* Hatırlatma / Özet:

→ static data members class'ın elemanları fakat instance elemanı değil!!

→ static var elemanları sınıf başına 1 adet. Nasıl statikler, class'ın kapladığı alanı artırır.

```
class Myclass {
public:
    static int x; //declaration
};


int main()
{
    Myclass::x = 50;
    Myclass m;

    m.x = 30;
}
```

→ statiklere böyle data ile

→ static data members'a nokta / ok operatörleriyle erişmemekte böyle var.

Doğru init

Şu şekilde

```
class Myclass {
public:
    static int x; //declaration
};

//myclass.cpp

int Myclass::x; → static olduğu için ilk değeri = 0
artık kod segmentine ayrıldı

int main()
{
    std::cout << Myclass::x << "\n";
}
```

* Static Data Members'ların Initializationları:

* Sınıf içi ilk değer vermek için:

→ C++ 17 öncesi:

- Static var elemanı const olmalı
- const olduktan sonra tam sayı olmalı (integral type) → int gibi / double olamazdı
- Tam sayı olduğu için ENUMLAR da kaldı

→ C++ 17 Sonrası:

- Inline olarak tanıtılmalı.
- Her type inline olarak tanıtılabilir.

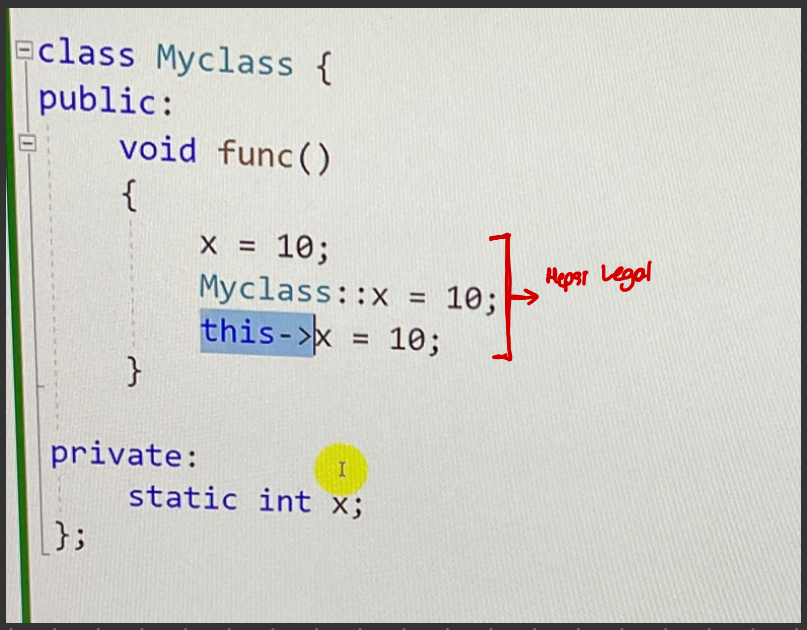
*→

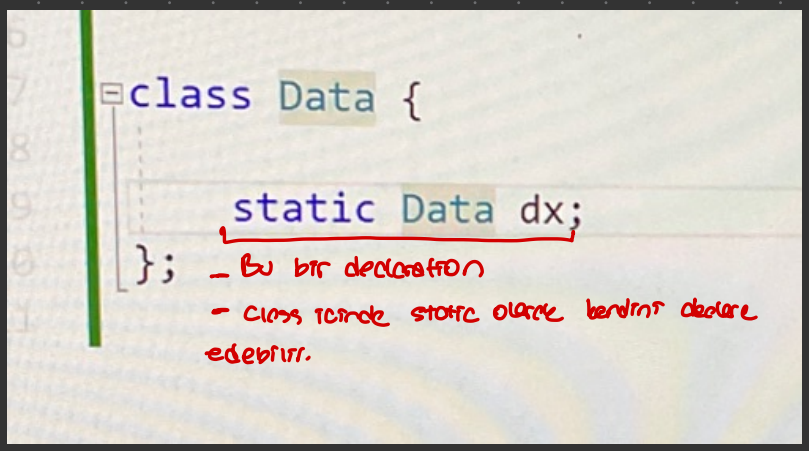
```
class Myclass {
public:
    Myclass() : x{ 10 } {
    }
private:
    static int x;
};
```

constructor initialization list'te static var elemanları alamaz.

↘ Syntax error.

* Static Data Members'ın Namespace'te Ammosi: → Privatelar için

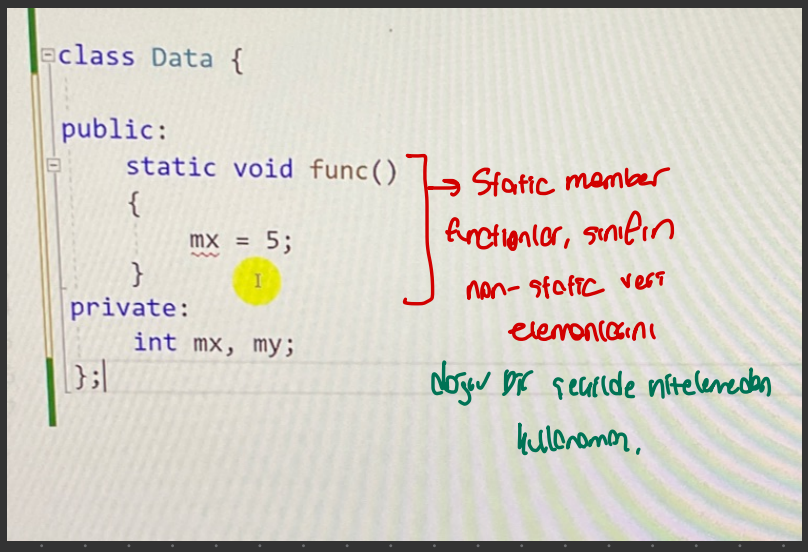


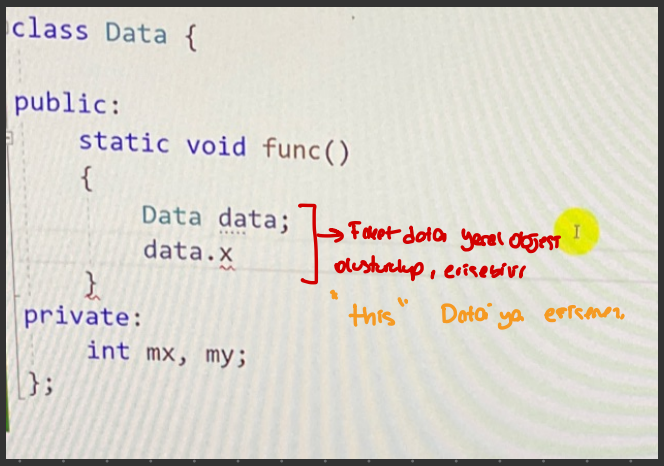


→ Complete type / Incomplete type'ların farkı!

* Static Member Functions:

→ Global fonksiyon gibi, this* 'a ihtiyacı yok.
→ Instance ile ilgili değil, Class ile ilgili değişken yapar.
→ Static mi değil mi sorusunun cevabı → instance'ı mı ilgilendiriyor
→ class'ı mı ilgilendiriyor.





→ Static member fonksiyonlar const olamaz.

→ Sınıfın static veri elemanına ilk değer veren yerin → ilk class scope'a bakar

```
class Myclass {
public:

    static int x;
    static int foo(int x, int y);
};

//myclass.cpp
int foo();

int Myclass::x = foo();
```

```
public:

    static int x;
    static int foo()
    {
        return 5;
    }
};

//myclass.cpp
int foo()
{
    return 1;
}

int Myclass::x = foo() + ::foo();

int main()
{
    std::cout << "Myclass::x = " << Myclass::x << "\n";

}
```

Burada class scope değil, Namespace scope'taki çağırılır.

* Named Constructor:

- Sınıfın constructor'ını doğrudan kullanmak yerine private yapıp constructor vazifesi gören ayrı bir constructor tanımlar.