

Nesne Tabanlı Programlama

Bölüm 4

Operatörlere Yeni İşlevler Yüklenmesi, Getter Setter

Dr. Öğr. Üyesi Murat TAŞYÜREK (kayubmprogramlama1@gmail.com)

23 Ekim 2023

Kayseri Üniversitesi, Bilgisayar Mühendisliği Bölümü

Operatörlere Yeni İşlemlerin Yüklenmesi

- Nesne tabanlı programlama sayesinde var olan operatörlere (+,-,*,!) fonksiyon yazarak bu operatörlerin sizin belirlediğiniz işlemleri yapmasını sağlayabilirsiniz.
- Operatör fonksiyonları bir sınıfın üyesi de olabilir. Böylece o sınıftan üretilen nesneler üzerinde işlem yapan operatörler tanımlanmış olur.
- Operatör kullanımı fonksiyonun çağırılması (tetiklenmesi) anlamına gelir.
- Operatörlere yeni işlemler yükleyerek yapılabilecek her şey normal fonksiyonlar ile de yapılabilir.
- Bu sebeplerden dolayı operatöre işlem yüklemek ancak program daha avantajlı (anlaşılır, okunabilir vb.) duruma geliyorsa yapılmalıdır.

Operatörlere Yeni İşlemlerin Yüklenmesi

- Operand işlemleri genellikle integer, double gibi sayısal değerler için kullanılır.
- Ancak nesneler için de bu işlemlere ihtiyaç duyulabilir.
- Operand fonksiyonları statik olarak tanımlanır.
- Fonksiyon nesneden sonra . konmadan çağrılacağından dolayı bu fonksiyonun statik olarak tanımlanması gerekiyor.
- C# dilinde operatörlere yeni görevler yüklenmesi için "operator" kelimesi kullanılır.

Örnek 1

- Operatör ile geriye string, integers veya obje türünde geriye değer döndürülür.
- Calculater isminde iki adet sayı değeri tutan ve bu değerlere başlangıçta değer atamasına izin veren bir class oluşturalım.
- Clasımızın print isminde bir fonksiyonu olsun ve yazmış olduğumuz değerleri ekrana göndersin.
- Clasımızıda - operatörü ile birinci sayıdan ikinci sayıyı çıkarsın ve geriye aradaki farkı göndersin.
- Kodlayalım

Console App Calclateer Class

```
class Calculator
{
    public int number1, number2;
    1 reference
    public Calculator(int num1, int num2)
    {
        number1 = num1;
        number2 = num2;
    }

    // -operatörün görevi
    1 reference
    public static int operator -(Calculator c1)
    {
        int sonuc= c1.number1-c1.number2;
        return sonuc;
    }

    // print procedürü
    1 reference
    public void Print()
    {
        Console.WriteLine("Number1 = " + number1);
        Console.WriteLine("Number2 = " + number2);
    }
}
```




- operand
tanımlandı.

Kaynak Kodu

NNDers4Ornek0 NNDers4Ornek0.Calculator ope

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace NNDers4Ornek0
8 {
9     0 references
10     internal class Program
11     {
12         0 references
13         static void Main(string[] args)
14         {
15             Calculator clc = new Calculator(10, 25);
16             clc.Print();
17             int deger = -clc;
18             Console.WriteLine("iki sayı arasındaki fark:" + deger);
19         }
20     }
21     4 references
22     class Calculator...
```

 - operandı tetiklendi.

The screenshot shows a C# program in Visual Studio. The file is named `Program.cs` and is part of a project named `NNDers4Ornek0`. The code defines a namespace `NNDers4Ornek0` containing an internal class `Program`. The `Main` method creates a `Calculator` object, prints it, and calculates the difference between 10 and 25. The output window shows the execution results.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace NNDers4Ornek0
8 {
9     internal class Program
10     {
11         static void Main(string[] args)
12         {
13             Calculator clc = new Calculator(10, 25);
14             clc.Print();
15             int deger = -clc;
16             Console.WriteLine("iki sayı arasındaki fark:" + deger);
17         }
18     }
19     class Calculator
20     {
21         Number1 = 10
22         Number2 = 25
23         iki sayı arasındaki fark:-15
24         Press any key to continue . . .
25     }
26 }
```

Output:

```
C:\WINDOWS\system32\cmd.exe
Number1 = 10
Number2 = 25
iki sayı arasındaki fark:-15
Press any key to continue . . .
```

Örnek 2

- Operand ile yapılan işlemlerde geriye nesne de döndürülebilir.
- Elimizde iki tane kapsayıcı kutu olduğunu farzedelim.
- Bunlarında yükseklik, genişlik ve derinlik özelliklerinin olduğu düşünelim bu özellikler başlangıçta boş olsun ama sonradan sadece özel fonksiyonlarla set edilebilsin.
- İki bounding box toplandığında geriye dönen bounding box'ın özellikleri de ikisinin bütün özelliklerinin toplamı olsun.
- Kodlayalım

Console App Box Class

```
11 references
class Box
{
    private double length; // genişlik, uzunluk
    private double depth; // derinlik
    private double height; // yükseklik

    3 references
    public double getVolume()
    {
        return length * depth * height;
    }

    2 references
    public void setLength(double len)
    {
        length = len;
    }

    2 references
    public void setBreadth(double dept)
    {
        depth = dept;
    }

    2 references
    public void setHeight(double hei)
    {
        height = hei;
    }

    // Overload + operator
    1 reference
    public static Box operator +(Box b, Box c)
    {
        Box box = new Box();
        box.length = b.length + c.length;
        box.depth = b.depth + c.depth;
        box.height = b.height + c.height;
        return box;
    }
}
```

+ operatörü
tanımlama, +
operatörünü
yeni özellik
kazandırma

```
namespace NNDers4Ornek1_x
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Box Box1 = new Box();
            Box Box2 = new Box();
            Box Box3 = new Box();
            double volume = 0.0;

            // box 1 değerler
            Box1.setLength(6.0);
            Box1.setBreadth(7.0);
            Box1.setHeight(5.0);

            // box 2 değerler
            Box2.setLength(12.0);
            Box2.setBreadth(13.0);
            Box2.setHeight(10.0);


            // volume of box 1
            volume = Box1.getVolume();
            Console.WriteLine("Volume of Box1 : {0}", volume);

            // volume of box 2
            volume = Box2.getVolume();
            Console.WriteLine("Volume of Box2 : {0}", volume);

            // Add two object as follows:
            Box3 = Box1 + Box2;

            // volume of box 3
            volume = Box3.getVolume();
            Console.WriteLine("Volume of Box3 : {0}", volume);
        }
    }
}

11 references
class Box...
```



tanımladığımız
operatörün
çağırılması

```
17 // box 1 değerler
18 Box1.setLength(6.0);
19 Box1.setBreadth(7.0);
20 Box1.setHeight(5.0);
21
22 // box 2 değerler
23 Box2.setLength(12.0);
24 Box2.setBreadth(13.0);
25 Box2.setHeight(10.0);
26
27 // volume of box 1
28 volume = Box1.getVolume();
29 Console.WriteLine("Volume of Box1 : {0}", volume);
30
31 // volume of box 2
32 volume = Box2.getVolume();
33 Console.WriteLine("Volume of Box2 : {0}", volume);
34
```

C:\WINDOWS\system32\cmd.exe

```
Volume of Box1 : 210
Volume of Box2 : 1560
Volume of Box3 : 5400
Press any key to continue . . .
```

Hatalı kullanım

```
7 namespace NNDers4Ornek1_x
8 {
9     internal class Program
10     {
11         static void Main(string[] args)
12         {
13             Box Box1 = new Box();
14             Box Box2 = new Box();
15             Box Box3 = new Box();
16             double volume = 0.0;
17
18             // box 1 değerler
19             Box1.setLength(6.0);
20             Box1.setBreadth(7.0);
21             Box1.setHeight(5.0);
22
23             // box 2 değerler
24             Box2.setLength(12.0);
25             Box2.setBreadth(13.0);
26             Box2.setHeight(10.0);
27
28             // volume of box 1
29             volume = Box1.getVolume();
30             Console.WriteLine("Volume of Box1 : {0}", volume);
31
32             // volume of box 2
33             volume = Box2.getVolume();
34             Console.WriteLine("Volume of Box2 : {0}", volume);
35
36             // Add two object as follows:
37             Box3 = Box1 + Box2;
38
39             var Box4 = Box1 - Box2;
40
41             // volume of box 3
42             volume = Box3.getVolume();
43             Console.WriteLine("Volume of Box3 : {0}", volume);
44         }
45     }
46     class Box
47     {
48     }
49 }
```

Bu kullanımı deneyin - operand tanımlanmadığından hata verecektir.

Overload edilebilen ve edilemeyen operators

Overloadable and Non-Overloadable Operators

The following table describes the overload ability of the operators in C# -

Sr.No.	Operators & Description
1	+, -, !, ~, ++, -- These unary operators take one operand and can be overloaded.
2	+, -, *, /, % These binary operators take one operand and can be overloaded.
3	==, !=, <, >, <=, >= The comparison operators can be overloaded.
4	&&, The conditional logical operators cannot be overloaded directly.
5	+=, -=, *=, /=, %= The assignment operators cannot be overloaded.
6	=, ., ?:, ->, new, is, sizeof, typeof These operators cannot be overloaded.

C# Getter Setter

- The get method returns the value of the variable name . The set method assigns a value to the name variable. The value keyword represents the value we assign to the property.
- C# programlama dilinde sınıftan türetilen bir nesnenin özelliğini okumak için get, özelliğe değer atamak için de set metodu kullanılır.
- Bu metotlarda getter setter olarak ifade edilir.
- Öğrenci sınıfı ve sadece adı öz niteliğinin olduğu ve bunun için okuma ve yazma metotlarının olduğu örneği inceleyim.

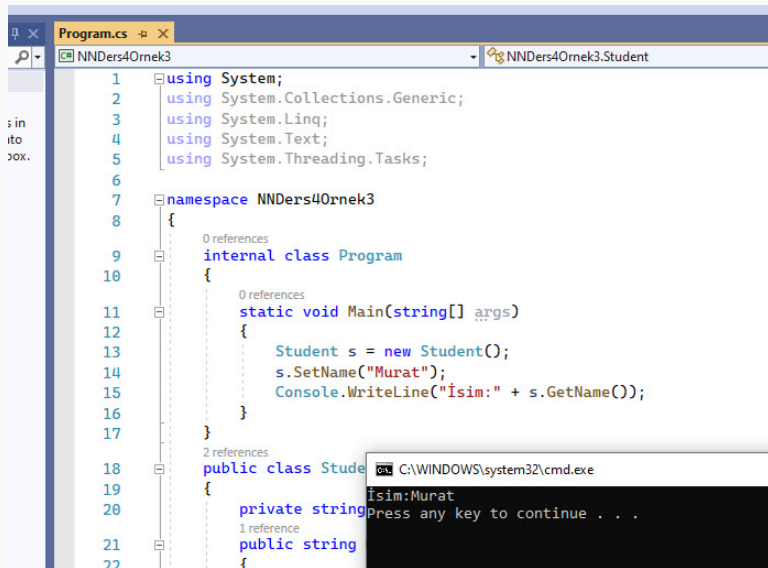
Kaynak Kodu

```
0 references
internal class Program
{
    0 references
    static void Main(string[] args)
    {
        Student s = new Student();
        s.SetName("Murat");
        Console.WriteLine("İsim:" + s.GetName());
    }
}

2 references
public class Student
{
    private string _name;

    1 reference
    public string GetName() → okuma için
    {
        return this._name;
    }

    1 reference
    public void SetName(string name) → değer atama için
    {
        this._name = name;
    }
}
```



The screenshot shows a Visual Studio IDE with a C# file named `Program.cs` open. The file is part of a project named `NNDers4Ornek3`. The code defines a namespace `NNDers4Ornek3` containing an internal class `Program` and a public class `Student`. The `Program` class has a `Main` method that creates a `Student` object, sets its name to "Murat", and prints it to the console. The `Student` class has a private `isim` property and a public `isim` property. The output window shows the result of the program execution: `İsim:Murat` followed by a prompt to press any key to continue.

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace NNDers4Ornek3
8 {
9     0 references
10     internal class Program
11     {
12         0 references
13         static void Main(string[] args)
14         {
15             Student s = new Student();
16             s.SetName("Murat");
17             Console.WriteLine("İsim:" + s.GetName());
18         }
19     }
20     2 references
21     public class Student
22     {
23         private string isim;
24         1 reference
25         public string isim
26     {
27         get { return isim; }
28         set { isim = value; }
29     }
30     }
```

Output:

```
C:\WINDOWS\system32\cmd.exe
İsim:Murat
Press any key to continue . . .
```


Getter Setter Tanımlama Kaynak Kodu

```
0 references
internal class Program
{
    0 references
    static void Main(string[] args)
    {
        Student s = new Student();
        s.Name="Murat";
        Console.WriteLine("İsim:" + s.Name);
    }
}

2 references
public class Student
{
    2 references
    public string Name { get; set; }
}
```



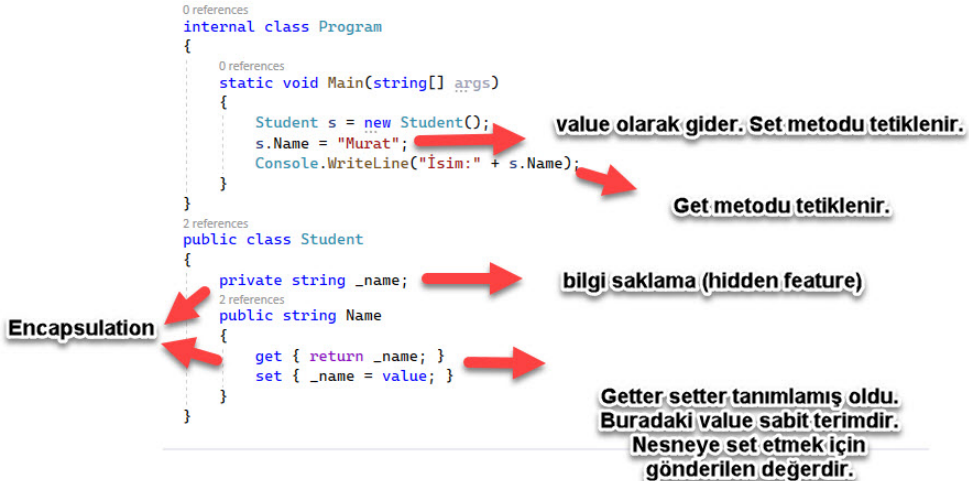
getter ve setter tanımlanmış oldu. Bu sayede bu class'a bir property kazandırmış olduk. Arka planda gizli bir name olduğunu onu değer atamak için de iki adet metod yazıldığını düşünebilirsiniz.

```
C# Ders4Ornek5 Ders4Ornek5.Student
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace Ders4Ornek5
8 {
9     0 references
10     internal class Program
11     {
12         0 references
13         static void Main(string[] args)
14         {
15             Student s = new Student();
16             s.Name="Murat";
17             Console.WriteLine("İsim:" + s.Name);
18         }
19     }
20     2 references
21     public class Student
22     {
23         2 references
24         public string Name { get; set; }
25     }
26 }
```

C:\WINDOWS\system32\cmd.exe

İsim:Murat
Press any key to continue . . .

Getter Setter Hidden property örneği (yaygın kullanım)



```
lers4Ornek6 NNDers4Ornek6.Student Nar
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace NNDers4Ornek6
8 {
9     internal class Program
10     {
11         static void Main(string[] args)
12         {
13             Student s = new Student();
14             s.Name = "Murat";
15             Console.WriteLine("İsim: " + s.Name);
16         }
17     }
18     public class Student
19     {
20         private string _name;
21         public string Name
22         {
23             get { return _name; }
24             set { _name = value; }
25         }
26     }
27 }
```

C:\WINDOWS\system32\cmd.exe

İsim:Murat

Press any key to continue

- Bir kişi için soyadını doğarken aldığını ve daha sonra değiştiremediğini
- Adını doğtuktan sonra değiştirebildiğini
- TC kimlik numarasının doğduktan sonra değiştirebildiğini
- Adı sorunca tamamını söylediğini, adı isminde property tanımlandığını
- Soyadını sorunca ilk ve son harfini gösterdiğini diğerlerini göstermediğini
- TC kimlik numarası sorduğundan son rakamı çift ise tamamını gösterdiğini tek ise olduğu durumda ise 0 gösteren sınıfı ve sınıftan türetilen nesneyi kodlayınız.

Person Class

```
public class Person
{
    private string _surname;
    private int _TCKimlikNO;
    1 reference
    public Person(string surname)
    {
        this._surname = surname;
    }
    2 references
    public string Name
    {
        get;
        set;
    }
    1 reference
    public string Surname
    {
        get
        {
            string ss = _surname.Substring(0, 1);
            for (int i = 0; i < _surname.Length - 1; i++)
                ss += " ";
            ss += _surname.Substring(_surname.Length - 1);
            return ss;
        }
    }
    2 references
    public int TCKimlikNO
    {
        set
        {
            _TCKimlikNO = value;
        }
        get
        {
            if (this._TCKimlikNO % 2 == 0)
                return _TCKimlikNO;
            else
                return 0;
        }
    }
}
```

Feature olarak ifade edilir.

kurucu metod

Property olarak ifade edilir.

sadece get metodu olabilir. Sadece set metodu da olabilir.

Metotların içi özelleştirilebilir.

Source Code

0 references

```
internal class Program
```

```
{
```

0 references

```
static void Main(string[] args)
```

```
{
```

```
    Person p = new Person("Taşyürek");
```

```
    p.Name = "Murat";
```

```
    p.TCKimlikNO = 2222221;
```

```
    Console.WriteLine("Adı:" + p.Name);
```

```
    Console.WriteLine("Soyadı:" + p.Surname);
```

```
    Console.WriteLine("Tc Kimlik No:" + p.TCKimlikNO);
```

```
}
```

```
}
```

3 references

```
public class Person...
```

The screenshot shows the Visual Studio IDE with a C# project named 'NNDers4Ornek7'. The code in 'Program.cs' defines an internal class 'Program' with a 'Main' method and a 'Person' class. The 'Main' method creates a 'Person' object with the name 'Taşyürek', sets the name to 'Murat' and the TCKimlikNO to 2222221, and prints these details. The 'Person' class has private fields for name and TCKimlikNO, and a constructor. The output window shows the execution results: 'Adı:Murat', 'Soyadı:T*****k', 'Tc Kimlik No:0', and a prompt to press any key to continue.

```
9      internal class Program
10     {
11         0 references
12         static void Main(string[] args)
13         {
14             Person p = new Person("Taşyürek");
15             p.Name = "Murat";
16             p.TCKimlikNO = 2222221;
17             Console.WriteLine("Adı:" + p.Name);
18             Console.WriteLine("Soyadı:" + p.Surname);
19             Console.WriteLine("Tc Kimlik No:" + p.TCKimlikNO);
20         }
21         3 references
22         public class Person
23         {
24             private string name, _surname;
25             private int _TCKimlikNO;
26             1 reference
27             public Person(string surname, string name, int TCKimlikNO)
```

Output:

```
C:\WINDOWS\system32\cmd.exe
Adı:Murat
Soyadı:T*****k
Tc Kimlik No:0
Press any key to continue . . .
```


Ödev - Blöf Oyunu

- 2 adet bilgisayar mühendisliği öğrencisi ve 1 adet bilgisayar (kodladığınız oyuncu) kendisi mesleki bilgilerini de kullanarak **nesne yönelimli programlama** felsefesi ile blöf oyunu oynamaktadır.
- Blöf iskambil kartları ile oynanan bir oyundur.
- Oyunda 52 adet kart bulunmaktadır.
- Oyun başlangıcında kartlar iyice karıştırılır. Tüm oyunculara 13 adet dağıtılır.
- Tasarlayacağınız oyunda öğrenci olan 2 oyuncu kullanıcıdan hangi hamleleri yapması gerektiğini sorarken bilgisayar oyuncusu daha önceki yapılan hamleleri de hesaplayıp hamle işlemi yapmalıdır.
- İlgili oyuncu c# dilinde nesne tabanlı programlama tekniği ile kodlayınız.
- 30 Ekim 06.59'a kadar gönderilenler 8+2 puan, vizeye kadar gönderenler 8 puan(kayubmprogramlama1@gmail.com).