Section 17 : Friend and Static Members / Inner Classes

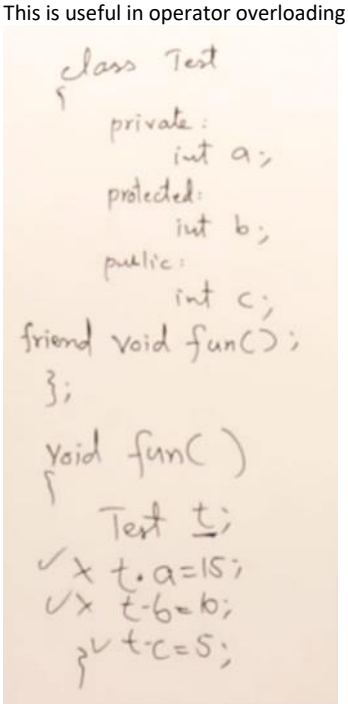
15 April 2022 16:49

© Rajat Kumar
https://www.linkedin.com/in/imRajat/
https://github.com/im-Rajat

Section 17 : Friend and Static Members / Inner Classes

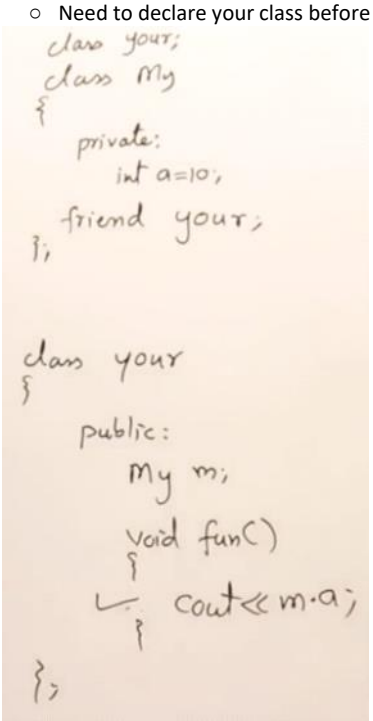
Friend Functions :

- Friend function is a global function outside function which can access all the members of a class upon object, not directly u pon objects only. Then it can access private, also protected also public.
- Also class should say that that function though doesn't belong to us, but it's a friend for us so it can access all the membe rs of our objects.
- This is useful in operator overloading mostly.



Friend Class :

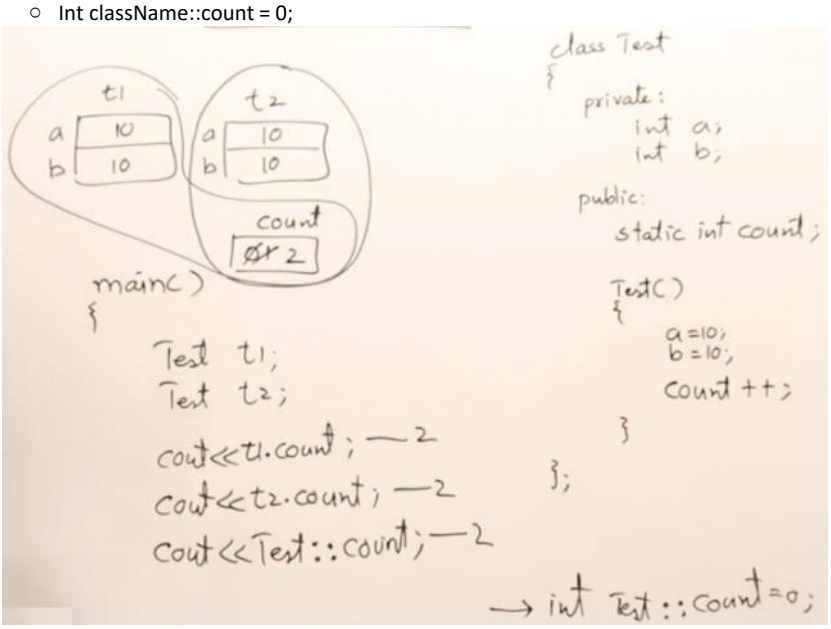
- If we want one class to access private members of another class upon object without inheritance, then we have to declare the class as a friend inside the other class as friend.
  - Need to declare your class before my, otherwise compiler will not recognize your class.



- They can access members of objects of other classes.
- So your class, we can call it as a container of objects of my class. So in container classes, if they want to access private members or protected members, then we can declare them as friends inside those classes.

Static Members :

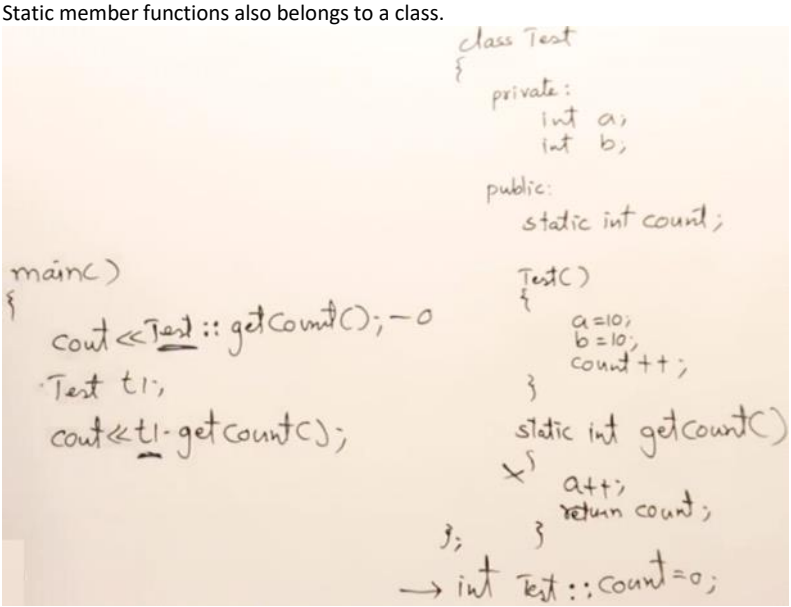
- If we create an object of a class, every object will have its own data members that are declared in the class. But we make a data member as static it will remain same across all the object.
- The static variable/member belong to class not to every object.
- The static member memory is allocated only one time and shared among all objects.
- So static variables or static data members of a class belongs to a class that doesn't belong to an object, and all the object s can change it. So there will be only one copy and every object will share that copy.
- When we have a static variable inside the class, we must declare it outside again using scope resolution.
    - Int className::count = 0;



- It is just like global variable. We are making it accessible only to the objects of this class. So that is static variable. S o only one time the memory is allocated and all objects can access it.
- The static data numbers can be accessed using object also, or they can also be accessed using class name. If they are public, we can directly access them using class name, they can be accessed using object name as well as they can be accessed upon the class. So they actuall y belong to a class.

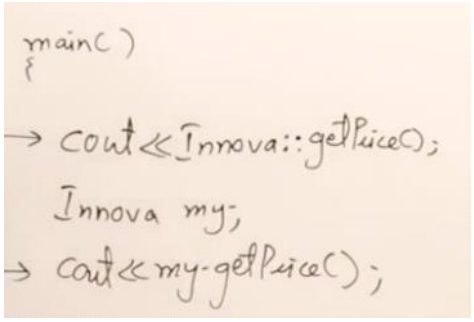
Static Member Functions :

- Static member functions can access only static data members of a class. They cannot access non static data numbers.
- Static member functions also belongs to a class.



Logic behind Static Members and Functions :

- If we went to a car showroom, we can know the price of car without buying the car.
- Which means we have a class name Innova, we have a static member/function for price, we don't need to create an object, we ca n access using class directly.



Few Points/Usage regarding Static Members :

- Static Members can be used as counter.
- Static members can be used as a shared memory for all the objects like one object of write something there and the other obje ct can read something from there.
- Static members can provide information about the class. For example, car price is information about the class.

Inner/Nested Classes :

- Inner class is writing a class inside another class so that it is useful only within that class.
- Inner class can access the members of outer class if they are static.
- Outer class can create the object of inner class.
- Using that object is can it access all the members of the class. The outer class can access all the members of the class.
- We can access only those members which are public. We cannot access private and protected members of the class.
- It's limited scope class that is visible only inside outer class.
- We can create objects of inner class outside the outer class by using score resolution (Possible only if declared as public)
  - Object creation in main :
    - Outer::Inner i;
    - i.Display(); // output : Display of Inner
  - We can also make it private, so it won't be accessible outside.

```
class Outer {
    public:
        void Display() {
            cout<<"Display of Outer\n";
        }
        void fun() {
            i.Display();
        }

        class Inner {
            public:
                void Display() {
                    cout<<"Display of Inner\n";
                }
        };

        Inner i;

};
```

- Friend Class can access private member of another class, upon its object.
- Static Functions are the functions of a class.
- Friend functions are not member functions of a class.
- Static member functions can be called using Class name and also using Object.
- Static members are used for providing information of class.