

**MALATYA
TURGUT ÖZAL ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ
ALGORİTMA VE PROGRAMLAMA 2**

SINIFLAR VE NESNELER-3

YIKICI FONKSİYONLAR

- Nesneler yok edileceği zaman çalıştırılan fonksiyondur.
- Yapıcı fonksiyonlarda olduğu gibi sınıf ismi ile aynı isimde olmak zorundadır.
- Fakat ismin başında Tilda işareti ~ olması gereklidir.
- Değer döndürmezler
- Parametre almazlar
- Bir sınıf içerisinde sadece 1 adet yıkıcı fonksiyon olabilir.
- Yapıcı fonksiyonda olduğu gibi tanımlanmazsa derleyici tarafından otomatik olarak tanımlanır.

YIKICI FONKSİYONLAR Örnek-1

```
1  #include <iostream>
2  using namespace std;
3
4  class test{
5      public:
6          int a;
7          test(){
8              cout<<"Yapici fonksiyon calisti.."<<endl,
9              a=15;
10         }
11
12         ~test(){
13             cout<<"Gorev bitti Yikici Fonksiyon Calisiyor.."<<endl;
14         }
15
16         void goruntule(){
17             cout<<"a degeri="<<a<<endl;
18         }
19     }x;
20
21     int main() {
22         x.goruntule();
23
24         test y;
25         y.goruntule();
26
27         return 0;
28     }
```

C:\Users\acer\Desktop\ders slaytlar\algoritma2\ders slay

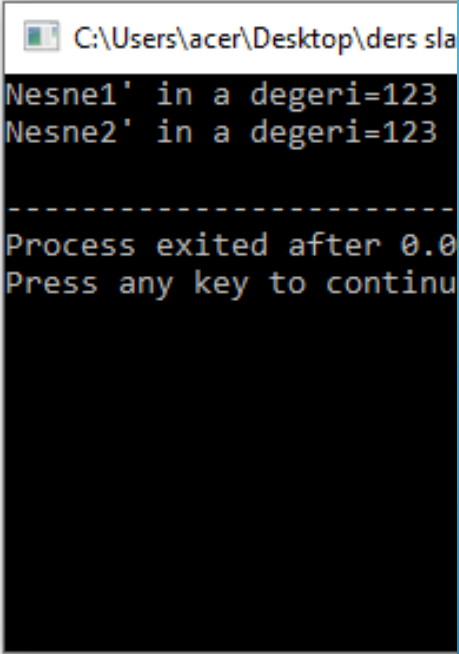
```
Yapici fonksiyon calisti..
a degeri=15
Yapici fonksiyon calisti..
a degeri=15
Gorev bitti Yikici Fonksiyon Calisiyor..
Gorev bitti Yikici Fonksiyon Calisiyor..

-----
Process exited after 0.04902 seconds with
Press any key to continue . . .
```

NESNEDEN NESNEYE ATAMA YAPMAK

- İstediğimiz zaman, oluşturduğumuz sınıf nesnelerini birbirlerine atayabiliriz. Örnek-2

```
1  #include <iostream>
2  using namespace std;
3
4  class deneme_sinif{
5      int a;
6
7      public:
8          void atamayap(int);
9          int goruntule();
10 };
11
12 void deneme_sinif::atamayap(int sayi){
13     a=sayi;
14 }
15
16 int deneme_sinif::goruntule(){
17     return a;
18 }
19
20 int main() {
21     deneme_sinif nesne1, nesne2;
22     nesne1.atamayap(123);
23     nesne2=nesne1;
24
25     cout<<"Nesne1' in a degeri="<<nesne1.goruntule()<<endl;
26     cout<<"Nesne2' in a degeri="<<nesne2.goruntule()<<endl;
27
28     return 0;
29 }
```



The screenshot shows a Windows command prompt window titled "C:\Users\acer\Desktop\ders sla". The output of the program is as follows:

```
Nesne1' in a degeri=123
Nesne2' in a degeri=123

-----
Process exited after 0.0
Press any key to continue
```

INLINE FONKSİYONLAR

- Programlarımızı yazarken fonksiyon oluşturmak yararlıdır.
- Ancak fonksiyonlar ne kadar çok olursa, fonksiyon çağrılarında okadar çok olacaktır.
- İşte bu durumda satır içi fonksiyonlar anlamına gelen inline fonksiyonlar devreye giriyor ve fonksiyon çağrılarının ek olarak getirdiği yükü azaltıyor.
- Tek dezavantajı çalışma zamanını azaltırken programın boyutunu arttırmasıdır.

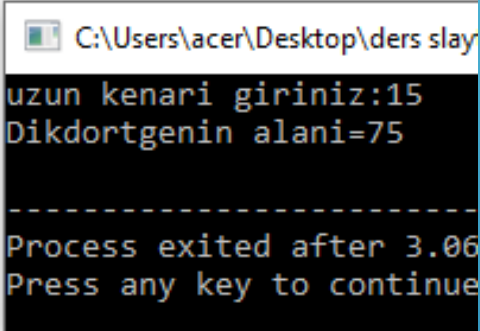
INLINE FONKSİYONLAR

- Normal fonksiyonlar ramde ayrı bir bölgede farklı bir programmış gibi tutulur ve çağırıldıkları zaman ramın o bölgesinden çekilirler. Bu bir miktar performans azalmasına sebep olabilir.
- inline fonksiyonlar ise çağırıldıkları her yere kopyalanırlar. Böylece performans kaybı yaşatmazlar. Ama çok büyük fonksiyonları inline olarak tanımlamak sıkıntılı olabilir.
- Derleyiciler genellikle hangi fonksiyonun inline olması gerektiğine karar verebilirler. Bu yüzden inline anahtar kelimesini kullanmak neredeyse gereksizdir.

INLINE FONKSİYONLAR. ÖRNEK-3

Fonksiyonu oluştururken başına inline kelimesini getiriyoruz. Geri kalan kısımlar normal fonksiyon ile aynı biçimde işlemekte.

```
1  #include <iostream>
2  using namespace std;
3
4  inline int dikdortgen(int sayi1,int sayi2){
5      int alan=sayi1*sayi2;
6      return alan;
7  }
8
9
10 int main() {
11     int uzun,kisa;
12
13     cout<<"kisa kenari giriniz:";
14     cin>>kisa;
15     cout<<"uzun kenari giriniz:";
16     cin>>uzun;
17
18     cout<<"Dikdortgenin alani="<<dikdortgen(kisa,uzun)<<endl;
19
20     return 0;
21 }
```

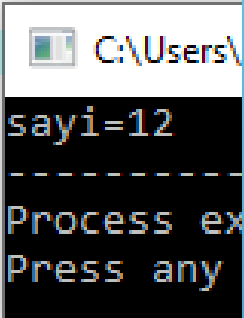


ARKADAŞ FONKSİYONLAR

- Arkadaş fonksiyonlar sınıfların dışında global olarak tanımlanmış bir fonksiyondur ve bu fonksiyon sınıf içerisindeki private verilere erişebilmeyi sağlar.
- Bildirimi, arkadaş olacağı sınıfın içinde public veya private kısmında herhangi bir yerde yapılabilir.
- Örneğin bir X sınıfımız olsun. Bu sınıf dışında tanımlanmış bir Y fonksiyonunu ise arkadaş olarak seçsin.
- X sınıfının içinde friend fonksiyonadı; şeklilde bir ifade ile tanımlanır.

ARKADAŞ FONKSİYONLAR-ÖRNEK 4

```
1  #include <iostream>
2  using namespace std;
3
4  class deneme{
5      int sayi;
6
7      public:
8          deneme(){
9              sayi=12;
10         }
11
12         friend int sayi_fonk(deneme x);
13     };
14
15     int sayi_fonk(deneme x){
16         return x.sayi;
17     }
18
19
20     int main() {
21         cout<<"sayi="<<sayi_fonk(a);
22         return 0;
23     }
```

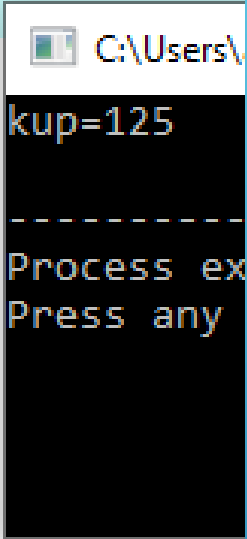


- 1- ilk olarak deneme adında bir sınıf oluşturduk.
- 2- Sınıfın dışarısında sayi_fonk() adında bir fonksiyon oluşturduk.
- 3- Bu fonksiyonu sınıf içerisinde friend int sayi_fonk() şeklinde arkadaş olarak belirttik.
- 4- Yapıcı fonksiyon ile private kısımda tanımladığımız sayi değişkenine 12 değerini atadık. Fonksiyonda ise 12 sayısını return ettik.

Buradaki önemli kısım sayi_fonk() fonksiyonumuzun içinde argüman olarak deneme x girdik. Böylece sınıf adımızı kullanarak deneme sınıfı içinden x adında bir örnekleme almış olduk. Daha sonra main() içinde sayi_fonk(a) ile a nesnemizi argüman olarak belirterek private sayi değişkenine eriştik ve ekranda 12 yazdırdık.

ARKADAŞ FONKSİYONLAR-ÖRNEK 5

```
1  #include <iostream>
2  using namespace std;
3
4  class kup{
5      int sayi;
6
7      public:
8          kup(){
9              sayi=5;
10         }
11
12         friend int kup_al();
13     };
14
15     int kup_al(){
16         return k.sayi*k.sayi*k.sayi;
17     }
18
19
20     int main() {
21         cout<<"kup="<<kup_al()<<endl;
22         return 0;
23     }
```



Bu örnek, bir önceki örneğe çok benzemektedir.

Private olarak bir sayi değişkeni tanımladık. Kup_al() fonksiyonunu arkadaş olarak bildirdik.

Fonksiyonda sayının küpünü almak için

return k.sayi*k.sayi*k.sayi;

İfadesini kullandık.

Yapıcı fonksiyonda private değişkene 5 değerini atamıştık.

Main() içinde bunu yazdırınca görüldüğü gibi ekrana 5'in küpü olan 125 çıktısını görmüş olduk.

SINIFTAN SINIFA ARKADAŞLIK

- Sadece sınıf ile fonksiyon arasında değil
- Sınıf ile sınıf arasındada arkadaşlık ilişkisi kurulabilir.
- Bir sınıfa ait bir fonksiyonu başka bir sınıf arkadaş olarak kabul edebilir.
- Tanımlama şu şekilde gerçekleştirilir.
- **friend** fonksiyon_tipi **sinifin_adi::fonksiyon_adi**(varsa parametreler)

SINIFTAN SINIFA ARKADAŞLIK-ÖRNEK-6

```
1  #include <iostream>
2  using namespace std;
3
4  class kamyon;
5
6  class araba{
7      int kisi_sayisi,hiz;
8
9      public:
10         araba(int y,int h){
11             kisi_sayisi=y;
12             hiz=h;
13         }
14
15         int arkadaslik(kamyon k);
16     };
17
18     class kamyon{
19         int tonaj,hiz;
20
21         public:
22             kamyon(int a,int b){
23                 tonaj=a;
24                 hiz=b;
25             }
26
27             friend int araba::arkadaslik(kamyon k);
28     };
29
30     int araba::arkadaslik(kamyon k){
31         return hiz-k.hiz;
32     }
33
34     int main() {
```

```
34     int m;
35     araba araba1(2,225);
36     araba araba2(4,85);
37
38     kamyon kamyon3(30000,75);
39     kamyon kamyon4(32000,85);
40
41     cout<<"araba1 ve kamyon3 u karsilastiralim:"<<endl;
42     m=araba1.arkadaslik(kamyon3);
43
44     if(m<0)
45         cout<<"kamyon daha hizli";
46     else if(m==0)
47         cout<<"Araba ile kamyonun hizi esit";
48     else
49         cout<<"Araba daha hizli";
50
51     cout<<endl<<endl;
52     cout<<"araba2 ve kamyon4 u karsilastiralim:"<<endl;
53     m=araba2.arkadaslik(kamyon4);
54
55     if(m<0)
56         cout<<"kamyon daha hizli";
57     else if(m==0)
58         cout<<"Araba ile kamyonun hizi esit";
59     else
60         cout<<"Araba daha hizli";
61
62     return 0;
63 }
64 }
```

araba1 ve kamyon3 u karsilastiralim:
Araba daha hizli

araba2 ve kamyon4 u karsilastiralim:
Araba ile kamyonun hizi esit

SINIFTAN SINIFA ARKADAŞLIK-ÖRNEK-6 (Açıklaması)

Bu örneğimiz ilk bakışta biraz karmaşık gelebilir. Bu nedenle programda yaptıklarımızı adım adım anlatalım.

1. Programımız araba ve kamyon bilgilerini alacak. Bunları 2 ayrı sınıfta tutacak. Daha sonra sınıftan sınıfa arkadaşlık kurulacak. Araba ve kamyon sınıflarından nesneler oluşturulup hızları karşılaştırılacak.
2. İlk olarak en üstte kamyon sınıfının protatip tanımını yaptık.
3. Daha sonra araba sınıfını tanımladık private olarak kişi_sayisi ve hiz değişkenlerini tanımladık. Yapıcı fonksiyonunu oluşturduk ve argümanları private verilere atadık.
4. Yine araba sınıfına ait olan arkadaslik adındaki fonksiyonun protatipini belirttik.
5. Daha sonra en üstte protatipini belirttiğimiz kamyon sınıfını oluşturduk, private olarak tonaj ve hiz adında 2 değişken tanımladık. Yine bu defa kamyon sınıfının Yapıcı fonksiyonunu oluşturduk ve argümanları private verilere atadık
6. Daha sonra araba sınıfının arkadaşlık adındaki fonksiyonunu kamyon sınıfı içinde arkadaş olarak kabul etmesi için gerekli tanımı yaptık.

SINIFTAN SINIFA ARKADAŞLIK-ÖRNEK-6 (Açıklaması)

7. Kamyon sınıfının yazımı bittikten sonra araba sınıfının arkadaslik adındaki fonksiyonun yazdık ve fonksiyon içinde arabanın hızından kamyonun hızını çıkartıp return ile değeri döndürdük.
8. Main() içinde araba sınıfından araba1,araba2 ve kamyon sınıfında kamyon3,kamyon4 adlı nesneleri oluşturduk ve içine değerlerini girdik.
9. Daha sonra m değişkeni ile araba sinifinin araba1 nesnesinin arkadaslik adındaki fonksiyonu çağırdık ve fonksiyona kamyon sınıfının kamyon3 nesnesini parametre olarak gönderdik. Böylece araba1 ile kamyon3 ün hızını karşılaştırmak için gerekli olan m değerini elde ettik.
10. Daha sonra gelen m değerlerine göre if-else if-else kullanarak hangisinin daha hızlı olduğunu ekrana yazdırdık.
11. 9 ve 10. maddedeki gibi bukez araba2 ile kamyon4 dü karşılaştırdık.
12. Araba sınıfındaki kişi_sayisi ve kamyon sınıfındaki tonaj değişkenlerini programı biraz süslemek için kullandık. Bu değişkenler ile bir işlem yapmadık ☺