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 fonskiyonun çağrılması (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 fonskiyonları statik olarak tanımlanır.
- Fonskiyon nesneden sonra . konmadan çağrılacağından dolayı bu fonksiyonun statik olarak tanımlanması gerekiyor.
- C# dilinde operaratörlere yeni görevler yüklenmesi için "operator" kelimesi kullanılır.

Örnek 1

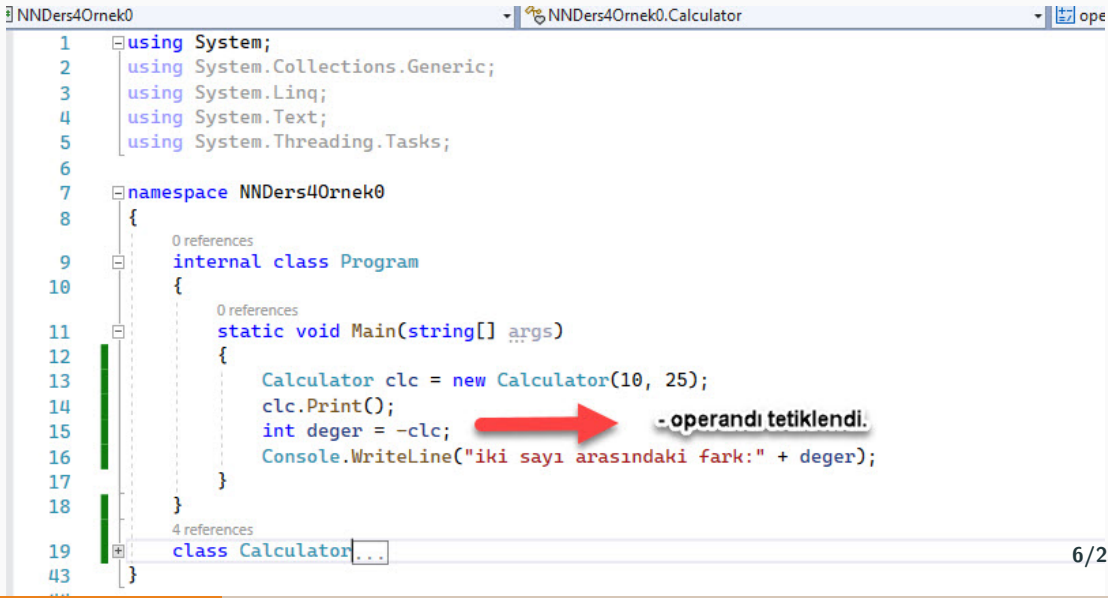
- Operatör ile geriye string, integers veya obje türünde geriye değer döndürülür.
- Calculater isminde iki adet sayı değeri tutan ve bu değerlere başlangıçta değer atamasına izin veren bir class oluşturalım.
- Clasımızın print isminde bir fonksiyonu olsun ve yazmış olduğumuz değerleri ekrana göndersin.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

Console App Calculateer Class

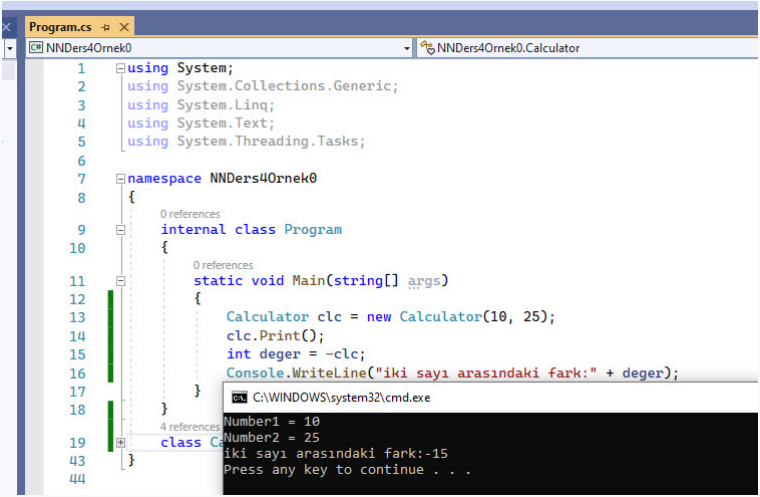
```
class Calculator
{

    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
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace NNDers4Ornek0
{
0 references
internal class Program
{
0 references
static void Main(string[] args)
{
Calculator clc = new Calculator(10, 25);
clc.Print();
int deger = -clc;
Console.WriteLine("iki sayı arasındaki fark:" + deger);
}
}
4 references
class Calculator...
}
- operandı tetiklendi.

Çıktı


Program.cs
NNDers4Ornek0
NNDers4Ornek0.Calculator
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace NNDers4Ornek0
{
0 references
internal class Program
{
0 references
static void Main(string[] args)
{
Calculator clc = new Calculator(10, 25);
clc.Print();
int deger = -clc;
Console.WriteLine("iki sayı arasındaki fark:" + deger);
}
}
4 references
class Ca
}
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 fonskiyonlarla set edilebilsin.
- İki bounding box toplandığında geriye dönen bounding box'ın özellikeri 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 tnaımlama, + operatörünü yeni özellik kazandırma

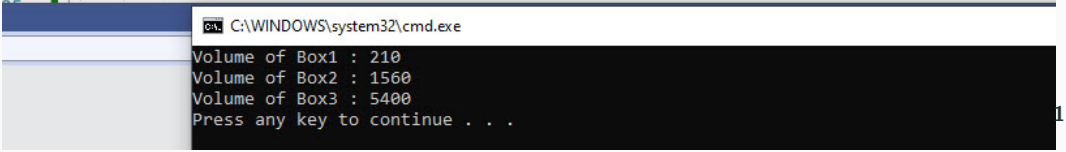
Kaynak Kodu

```
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 çağrılması

Çıktı

```
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);
```



Hatalı kullanım

```
7   namespace NNDers4Ornek1_x
8   {
        0 references
9       internal class Program
10      {
            0 references
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      }
        11 references
46      class Box...
79  }
80
```

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 mnetotlarda getter setter olarak ifade edilir.
- Öğrenci sınıfı ve sadece adı öz niteğiliğin olduğu ve bunun için okuma ve yazma metotlarının olduğu örneği inceleyim.

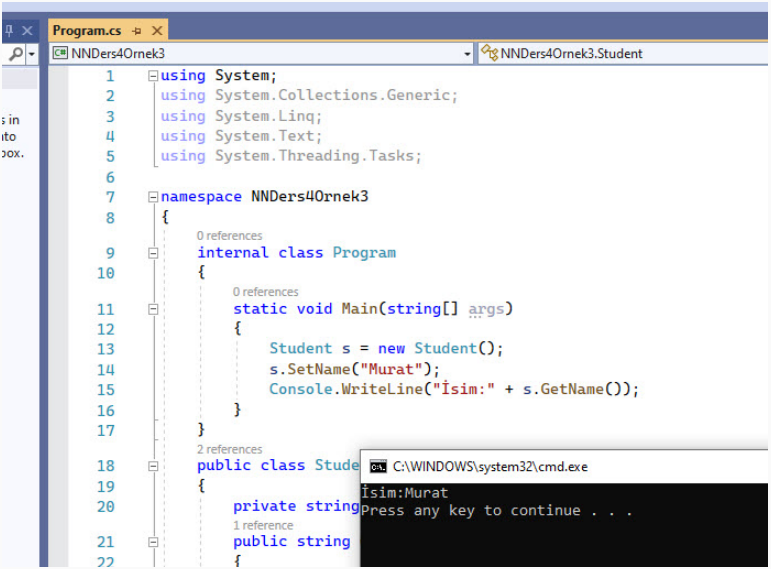
Kaynak Kodu

```
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()
    {
        return this._name;
    }
    1 reference
    public void SetName(string name)
    {
        this._name = name;
    }
}
```

okuma için

değer atama için

Çıktı

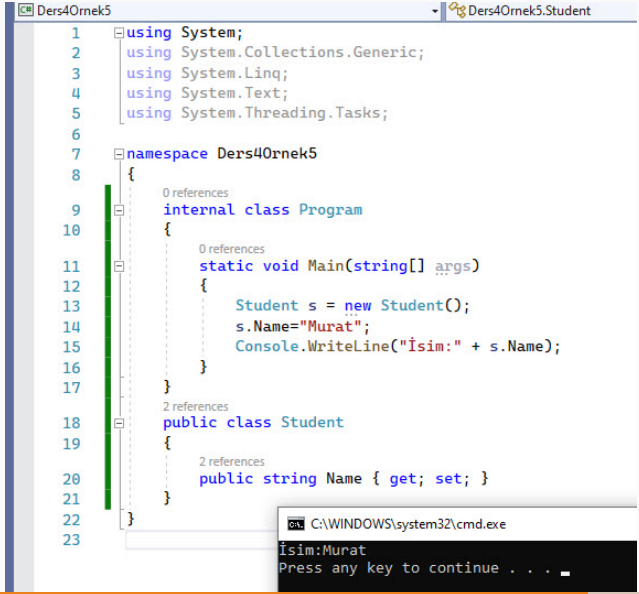

Program.cs
NNDers4Ornek3
NNDers4Ornek3.Student
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace NNDers4Ornek3
{
0 references
internal class Program
{
0 references
static void Main(string[] args)
{
Student s = new Student();
s.SetName("Murat");
Console.WriteLine("İsim:" + s.GetName());
}
}
2 references
public class Stude
{
private string
1 reference
public string
{
C:\WINDOWS\system32\cmd.exe
İsim:Murat
Press any key to continue . . .

Getter Setter Tanımlama Kaynak Kodu

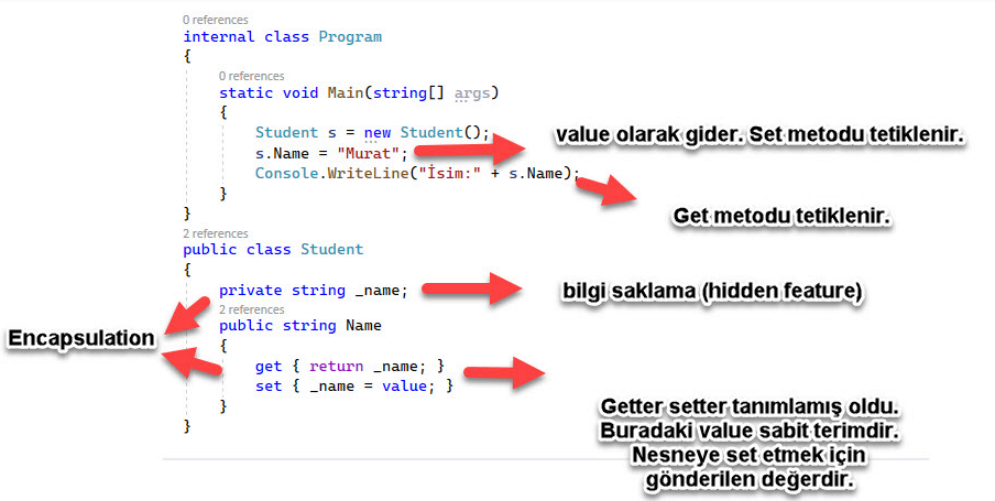
```
0 references
internal class Program
{
    0 references
    static void Main(string[] args)
    {
        Student s = new Student();
        s.Name="Murat";
        Console.WriteLine("İsim:" + s.Name);
    }
}
2 references
public class Student
{
    2 references
    public string Name { get; set;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

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.

Çıktı


Ders4Ornek5
Ders4Ornek5.Student
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Ders4Ornek5
{
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; }
}
}
C:\WINDOWS\system32\cmd.exe
İsim:Murat
Press any key to continue . . .

Getter Setter Hidden property örneği (yaygın kullanım)



Çıktı

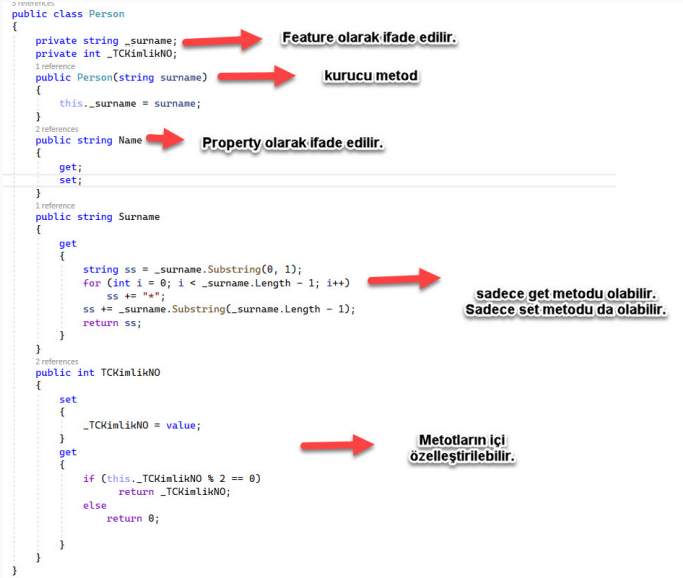
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace NNDers4Ornek6
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Student s = new Student();
            s.Name = "Murat";
            Console.WriteLine("İsim:" + s.Name);
        }
    }
    public class Student
    {
        private string _name;
        public string Name
        {
            get { return _name; }
            set { _name = value; }
        }
```

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

Örnek

- 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.

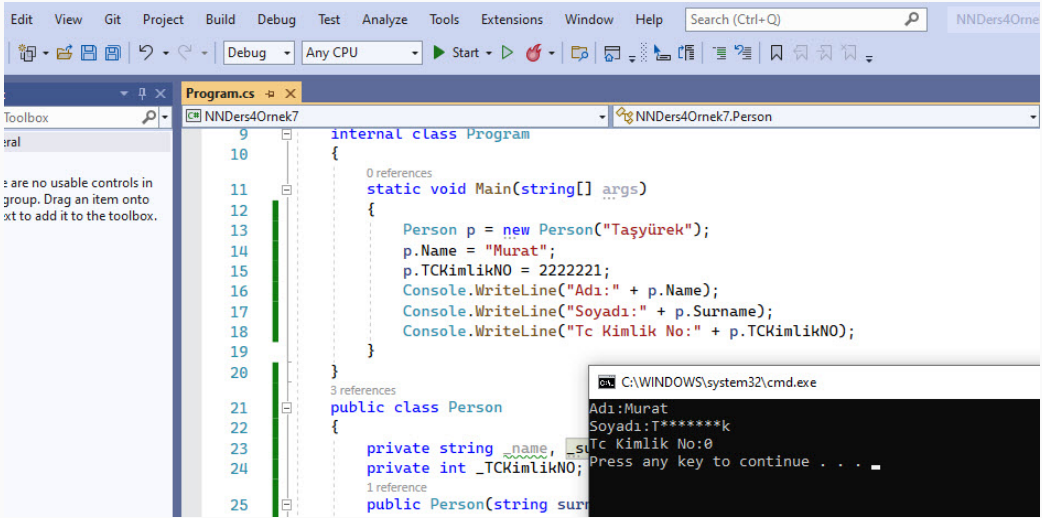
Person Class



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...
```

Çıktı


Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Debug
Any CPU
Start
Toolbox
There are no usable controls in this group. Drag an item onto the text to add it to the toolbox.
Program.cs
NNDers4Ornek7
NNDers4Ornek7.Person
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
{
private string _name, _su
private int _TCKimlikNO;
1 reference
public Person(string surr
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.
- İlgilli 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).