

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

SINIFLAR VE NESNELER-devam

YAPICI FONKSİYONLAR

- Oluşturduğumuz sınıflar içerisinde yer alan her hangi bir değişkene başlangıç değeri verebilmek için yapıcı fonksiyonlar kullanılır.
- Şuana kadarki hiçbir örnekte yapıcı fonk. Kullanmadığımız için değişkenlere başlangıç değeri veremedik.
- Yapıcı fonksiyonun ismi sınıf ile aynı olmak zorundadır.
- Bir nesne oluşturduğumuz zaman o nesnenin yapıcı fonksiyonu otomatik olarak çağrılmaktadır.
- Değer döndürmezler bu yüzden return gibi bir geri dönüş değerleri bulunmamaktadır.

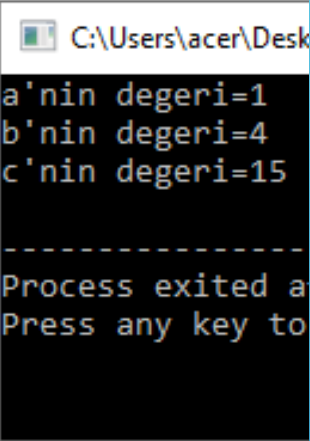
YAPICI FONKSİYONLAR- devam

- Birden fazla yapıcı fonksiyon tanımlanabilmektedir. Fakat hepsinin ismide sınıf ismi ile aynı olmak zorundadır. Ancak Fonksiyonun alacağı parametre değerleri değiştirilmelidir. Aynı isim aynı parametrelili fonksiyonlar kullanılırsa hata verecektir.
- Nesne oluşturulacağı zaman sadece bir kere çalıştırılırlar.
- Parametre alabilirler.
- Şu ana kadarki örneklerde olduğu gibi eğer yapıcı fonksiyon tanımlamazsak, derleyici tarafından arka planda otomatik olarak oluşturulmaktadır.

Yapıcı fonksiyonların daha iyi anlaşılabilmesi için birkaç örnek yapalım.

ÖRNEK 2

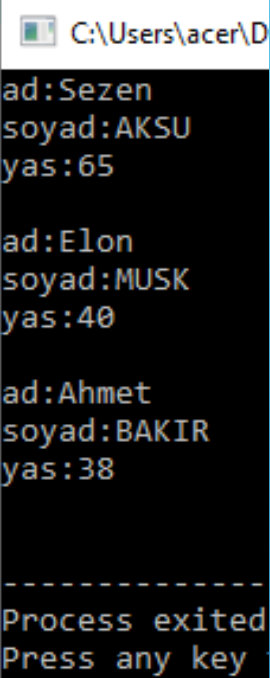
```
1  #include <iostream>
2  using namespace std;
3
4  class sayilar{
5      public:
6          int a,b,c;
7
8          sayilar(){
9              a=1;
10             b=4;
11             c=15;
12         }
13     }sayi;
14
15     int main() {
16
17         cout<<"a'nin degeri="<<sayi.a<<endl;
18         cout<<"b'nin degeri="<<sayi.b<<endl;
19         cout<<"c'nin degeri="<<sayi.c<<endl;
20
21         return 0;
22     }
```



Normalde değişkenlere sınıf içerisinde başlangıç değere veremeyeceğimizi söylemiştik. Ancak **yapıcı fonksiyonlar** sayesinde değişkenlere başlangıç değeri verebildik. Daha sonra main() fonksiyonunda nesnemiz aracılığıyla bu değişkenlere 4 ulaştık ve ekrana yazdırdık.

ÖRNEK 3: Yapıcı fonksiyona parametre geçirme

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class insanlar{
6      string ad, soyad;
7      int yas;
8
9      public:
10
11      insanlar(string a,string b,int c){
12          ad=a;
13          soyad=b;
14          yas=c;
15      }
16
17      void goster(){
18          cout<<"ad:"<<ad<<endl;
19          cout<<"soyad:"<<soyad<<endl;
20          cout<<"yas:"<<yas<<endl<<endl;
21      }
22  };
23
24  int main() {
25
26      insanlar insan1("Sezen","AKSU",65);
27      insanlar insan2("Elon","MUSK",40);
28      insanlar insan3("Ahmet","BAKIR",38);
29
30      insan1.goster();
31      insan2.goster();
32      insan3.goster();
33
34      return 0;
35  }
```



- Bu örneğimizde yapıcı fonksiyona parametre geçirdik. Yapıcı fonksiyondaki argümanları veri tiplerine uygun bir şekilde **private** değişkenlere atadık.
- Daha sonra **main()** içinde sınıf adı nesne adı (**insanlar insan1**) şeklinde bildirimi yaptık ve yine bu kısımda **parantez içinde** ad, soyad, yaş değerlerini girdik.
- Son olarak oluşturduğumuz nesnelerin **goster()** fonksiyonlarını çağırarak ekran çıktılarını elde ettik.

ÖRNEK 3-a: Bir önceki örnekte yapıcı fonksiyona parametre geçirdik. Fakat yapıcı fonksiyonda değişkenlere başlangıç değeri vermek isteseydik ne yapacaktık. O zaman sınıfla aynı isimde bir yapıcı fonksiyon daha oluşturacaktık ve parametre almayacaktı.

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class insanlar{
6      string ad, soyad;
7      int yas;
8
9      public:
10
11     → insanlar(){
12         ad="Ali";
13         soyad="VELI";
14         yas=49;
15     }
16
17     → insanlar(string a,string b,int c){
18         ad=a;
19         soyad=b;
20         yas=c;
21     }
22
23     void goster(){
24         cout<<"ad:"<<ad<<endl;
25         cout<<"soyad:"<<soyad<<endl;
26         cout<<"yas:"<<yas<<endl<<endl;
27     }
28 };
29
```

```
int main() {

    insanlar insan1("Sezen","AKSU",65);
    insanlar insan2("Elon","MUSK",40);
    insanlar insan3("Ahmet","BAKIR",38);

    insan1.goster();
    insan2.goster();
    insan3.goster();

    insanlar insan4;
    insan4.goster();

    return 0;
}
```

```
ad:Sezen
soyad:AKSU
yas:65

ad:Elon
soyad:MUSK
yas:40

ad:Ahmet
soyad:BAKIR
yas:38

ad:Ali
soyad:VELI
yas:49
```

ÖRNEK 3-b: Bir önceki örneği biraz daha karmaşıktıralım. Üçüncü yapıcı fonksiyonuda ekleyelim. Bu fonksiyonda sadece ismi kullanıcıdan klavyeden alsın, soyad ve yas degerlerini de sabit atasın.

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class insanlar{
6      string ad, soyad;
7      int yas;
8
9      public:
10
11     insanlar(){
12         ad="Ali";
13         soyad="VELI";
14         yas=49;
15     }
16
17     insanlar(string a,string b,int c){
18         ad=a;
19         soyad=b;
20         yas=c;
21     }
22
23     insanlar(string a){
24         ad=a;
25         soyad="PEKER";
26         yas=60;
27     }
28
29     void goster(){
30         cout<<"ad:"<<ad<<endl;
31         cout<<"soyad:"<<soyad<<endl;
32         cout<<"yas:"<<yas<<endl<<endl;
33     }
34 };
```

```
int main() {

    insanlar insan1("Sezen","AKSU",65);
    insanlar insan2("Elon","MUSK",40);
    insanlar insan3("Ahmet","BAKIR",38);

    insan1.goster();
    insan2.goster();
    insan3.goster();

    insanlar insan4;
    insan4.goster();

    string ad;
    cout<<"ad giriniz:";
    cin>>ad;
    insanlar insan5(ad);
    insan5.goster();

    cout<<"Baska bir ad daha giriniz:";
    cin>>ad;
    insanlar insan6(ad);
    insan6.goster();

    return 0;
}
```

```
ad:Sezen
soyad:AKSU
yas:65

ad:Elon
soyad:MUSK
yas:40

ad:Ahmet
soyad:BAKIR
yas:38

ad:Ali
soyad:VELI
yas:49

ad giriniz:Mustafa
ad:Mustafa
soyad:PEKER
yas:60

Baska bir ad daha giriniz:Hakan
ad:Hakan
soyad:PEKER
yas:60
```


SINIFIN FONKSİYONLARINI SINIF DIŞINDA YAZMA

- Bir sınıfın fonksiyonları sınıf dışındada yazılabilir.
- Bunu yaparken öncelikle sınıfın içinde sadece o fonksiyonların ismini ve varsa parametrelerini yazıyoruz. Yani protatip tanımını yapıyoruz.
- Daha sonra sınıfın dışarısında fonksiyonlarımızın gövdesini oluşturuyoruz.
- Fonksiyonun ismi **sınıf_adı::fonksiyon_adı** şeklinde sınıf adı daha sonra :: (iki tane üstüste iki nokta) ve ardından fonksiyon adı şeklinde.
- Örneğin önceki örneklerdeki fonksiyonları sınıf dışında yazsaydık.

```
insanlar::insanlar(string a, string b,string c){  
    Komutlar ....  
}
```

```
void insanlar::goster(){  
    Komutlar ...  
}
```


ÖRNEK-4

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class arabalar{
6      string marka, model;
7      int yil;
8
9      public:
10
11          arabalar();
12          arabalar(string x,string y,int z);
13          void yazdir();
14  };
15
16  arabalar::arabalar(){
17      marka="audi";
18      model="a3";
19      yil=2021;
20  }
21
22  arabalar::arabalar(string x,string y, int z){
23      marka=x;
24      model=y;
25      yil=z;
26  }
27
28  void arabalar::yazdir(){
29      cout<<"Marka:"<<marka<<endl;
30      cout<<"Model:"<<model<<endl;
31      cout<<"Yil:"<<yil<<endl<<endl;
32  }
```

```
33
34  int main() {
35
36      arabalar araba1("Opel", "Astra", 2015);
37      arabalar araba2("Ford", "Focus", 2017);
38      arabalar araba3;
39
40      araba1.yazdir();
41      araba2.yazdir();
42      araba3.yazdir();
43
44
45      return 0;
46  }
```

```
Marka:Opel
Model:Astra
Yil:2015
```

```
Marka:Ford
Model:Focus
Yil:2017
```

```
Marka:audi
Model:a3
Yil:2021
```