

DATA STRUCTURES

Part – II

Değer ve Referans Veri Tipleri, Metotlar
C# Programlama Dili Örnekleri

Types Overview

- ◆ A C# program is a collection of types
 - Classes, structs, enums, interfaces, delegates
- ◆ C# provides a set of predefined types
 - E.g. `int`, `byte`, `char`, `string`, `object`, ...
- ◆ You can create your own types
- ◆ All data and code is defined within a type
 - No global variables, no global functions

Types Overview

- ◆ Types contain:
 - Data members
 - Fields, constants, arrays
 - Events
 - Function members
 - Methods, operators, constructors, destructors
 - Properties, indexers
 - Other types
 - Classes, structs, enums, interfaces, delegates

Değer ve Referans Veri Tipleri

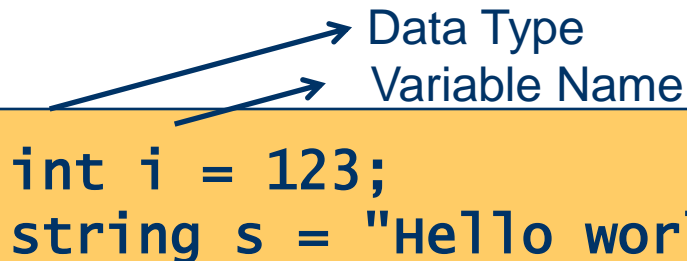
Types Overview

- ◆ Types can be instantiated...
 - ...and then used: call methods, get and set properties, etc.
- ◆ Can convert from one type to another
 - Implicitly and explicitly
- ◆ Types are organized
 - Namespaces, files, assemblies
- ◆ There are two categories of types: value and reference
- ◆ Types are arranged in a hierarchy

Types

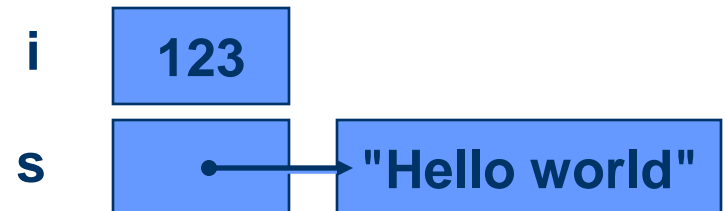
Unified Type System

- ◆ Value types
 - Directly contain data
 - Cannot be null
- ◆ Reference types
 - Contain references to objects
 - May be null



int i = 123;
string s = "Hello world";

Diagram illustrating variable declarations. The label "Data Type" has an arrow pointing to "int" and "string". The label "Variable Name" has an arrow pointing to "i" and "s".



Types

Unified Type System

◆ Value types

- Primitives `int i; float x;`
- Enums `enum State { Off, On }`
- Structs `struct Point {int x,y;}`

◆ Reference types

- Root `object`
- **String** `string`
- **Classes** `class Foo: Bar, IFoo {...}`
- Interfaces `interface IFoo: IBar {...}`
- **Arrays** `string[] a = new string[10];`
- Delegates `delegate void Empty();`

Sabitler ve Değişkenler

- Sabitler, değişmeyen değer ve ifadelerin saklanması amacı ile kullanılırlar.

`const double PI = 3.142857143;`

- Değişkenler, değerleri program içerisinde geçici olarak saklamak için kullanılırlar.

`[Değişken tipi] [Değişken adı] = [İlk Değer]`

METOTLAR

Böl ve Çöz (Divide and Conquer)

- Yazılım Mühendisliği deneyimleri, büyük programlar geliştirmenin en iyi yolunun küçük program parçaları yazıp onları birleştirmek olduğunu göstermiştir.
- Böl ve Çöz olarak bilinen bu yöntem aynı zamanda, hatalardan arındırmayı, programı gelişen şartlara göre büyütmeyi, değişiklikler yapmayı kolaylaştırmak ve anlaşılabilirliği artırmak gibi birçok avantajı da beraberinde getirmektedir.
- C#'ta programları oluşturan en temel bloklar, sınıf (class) ve metotlardır (method). Metotlar yazılım içinde yeniden kullanılarak kodu ve yazılım geliştirme süresini kısaltmaktadır.

Metotlar (Method)

- Bir işlemin yapılması için bir veya daha fazla ifade kullanmak gerekir. Verilen bir matrisi ekrana yazdırmak gibi. İlgili kodu “yazdir()” adını verdiğimiz bir metot içine yazarak istediğimiz zaman, ismi ile çağırabiliriz
- .NET Framework Class Library (FCL) kapsamında, System ad uzayı (<http://msdn.microsoft.com/en-us/library/system.aspx>) matematik hesaplamalarını (Math sınıfı), string, karakter, girdi/çıkıtlı işlemlerini ve diğerlerini yapmak için hazır sınıflar ve metotlar içermektedir. Ayrıca değişik alanlarda hazırlanmış veya kendimizin daha önceden hazırladığı metotları da kullanmak mümkündür.

Çok Kullanılan Hazır Metotlar

Bazı Math Sınıfı Metotları

<http://msdn.microsoft.com/en-us/library/system.math.aspx>

Abs(x)	Mutlak değer	Abs(-5.3) == 5.3
Ceiling(x)	x'i kendinden küçük olmayan en küçük tamsayıya yuvarlar	Ceiling(-9.8) == -9.0
Floor(x)	x'ten büyük olmayan en büyük tamsayıyı döndürür	Floor(-9.8) == -10.0
Cos(x) Sin(x), Tan(x)	Radyan cinsinden trigonometrik fonksiyonlar	Cos(0.0) == 1.0
Exp(x)	e^x	Exp(1.0) yaklaşık 2.718..
Log(x)	Logaritma	Log(2.718) yaklaşık 1.0
Max(x,y), Min(x,y)	Max ve Min fonksiyonları 2 sayıdan büyük/küçük olanı döndürür.	Max(3.7,12.3) == 12.3 Min(3.7,12.3) == 3.7
Pow(x,y)	Üs : x^y	Pow(9.0,.5) == 3.0
Sqrt	Karekök	Sqrt(900.0) == 30.0

METOTLARIN GENEL BİÇİMİ

```
erişim dönüş_tipi isim(parametre listesi)
{
    metodun gövdesi
}
```

Erişim : public, private gibi

Dönüş_tipi : metodun döndürdüğü veri tipi. Değer döndürmüyorsa void.

Parametre listesi : “,” lerle ayrılmış tip ve parametre ismi.

Değer Döndürmeyen Metotlar

void metotları

```
public void yazdir()  
{  
    Console.WriteLine(Merhaba);  
}
```

Çağırılması :
yazdir();

Değer Döndüren Metotlar

```
public int topla(int a, int b)
{
    return (a+b);
}
```

Çağırılması :

```
int y=topla(5,6);
```

Metodun iki de parametresi var.

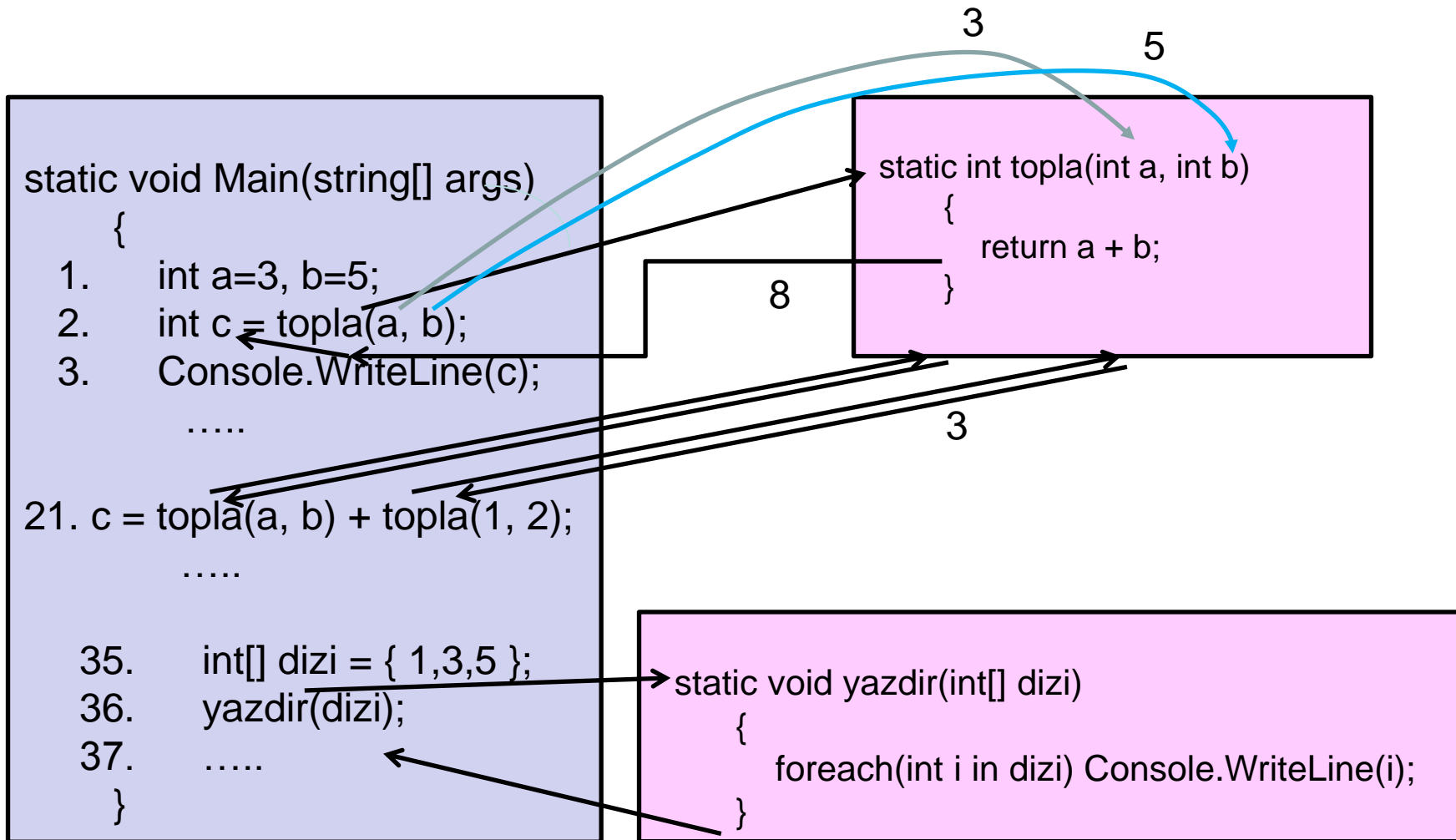
Parametre, Argüman, Return

Argüman : Metoda aktarılan değer

Parametre : Argümanı kabul eden değişken

Return : Metottan çıkmak veya geri dönmek

Metot Çağrımı



Metot Örneği – I

Verilen bir string'i n kere yazdıran metot

```
static void Main(string[] args)
{
    String str = "test";
    nyazdir(str, 10);
    Console.ReadKey();
}
```

```
public static void nyazdir(String str, int n)
{
    for (int i = 0; i < n; ++i)
        Console.WriteLine(str);
}
```

Metot Örneği – II

Faktöryel hesaplayan metot

```
static void Main(string[] args)
{
    int sayi = 5;
    Console.WriteLine(f(sayi));
}

public static String f(int n)
{
    int carpim = 1;
    if (n<0) return "Invalid Input For Function";
    else
        for(int i=1; i<=n;++i)
            carpim*=i;
    return ""+carpim;
}
```

Sayı tipinde veri döndüren faktöryel fonksiyonunu yazınız.

Metoda Değer Veri Tipi Gönderme

```
static void Main(string[] args)
{
    int sayi = 5;
    Console.WriteLine("Metottan Once " + sayi);
    degerArttir(sayi);
    Console.WriteLine("Metottan Sonra " + sayi);
}
```



Metottan önce 5



Metottan sonra 5

```
public static void degerArttir(int deger)
{
    deger += 1;
}
```

Metoda Nesne Gönderme

```
class Program
{
    static void Main(string[] args)
    {
        TamsayıSınıfı d = new TamsayıSınıfı();
        d.sayi = 5;
        Console.WriteLine("Metottan Önce " + d.sayi);
        degerArttir(d);
        Console.WriteLine("Metottan Sonra " + d.sayi);
    }

    public static void degerArttir(TamsayıSınıfı d)
    {
        d.sayi += 1;
    }
}
```

```
class TamsayıSınıfı
{
    public int sayi;
}
```

Metottan önce 5

Metottan sonra 6

Nesne elemanının değeri değişmektedir.

Metoda Dizi Gönderme

```
static void Main(string[] args)
{
    double[] dizi = { 5,5,5,5 };
    Console.WriteLine("\nMetottan Once = ");
    for(int i=0; i<dizi.Length; ++i)
        Console.Write(dizi[i]+" ");
    degerArttir(dizi,2);
    Console.WriteLine("\nMetottan Sonra = ");
    for(int i=0; i<dizi.Length; ++i)
        Console.Write(dizi[i]+" ");
}
```

Metottan Once = 5 5 5 5

Metottan Sonra = 5 5 10 5

```
public static void degerArttir(double[] dizi, int indis)
{
    dizi[indis] += 5;
}
```

Dizi elemanının değeri değişmektedir.



12

Debugging and Handling Exceptions

C# Programming: From Problem Analysis to Program Design
3rd Edition

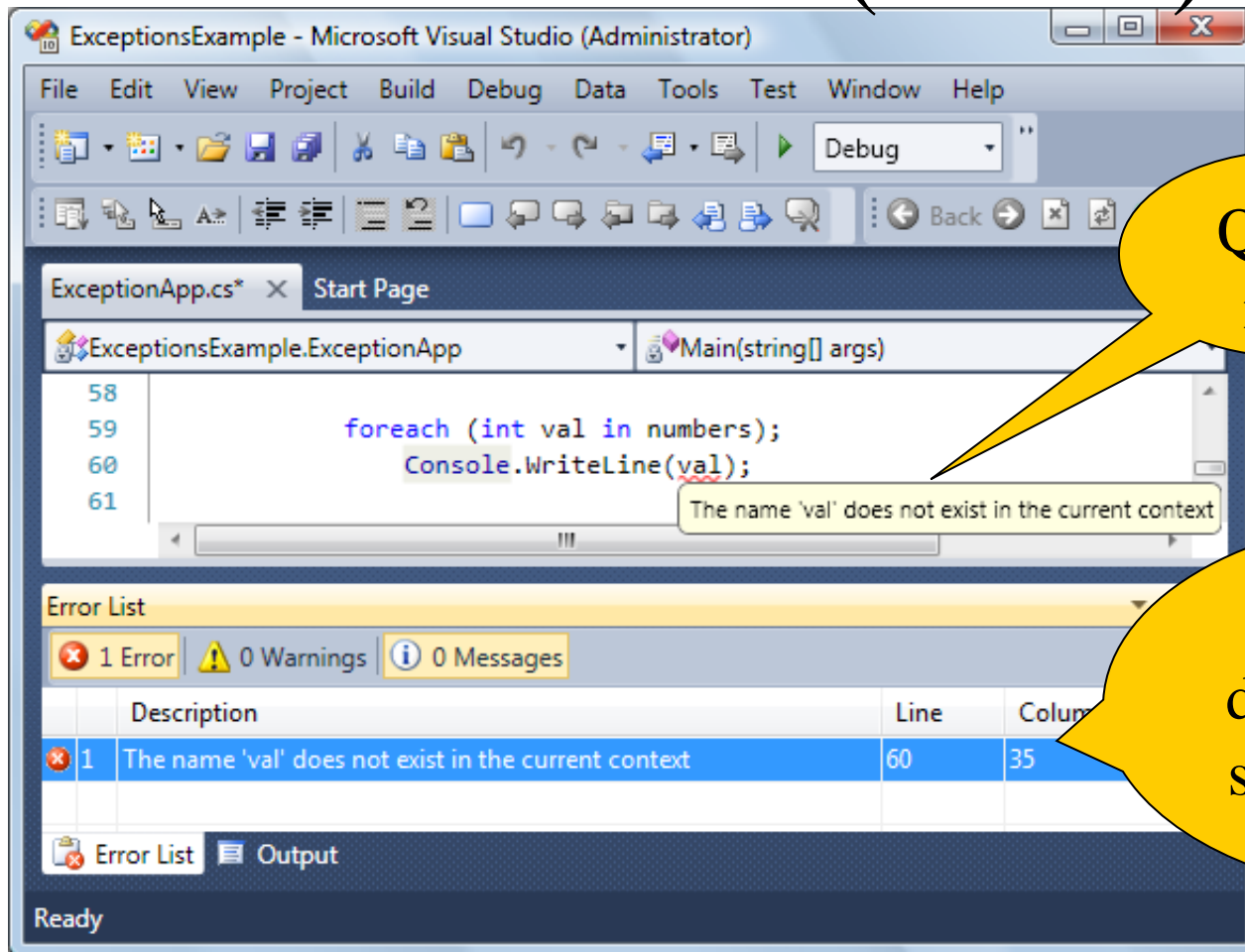
Chapter Objectives

- Learn about exceptions, including how they are thrown and caught
- Gain an understanding of the different types of errors that are found in programs
- Look at debugging methods available in Visual Studio
- Discover how the Debugger can be used to find run-time errors
- Become aware of and use exception-handling techniques to include try...catch...finally clauses
- Explore the many exception classes and learn how to write and order multiple catch clauses

Errors

- Visual Studio IDE reports errors as soon as it is able to detect a problem
- Syntax errors
 - Language rule violation

Errors (continued)



Quick
info

Error message
does not always
state the correct
problem

Figure 11-1 Syntax error – extraneous semicolon

Run-Time Errors

- Just because your program reports no syntax errors does not necessarily mean it is running correctly
- One form of run-time error is a logic error
 - Program runs but produces incorrect results
 - May be off-by-one in a loop
 - Sometimes users enter incorrect values
- Finding the problem can be challenging

Debugging in C#

- Desk check
- Many IDEs have Debuggers
- Debuggers let you observe the run-time behavior
 - You can break or halt execution
 - You can step through the application
 - You can evaluate variables
 - You can set breakpoints
- Debug menu offers debugging options

Debugging in C# (continued)

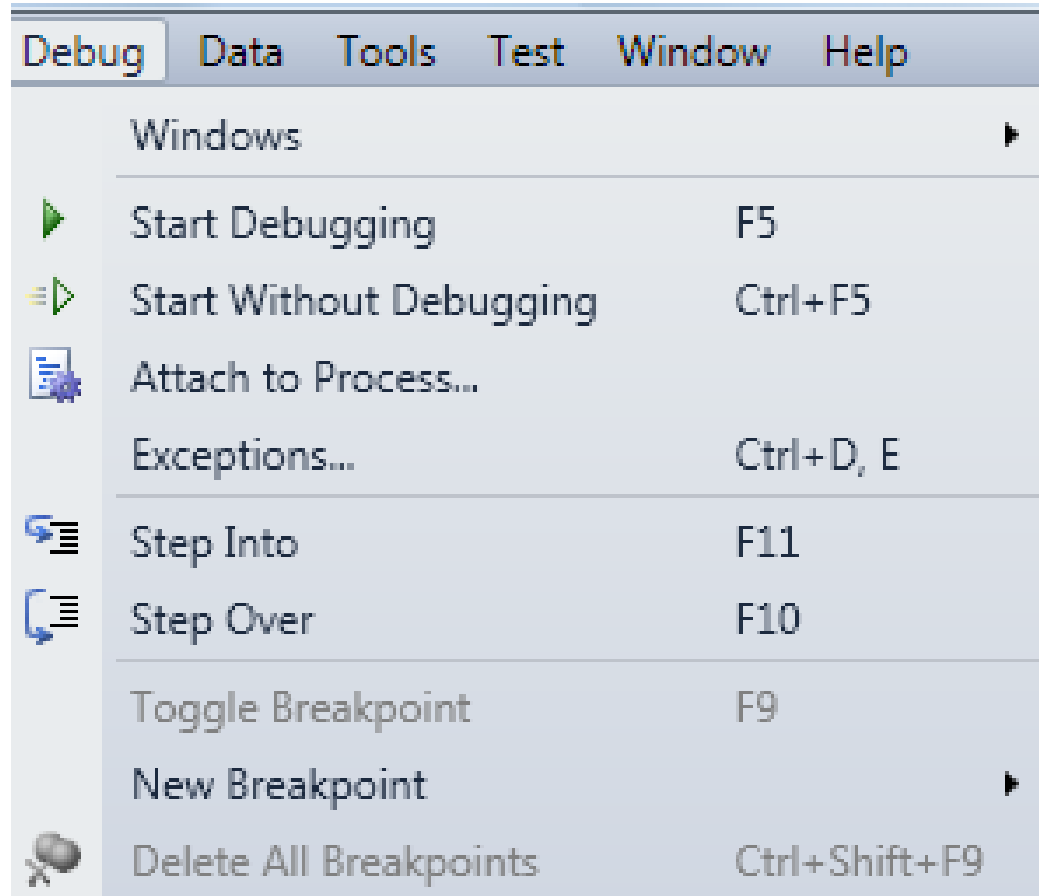
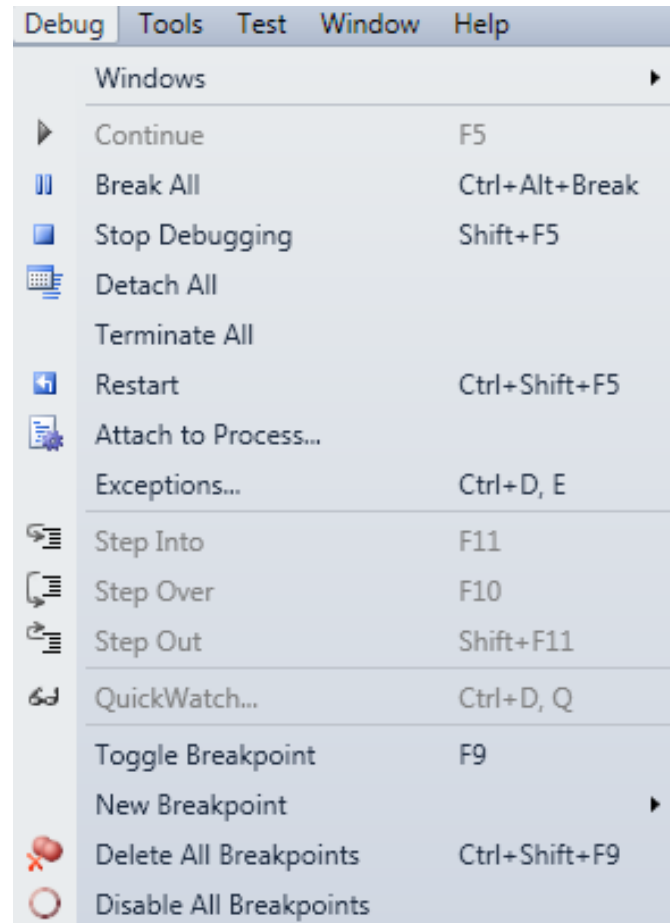


Figure 12-2 Debug menu options

Debugging in C# (continued)



Select **Start Debugging** and number of options to run your program doubles

Figure 12-3 Debug menu options during debugging mode

Breakpoints

- Markers placed in an application, indicating the program should halt execution when it reaches that point
- Break mode
 - Examine expressions
 - Check intermediate results
- Use Debug menu to set Breakpoint
 - F9 (shortcut)
 - Toggles

Breakpoints (continued)

- Red glyph placed on the breakpoint line

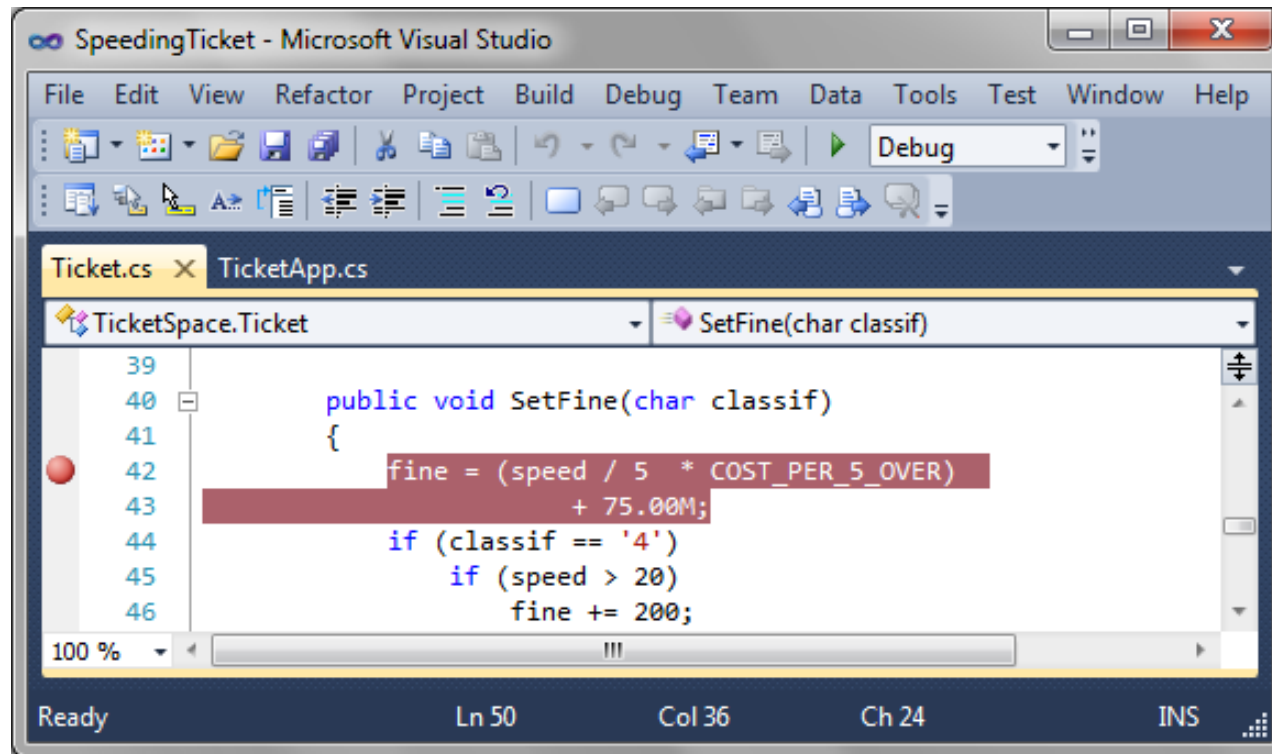


Figure 12-4 Breakpoint set

Break Mode

- In Break mode, Debugger displays Locals window
 - All variables and their values are shown

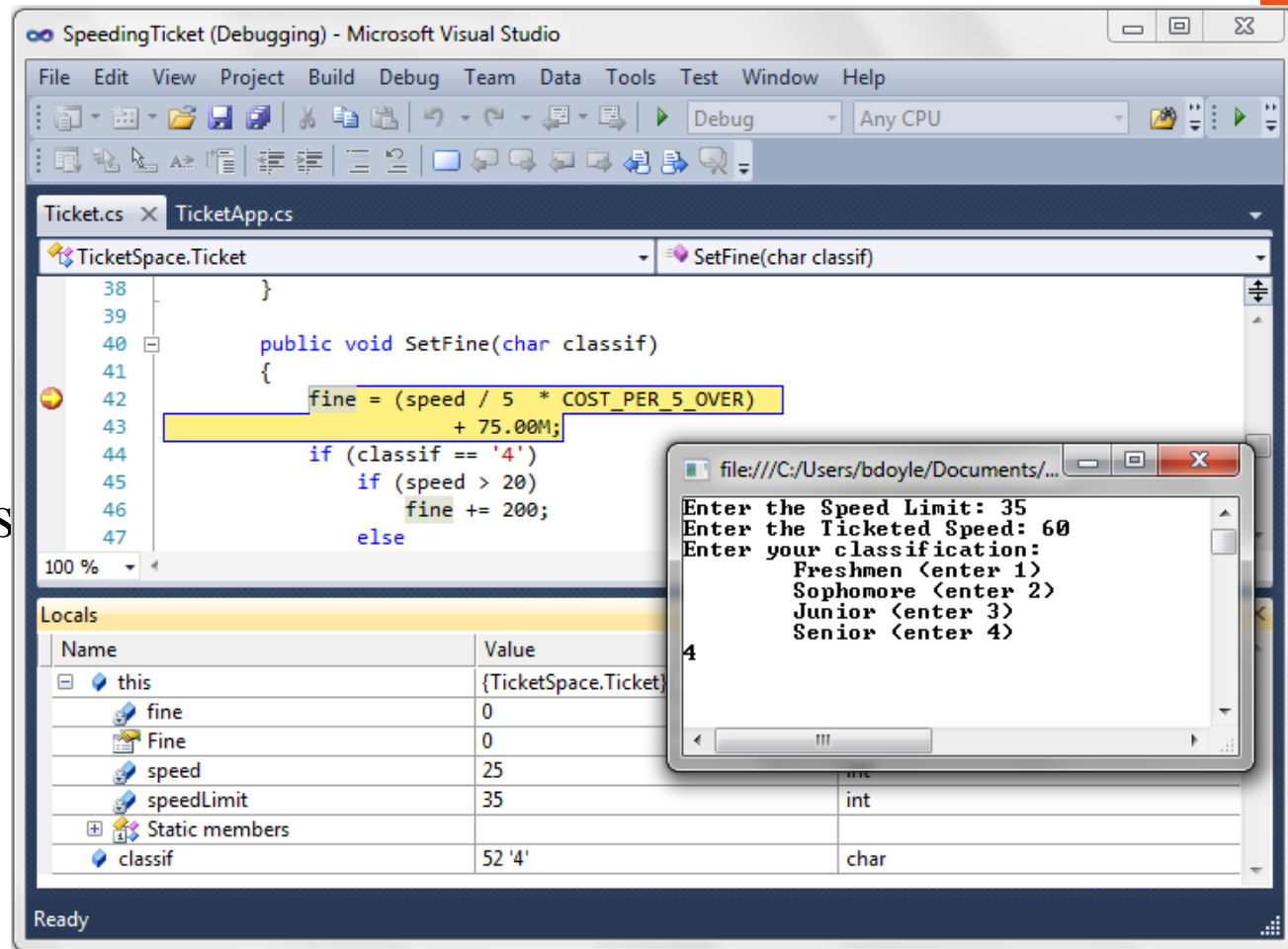


Figure 12-5 Locals window at the breakpoint

Break Mode (continued)

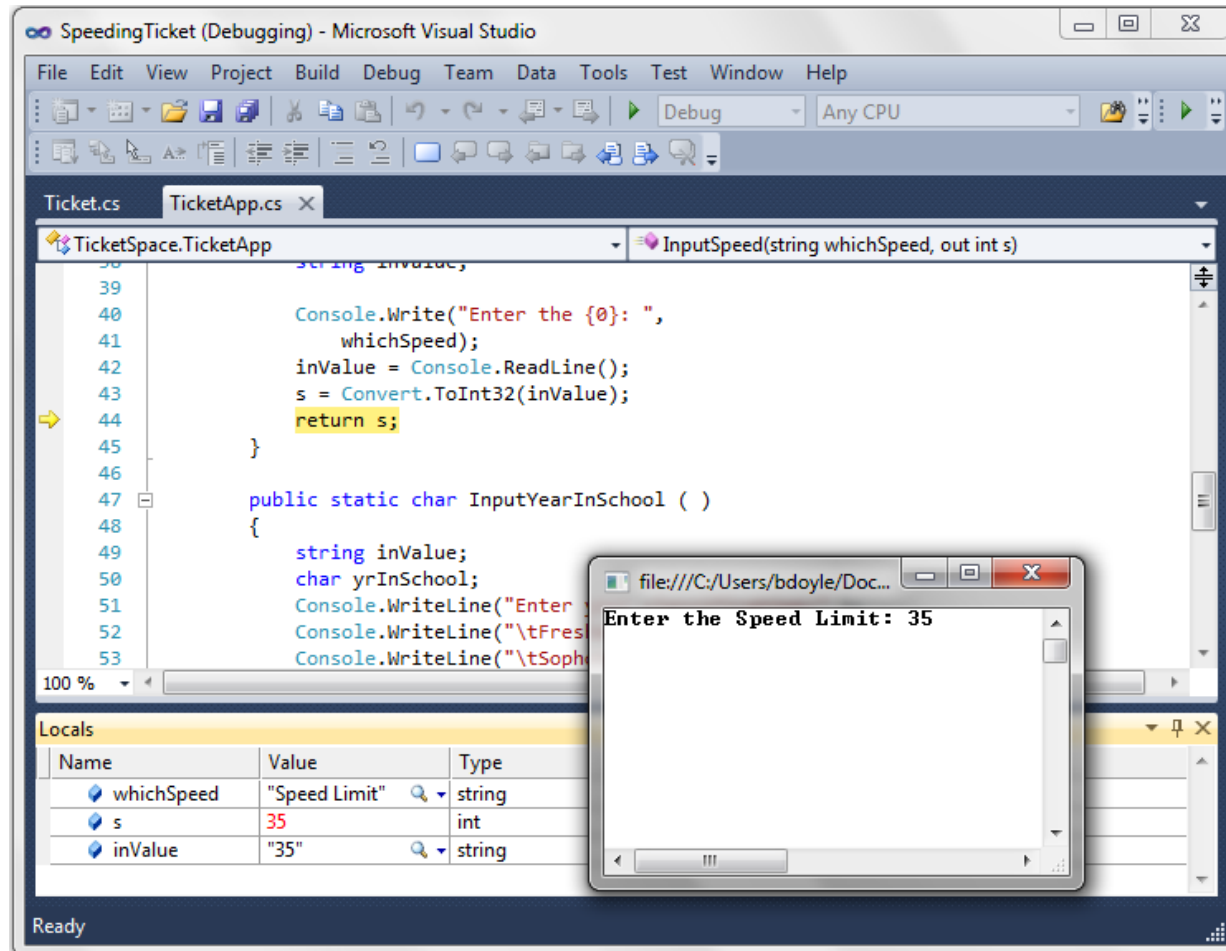


Figure 12-7 Breakpoint location

Debugging in C#

- Continue
 - Takes the program out of break mode and restores it to a run-time mode
 - If more than one breakpoint set, Continue causes the program to execute from the halted line until it reaches the next breakpoint
- Stepping through code
 - Execute code line by line and see the execution path
 - Examine variable and expression values as they change

Stepping Through Code

- Step Into (F11)
 - Program halts at the first line of code inside the called method
- Step Over (F10)
 - Executes the entire method called before it halts
- Step Out (Shift+F11)
 - Causes the rest of the program statements in the method to be executed, and then control returns to the method that made the call

Watches

- Can set Watch windows during debugging sessions
- Watch window lets you type in one or more variables or expressions to observe while the program is running
- Watch window differs from Locals window, which shows all variables currently in scope
- Quick Watch option on Debug menu lets you type a single variable or expression

Watches (continued)

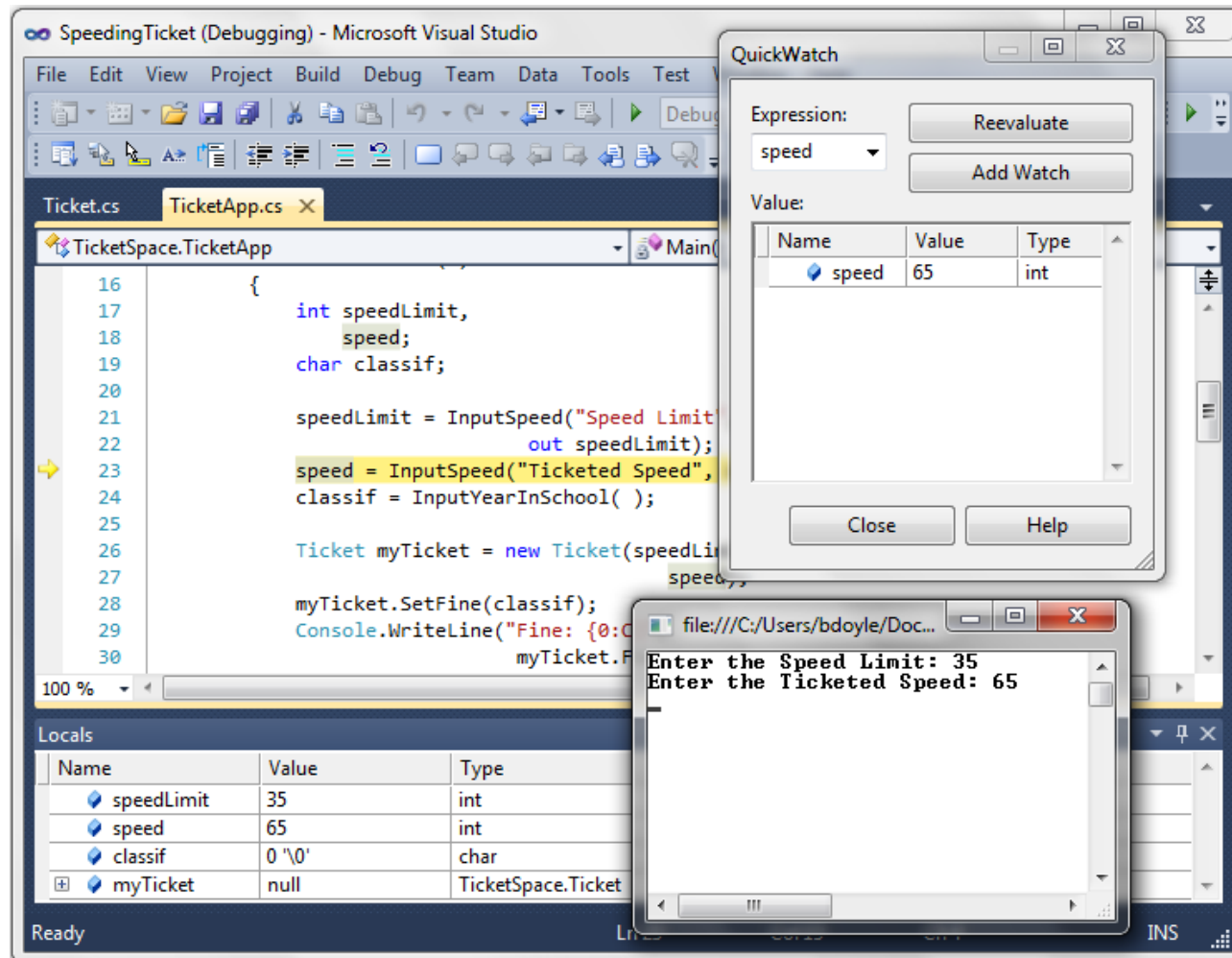


Figure 12-8 QuickWatch window

C# Programlama Dili Örnekleri

ÖRNEKLER BÖLÜMÜNÜN İÇERİĞİ

1. Değişken Tanımlama
2. Aritmetik İşlemler
3. String'ler
4. I/O işlemleri
5. Metotlar
6. Diziler (Array)
7. Denetim Yapıları (if, for, while, ...)
8. GUI ...

Örnek 1

Ekрана Yazdırma, Write ve WriteLine

Ekрана Yazdırma Komutu

```
using System;

class Merhaba
{
    public static void Main(string[] args)
    {
        Console.WriteLine("Merhaba");
    }
}
```

Ekran Çıktısı : Merhaba

Örnek 2

Klavyeden Okuma ve ReadLine

Klavyeden Okuma Komutu ve string

Klavyeden bir metin girilmesini bekler. “Enter” tuşuna basılınca ad değişkenine atanır ve program sonlanır.

```
using System;

class Okuma
{
    public static void Main(string[] args)
    {
        string ad = Console.ReadLine();
    }
}
```

Örnek 3

Veri Tipleri, Değişkenler, İşlemler

```
using System;

class Degiskenler
{
    public static void Main(string[] args)
    {
        double d = 5.8;
        float f = 7.3f;
        int i = 5;
        float fkare = f * f;
        double kareToplam = d * d + f * f + i * i;
        Console.WriteLine(kareToplam);
    }
}
```

Ekran Çıktısı : 111,930002784729


Örnek 4

Veri Tiplerinde Dönüşüm

Tip Dönüşümleri

```
using System;

class TipDonusum
{
    public static void Main(string[] args)
    {
        double sayi = Double.Parse(Console.ReadLine());
        Console.WriteLine("Double : " + Math.Sqrt(sayi) +
            " " + "Int : " + (int)Math.Sqrt(sayi));
    }
}
```



Ekran Çıktısı :
C:\ALG>Ornek4
9,1
Double : 3,01662062579967 Int : 3

Örnek 5

İki sayıyı toplayan metot ve kullanımı

```
using System;

class Topla
{
    public static void Main(string[] args)
    {
        Console.WriteLine(topla(5, 6));
    }

    public static int topla(int sayi1, int sayi2)
    {
        return sayi1 + sayi2;
    }
}
```

Ekran Çıktısı : 11

Örnek 6

Tamsayı, Döngü, Dizi, Metot

int dizi[] = { 5,6,7,8 };
veya benzer şekilde
verilen bir tamsayı
dizisinin elemanlarının
toplamını bulan metodu
içeren C# programını
yazınız.

Ekran Çıktısı : 26

```
using System;

class DiziTopla
{
    public static void Main(string[] args)
    {
        int[] dizi = { 5,6,7,8 };
        Console.WriteLine(topla(dizi));
    }

    public static int topla(int[] dizi)
    {
        int toplam = 0;
        for(int i=0; i<dizi.Length; ++i)
            toplam+=dizi[i];
        return toplam;
    }
}
```

Örnek 7

string'ler

```
// Verilen bir string dizisini, ters sırada (sondan başa doğru)
// listeleyen C# programını yazınız.
using System;

class DiziListele
{
    public static void Main(string[] args)
    {
        string[] strDizi = { "Ali", "Zekiye", "Cemil", "Kemal" };
        int son = strDizi.Length - 1;
        for (int i = son; i >= 0; --i)
        {
            Console.WriteLine(strDizi[i]);
        }
    }
}
```

Ekran Çıktısı :

Kemal
Cemil
Zekiye
Ali

Örnek 8 – 1 (Main metodu)

if, if else

```
using System;
class DiziArama
{
    public static void Main(string[] args)
    {
        string[] strDizi={"Ali", "Zekiye", "Cemil", "Kemal"};

        string kelime = "Cemil";
        if (ara(strDizi,kelime))
            Console.WriteLine(kelime+" Dizide Bulundu");
        else
            Console.WriteLine(kelime+" Dizide Bulunamadı");
        kelime = "Yılmaz";
        if (ara(strDizi,kelime))
            Console.WriteLine(kelime+" Dizide Bulundu");
        else
            Console.WriteLine(kelime+" Dizide Bulunamadı");
    }
}
```

Verilen bir kişi adını bir dizide arayan ve bulunup bulunamadığını belirten C# metodunu yazınız. Aranılan kişinin string aranan = "Ali" şeklinde verildiğini varsayabilirsiniz.

Ekran Çıktısı:

**Cemil Dizide Bulundu
Yılmaz Dizide Bulunamadı**

Örnek 8 – 2 (ara metodu)

if, if else

```
public static bool ara(string[] dizi, string aranan)
{
    for(int i=0; i<dizi.Length; ++i)
        if (aranan.Equals(dizi[i])) return true;

    return false;
}
```

Örnek 9

Diziler

Boş bir diziye arka arkaya eleman ekleyen metodu içeren C# programını yazınız.

```
using System;
class DiziElemanEkle
{
    static string[] strDizi;
    static int elemanSayac = 0;

    public static void Main(string[] args)
    {
        strDizi = new String[10];

        elemanEkle("Ali");
        elemanEkle("Cemil");
        listele();
    }

    public static void elemanEkle(string yeniEleman)
    {
        strDizi[elemanSayac]=yeniEleman;
        elemanSayac++;
    }

    public static void listele()
    { for(int i=0; i<strDizi.Length; ++i)
        Console.WriteLine(strDizi[i]); }
}
```

Örnek 10

Matrisler

// 2 x 4'lük bir matris oluşturan ve elemanlarını listeleyen C# programını yazınız.

```
class MatrisListele
```

```
{
```

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

```
    { int[,] matris = { { 5,6,7,8 }, { 9, 10, 11, 12} };
```

```
        listele(matris); }
```

```
    public static void listele(int[,] matris)
```

```
    {
```

```
        for(int i=0; i<2; ++i)
```

```
        {
```

```
            for(int j=0; j<4; ++j)
```

```
                Console.Write(matris[i,j]+" ");
```

```
                Console.WriteLine();    }
```

```
    }
```

```
}
```

Ekran Çıktısı:

5 6 7 8

9 10 11 12

Örnek 11

String'ler

```
class Stringler
{ public static void Main(string[] args)
  { string s= "abcdefghijklmnpqrstuvwxyabcde";
    // e harfinin alfabadeki konumu
    Console.WriteLine(s.IndexOf('e'));
    // e harfinin 20. karakterden sonra konumu
    Console.WriteLine(s.IndexOf('e',20));
    // 5. karakterden 10 karakterlik string parçası
    Console.WriteLine(s.Substring(5,10));
    // String birleştirme
    Console.WriteLine(String.Concat(s,"ABCDEFGH"));
    // String atama
    s = "Merhaba"; Console.WriteLine(s);
    char[] charArray= new char[7];
    s.CopyTo(0,charArray,0,7);
    Console.WriteLine(charArray);
    s = s + new string(charArray);
  }
}
```

Ekran Çıktısı:

```
4
30
fghijklmno
abcdefghijklmnpqrstuv
wxyzabcdeABCDEFGH
Merhaba
Merhaba
```

BASİT ALIŞTIRMALAR

1. Verilen bir ismin, bir string dizisindeki kaçınıcı eleman olduğunu bulan programı yazınız.
2. Verilen bir ismin, bir string dizisinde kaç kere tekrarlandığını bulan programı yazınız.
3. Bir tamsayı dizisinde, belirtilen bir sayıdan küçük kaç tane sayı olduğunu bulan programı yazınız.
4. Sıralı bir tamsayı dizisinden, verilen bir sayıyı silen metodu yazınız.
5. Sıralı bir diziye, verilen bir sayıyı ekleyen metodu yazınız.
6. Parametre olarak gönderilen iki tane matrisi toplayarak üçüncü matrisi elde eden metodu yazınız.
7. Bir matrisin satırları toplamını bir diziye aktaran metodu yazınız.
8. "Random" sayılardan oluşturduğunuz 10 elemanlı bir dizinin çift numaralı elemanlarını bir matrisin ilk satırına, tek numaralı elemanlarını ikinci satırına yerleştiren C# metodunu yazınız.

Right-click your project in Solution Explorer and select Add reference... and then find System.Windows.Forms and add it.

Örnek 12

Mesaj Kutusu Kullanımı

```
using System;
using System.Windows.Forms;

class MesajKutusu
{
    public static void Main(string[] args)
    {
        string sayi1, sayi2;
        int tamsayi1, tamsayi2, toplam, carpim, fark,
        kalan;
        float bolum;

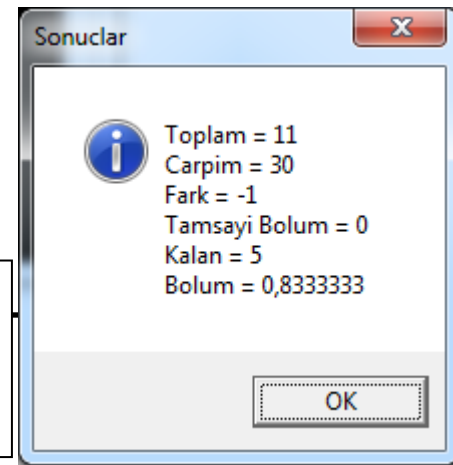
        Console.WriteLine("1.sayiyi veriniz");
        sayi1 = Console.ReadLine();
        Console.WriteLine("2.sayiyi veriniz");
        sayi2 = Console.ReadLine();

        tamsayi1 = Int32.Parse(sayi1);
        tamsayi2 = Int32.Parse(sayi2);
```

```
1.sayiyi veriniz
5
2.sayiyi veriniz
6
```

```
        toplam = tamsayi1 + tamsayi2;
        carpim = tamsayi1 * tamsayi2;
        fark = tamsayi1 - tamsayi2;
        bolum = tamsayi1 / tamsayi2;
        kalan = tamsayi1 % tamsayi2;

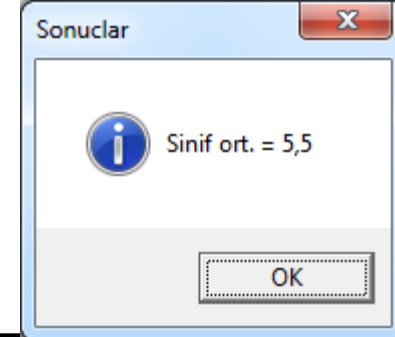
        MessageBox.Show("Toplam = " + toplam +
            "\nCarpim = " + carpim +
            "\nFark = " + fark + "\nTamsayi Bolum = " +
            bolum + "\nKalan = " + kalan +
            "\nBolum = " + (float)tamsayi1 / tamsayi2,
            "Sonuclar", MessageBoxButtons.OK,
            MessageBoxIcon.Information);
    }
}
```



Notu giriniz (Exit : -1)
5
Notu giriniz (Exit : -1)
6
Notu giriniz (Exit : -1)
-1

Örnek 13

While Döngüsü Kullanımı



Not ortalamasını bulan C# programı (-1 değeri girilene kadar notları okur.

```
using System;  
using System.Windows.Forms;
```

```
class NotOrt  
{  
    public static void Main(string[] args)  
    {  
        float ortalama;  
        int sayac=0, notu, toplam=0;  
  
        Console.WriteLine("Notu giriniz (Exit : -1)");  
        string str = Console.ReadLine();  
        notu = Int32.Parse(str);
```

```
        while(notu!=-1) {  
            toplam += notu; ++sayac;  
            Console.WriteLine("Notu giriniz (Exit : -1)");  
            str = Console.ReadLine();  
            notu = Int32.Parse(str);  
        };  
  
        string s;  
        if (sayac==0) s = "Not girilmedi!";  
        else s = "Sinif ort. = "+(float)toplam/sayac;  
  
        MessageBox.Show(s,"Sonuclar",  
            MessageBoxButtons.OK,MessageBoxIcon.Information);  
    }  
}
```

Projeyi, Form Uygulaması olarak açınca otomatik gelen ad uzayları

Örnek 14

Form Uygulaması

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication4
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

RichTextBox

```
public int kare(int i)
{ return i * i; }
```

```
private void button1_Click(object sender, EventArgs e)
{
    string str = "n" + "\t" + "kare(n)\n";
    for (int i = 0; i < 10; ++i)
        str += "" + i + "\t" + kare(i) + "\n";

    richTextBox1.Text = str;
}
```

Button

	kare(n)
0	0
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81

Yazdır

Örnek 15

Random Sayı Üretme ve Kullanma

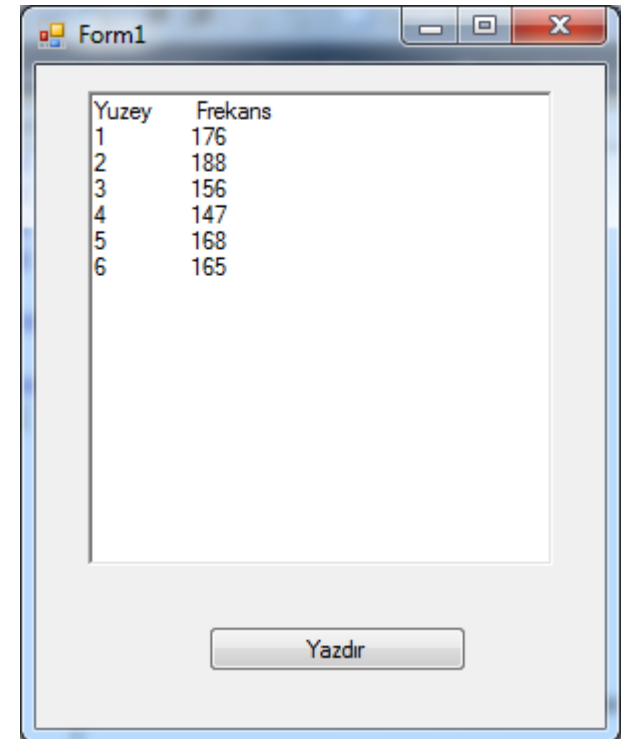
```
private void Form1_Load(object sender, EventArgs e)
{
    Random r = new Random();

    int[] frekans; frekans = new int[6];

    for (int tekrar = 0; tekrar < 1000; ++tekrar)
        frekans[(int)(r.Next(6))]+=;

    metAlan.ReadOnly = true;

    metAlan.Text = "Yuzey \t Frekans";
    for (int i = 0; i < 6; ++i)
        metAlan.AppendText("\n" + (i + 1) + "\t" + frekans[i]);
}
```



**Properties penceresinde
richTextBox1 adı metAlan
olarak değiştirilmeli
(Name sahası)**

Örnek 16

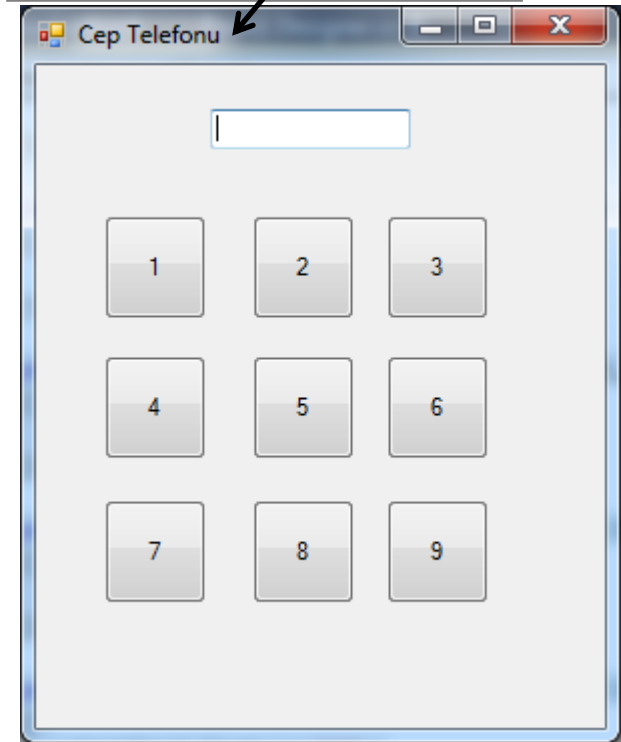
Cep Telefonu Form Uygulaması

```
private void button1_Click_1(object sender, EventArgs e)
{
    textBox1.Text += "1";
}

private void button2_Click(object sender, EventArgs e)
{
    textBox1.Text += "2";
}

private void button3_Click(object sender, EventArgs e)
{
    textBox1.Text += "3";
}
```

Formun Text sahası
Properties penceresinde
Cep Telefonu yapılmalı



Değişkenlerin Kapsama Alanı

```
static void Main(string[] args)
{
    int x = 5;
    if (x > 5)
    {
        int y = 12; } y degiskeninin tanımlı olduğu bölge
        // ...
    }
    // y tanımlı değil
    // x tanımlı
    for (int i = 0; i < 5; ++i)
        Console.WriteLine(i);
    i = 10; // Hatalı!
}
```