[0]=new Complex(10,30);
    ptr[0]->print();

    Complex arr[2];
    arr[0].setComplex(20,30);
    arr[1].setComplex(50,60);
                = arr
                = arr+1
    ptr[1]=&arr[0];
    ptr[2]=&arr[1];
    ptr[1]->print();
    ptr[2]->print();
    delete ptr[0];
```

```
"C:\Users\MAS-ALHassan\Des
10+30J
20+30J
50+60J
3
 object deleted
 object deleted
Process returned 0 (0x0)   execution time : 0.225 s
Press any key to continue.
```



↳ • here first Complex is in the heap but the other 2 are in the stack

```
int Complex::counter=0;
int main()
{
    Complex * ptr[3]={
        new Complex(3,4),
        new Complex(5,6),
        new Complex(7,8)
    };
  for(int i=0;i<3;i++)
        ptr[i]->print();
  for(int i=0;i<3;i++)
    delete ptr[i];
```



delete [] ptr ; ⇒ will not delete the array

• Relations

we have 4 relations (part , whole) (جزء , كل)

1. wall _ room : Composition : strong agregation :Consist of

• the part may belong to one whole , the life time of part &whole is the same (علاقة القلب بالإنسان) (الجدار والغرفة) (الاثنين اعتماد كامل على بعض)

2. team _ match : agregation : weak agregation

• if match is ended team will remain but the match depend on team. (احد الأطراف يعتمد على الاخر)

• Uni -direction

3. student _ instructor : association : deal with

• don't depend on each other and not part of

• this relation performed using pointers reffering to the other one

• Bi_direction

4. Inheritance : kind of (is a)

```
class A
{
}
Class B// whole
{
        int x
        A a1; //part

}

class match{
  team* a;

}
c c1
```

- in Rectangle Class ⇒ points are created first then Rect object

- But in destroying the Rect obj ⇒ Rect is deleted first then points

  - Rect (int x1, int y1, int x2, int y2) : ul (x1, y1), rl (x2, y2)

Complex * ptr OR Complex * ptr[3]

in both Counter will remain 0 because it's pointers not objects

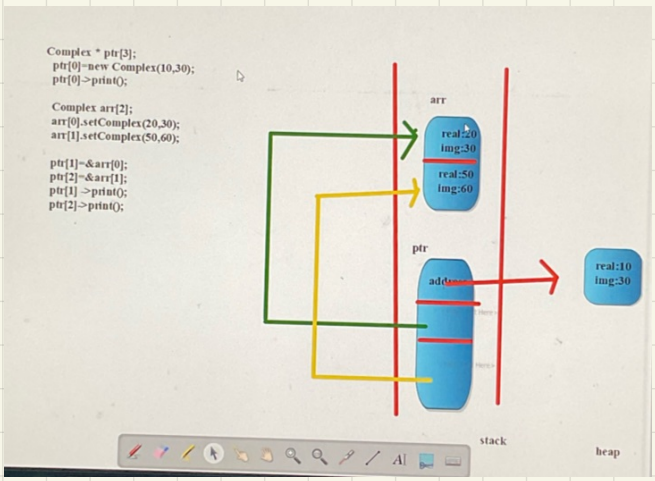
```
Complex * ptr[3];
  ptr[0] = new Complex(10,30);
  ptr[0] -> print();
```

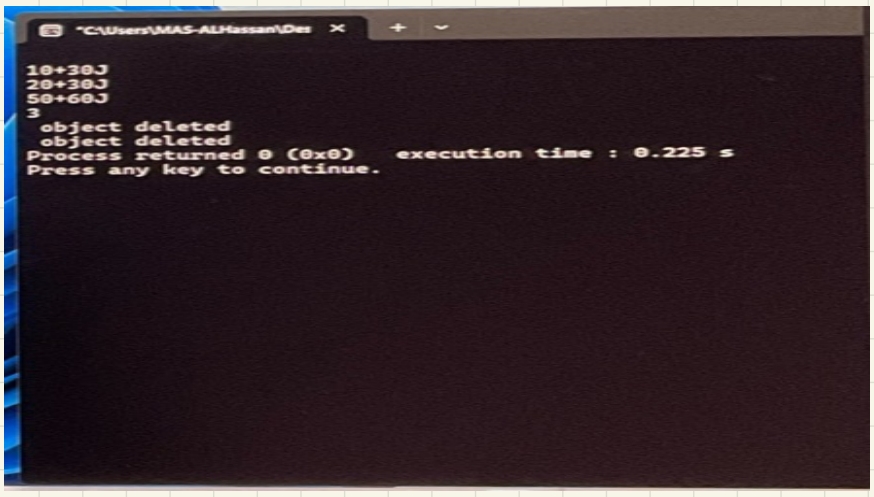
* Counter = 1

```
Complex arr[2];
arr[0] = setComplex(20,30);
arr[1] = setComplex(50,60);
ptr[1] = &arr[0]; //=arr
ptr[2] = &arr[1]; //arr+1
  ptr[1] -> print();
  ptr[2] -> print();
```

* Counter 3 #

```
delete ptr[0];
```





```
int Complex::counter=0;
int main()
{
    Complex * ptr[3]={
        new Complex(3,4),
        new Complex(5,6),
        new Complex(7,8)
    };
 for(int i=0;i<3;i++)
        ptr[i]->print();
 for(int i=0;i<3;i++)
   delete ptr[i];
```

```
int Complex::counter=0;
int main()
{
    Complex * ptr[3]={
        new Complex(3,4),
        new Complex(5,6),
        new Complex(7,8)
    };
 for(int i=0;i<3;i++)
        ptr[i]->print();
 //for(int i=0;i<3;i++)
   delete[] ptr;
```

don't delete

```
"C:\Users\MAS-ALHassan\Des
3+4J
5+6J
7+8J
Process returned 0 (0x0)   execution time : 0.221 s
Press any key to continue.
```

4