

* Inline Functions: → inline expansiondan farkını bil.

→ Inline Expansion:

- Bir optimizasyon tekniği (compiler yapar)
- Fonksiyonun çağırıldığı yerin girisi çıkışlarını tutup, linker'a işletmek yerine direkt çağırıldığı yerde execute edilir.

- Bir fonksiyon inline olarak tanımlanıp, header'a yazılırsa, ODR kuralının ihlal edilmediğini garantiler.

- Eğer header'da tüm fonksiyonlar, inline tanımlanırsa, bu durumda oop abstraksiyon amaçlı model oluşturulur. Böyle modellerden oluşan kütüphanelere, header-only library denir.

- Inline fonksiyon tanımlamak, compiler'a inline expansion şansı verir. (Garantilemez)

- C++ 17 ile inline variables eklendi

↳ tanımlandığı header'ı elden tüm dosyalarda sadece 1 adet tanım olmasını garantiler.

- Global fonksiyonlarda olduğu gibi, member fonksiyonlar da inline olarak header'da tanımlanabilir. Bu durum ODR'i ihlal etmez.

```
class Murat {  
public:  
    void set(int x);  
private:  
    int mx;  
};  
  
inline void Murat::set(int x) {  
    mx = x;  
}
```

ODR violation yok.

- Direct Class Definition içerisinde de fonksiyon tanımlanabilir. Implicitly Inline olur.

```
class Murat {  
public:  
    void set(int x) {  
        mx = x;  
    }  
private:  
    int mx;  
};
```

Set implicitly inline function oldu

* Constructors: - Sınıf nesnesini **hayata getiren** üye fonksiyon / işlevini sağlayan da destructor

- Bir sınıfın constructor ve destructorları **Global veya static olmaz!!**
- Constructorun adı **class adı ile aynı**
- Constructorların **return türü yoktur.**
- Constructorlar **overload** edilebilir.

* Bir nesnenin, **organın olmadan** çağırılabilen constructor: → **Default ctor**

→ Ya organın olmayarak, ya da **default argument** olarak.

→ **Special Member Functions:**

- Default Constructor
- Destructor
- Copy Constructor
- Move Constructor
- Copy Assignment
- Move Assignment

⇒ Değerleri **birbirini** imitasyon bile, bu fonksiyonlar kendi **generate** edilebilir. / Bna default ab dnyar.

* Constructor ve Destructor **CONST OLAMAZ.**

- Constructor ve Destructor **PRIVATE OLABİLİR** ama **Client** kullanamaz. **Protected** olabilir
- Sınıf içinde **inline** tanımlanabilir.
- Constructor, **non-static** olduğu için, **this** pointer kullanılabılır.

* Destructors:

- Constructor gibi ama **başında ~ (tilde)** var.
- Global olmaz, static olmaz, const olmaz.
- Parametresi **OLAMAZ!!** Overload edilemez!

* Statik Ümitli Nesnelerde Constructor / Destructor.

Statik ümit → ya **global olarak** tanımlanarak.
→ ya da **static anıtle**

1. Global Olarak Tanıtılması:

```
};  
  
Nec nx;  
Erg ex;  
  
int main()  
{  
    std::cout << "main başladı\n";  
    std::cout << "main sona eriyor\n";  
}
```

Bu ilk class'ın hem ctor hem de dtor'uar

main fonksiyonun önce construct edilir
birmeden hemen önce destruct edilir.

Fotoğ

1. Nec ctor
2. Erg ctor
3. Erg dtor
4. Nec dtor

en son construct edilen, ilk destruct edilir.

* **Fotoğ**, farklı class'lar
hangisi constructor önce
çagırılır belli değil.

↓
Static Initialization
fıra seo

2. Statik Ürünle → Bir fonksiyonun statik yerel değişkeni

• Bu nesnenin construct edilmesi için tanıtlığı fonksiyonun çağırılması gerekir.

• Fakat statik üretilir olduğu için.

```
Microsoft Visual Studio Debug Console
main başladı
Nec default ctor this = 001653C4
foo fonksiyonuna yapılan 1. çağrı
foo fonksiyonuna yapılan 2. çağrı
foo fonksiyonuna yapılan 3. çağrı
foo() foo fonksiyonuna yapılan 4. çağrı
foo fonksiyonuna yapılan 5. çağrı
static Nec main sona eriyor
static Nec destructor this = 001653C4
std::cout << "D:\KURSLAR\MART2022\Release\MART2022.exe (process 3540) exited with code 0.\n";
Press any key to close this window . . .

main()

std::cout << "main() exited with code 0.\n";
foo();
foo();
foo();
```

Fonksiyon kaç kez çağırılırsa, çağırılır, yalnızca 1 kez maine girilince destructor çağırılır.

* Otomatik Üretilen Sınıf Nesneleri: → Bir fonksiyon içinde tanımlanan nesneler otomatik üretilir.

• Fonksiyon kaç kez çağırılırsa, bu sınıf nesneleri 0 kadar construct ve destruct edilir.

• Array elemanı class olabilir.

→ tüm array elemanları için tek tek constructor / destructor çağırılır.