

DATA STRUCTURES

Part – I

C# Programlama Dili Temelleri

DERSİN İÇERİĞİ

1. C# DİLİNE GİRİŞ
2. KONSOL UYGULAMASI GELİŞTİRİLMESİ
3. FORM UYGULAMASI GELİŞTİRİLMESİ
4. VERİ TİPLERİ ve DEĞİŞKENLER
5. PROGRAM KONTROL (DENETİM) YAPILARI
6. DİZİLER, STRING ve KARAKTER DİZİLERİ

C# DİLİNE GİRİŞ

- C#, "event-driven", nesne yönelimli ve görsel bir programlama dilidir.
- Web tabanlı uygulamaların ve mobil iletişim cihazlarının yaygınlaşması sonucu, programlama ortamlarında oluşan gereksinimleri karşılamak ve yaşanmaya başlayan sorunları ortadan kaldırmak için .NET platformu ve C# programlama dili geliştirilmiştir. (Microsoft)
- C# Programları, IDE (Integrated Development Environment) kullanılarak hazırlanır. IDE ortamında, programların yazılması, işletilmesi, test edilmesi ve hatalardan arındırılması kolay olduğu için, bu şekilde uygulama yazılması işlemine RAD (Rapid Application Development) adı verilmektedir.

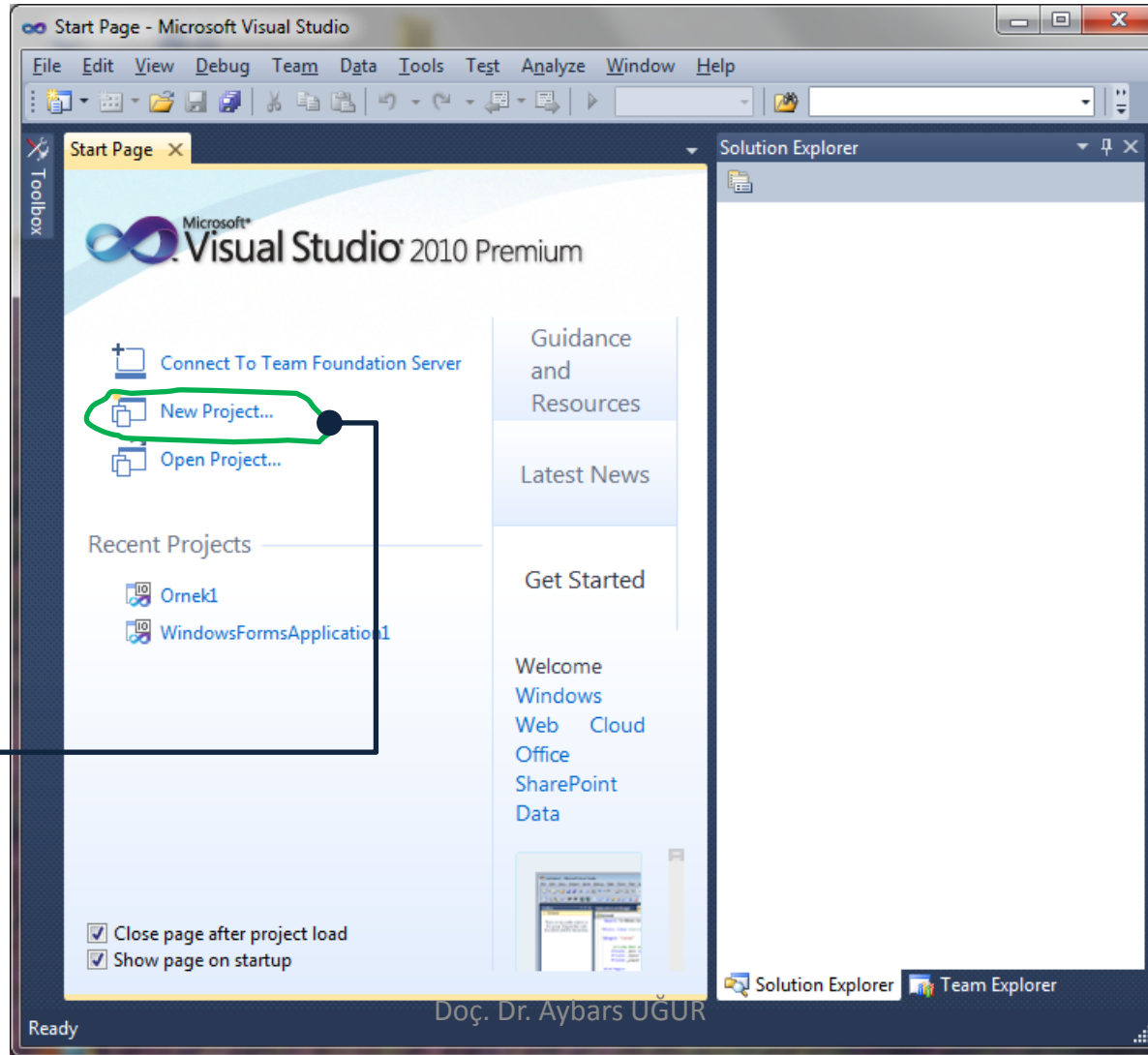
C# Sürümleri

Version	Language specification		Microsoft	Date	.NET Framework	Visual Studio
	ECMA	ISO/IEC				
C# 1.0	December 2002	April 2003	January 2002	January 2002	.NET Framework 1.0	Visual Studio .NET 2002
C# 1.2			October 2003	April 2003	.NET Framework 1.1	Visual Studio .NET 2003
C# 2.0	June 2006	September 2006	September 2005^[A]	November 2005	.NET Framework 2.0	Visual Studio 2005
C# 3.0	None ^[B]		August 2007	November 2007	.NET Framework 2.0 (Except LINQ/Query Extensions) [1] .NET Framework 3.0 (Except LINQ/Query Extensions) [2] .NET Framework 3.5	Visual Studio 2008 Visual Studio 2010
C# 4.0			April 2010	April 2010	.NET Framework 4	Visual Studio 2010
C# 5.0			n/a	August 2012	.NET Framework 4.5	Visual Studio 2012

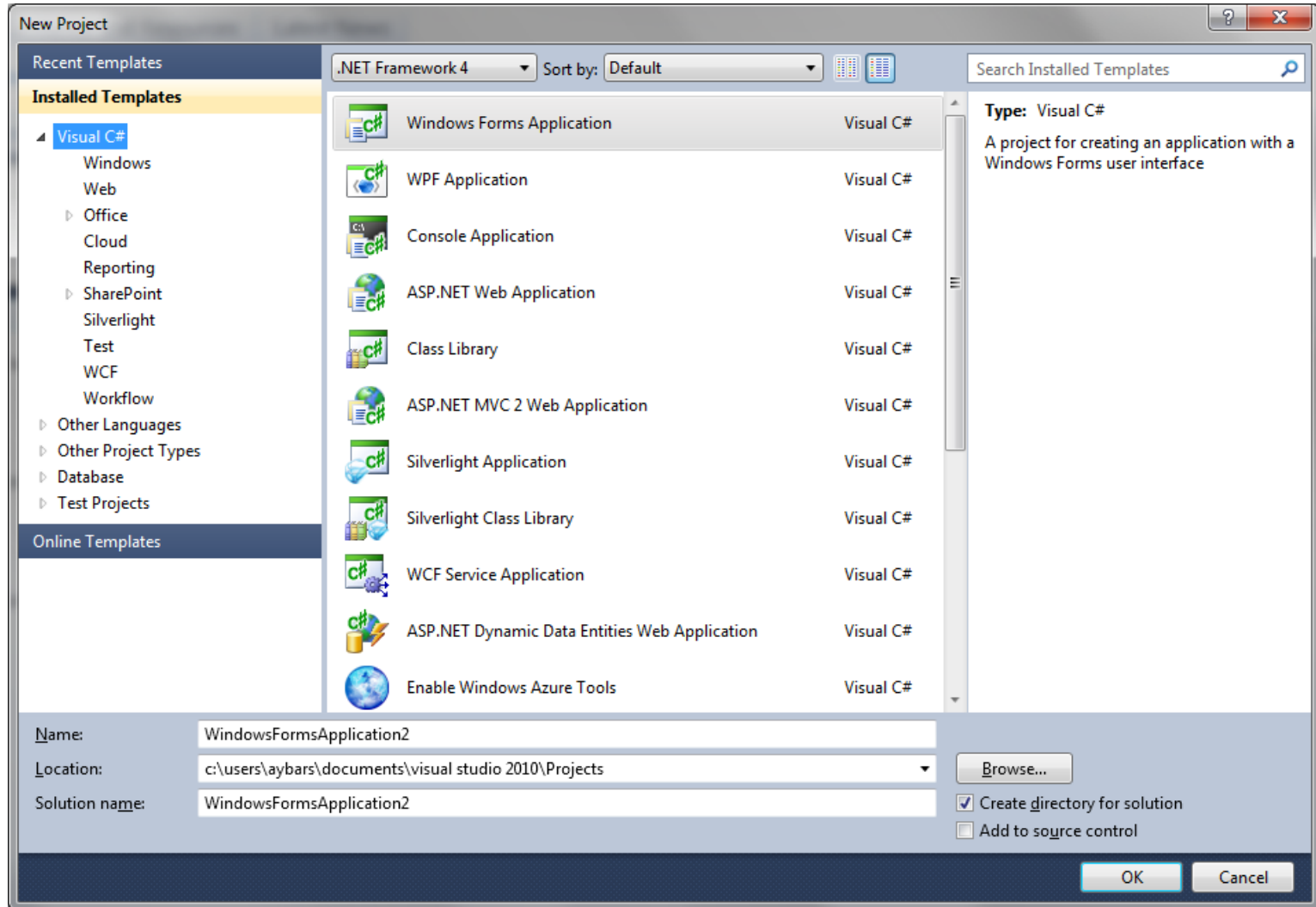
Visual Studio 2010

Açılış Ekranı

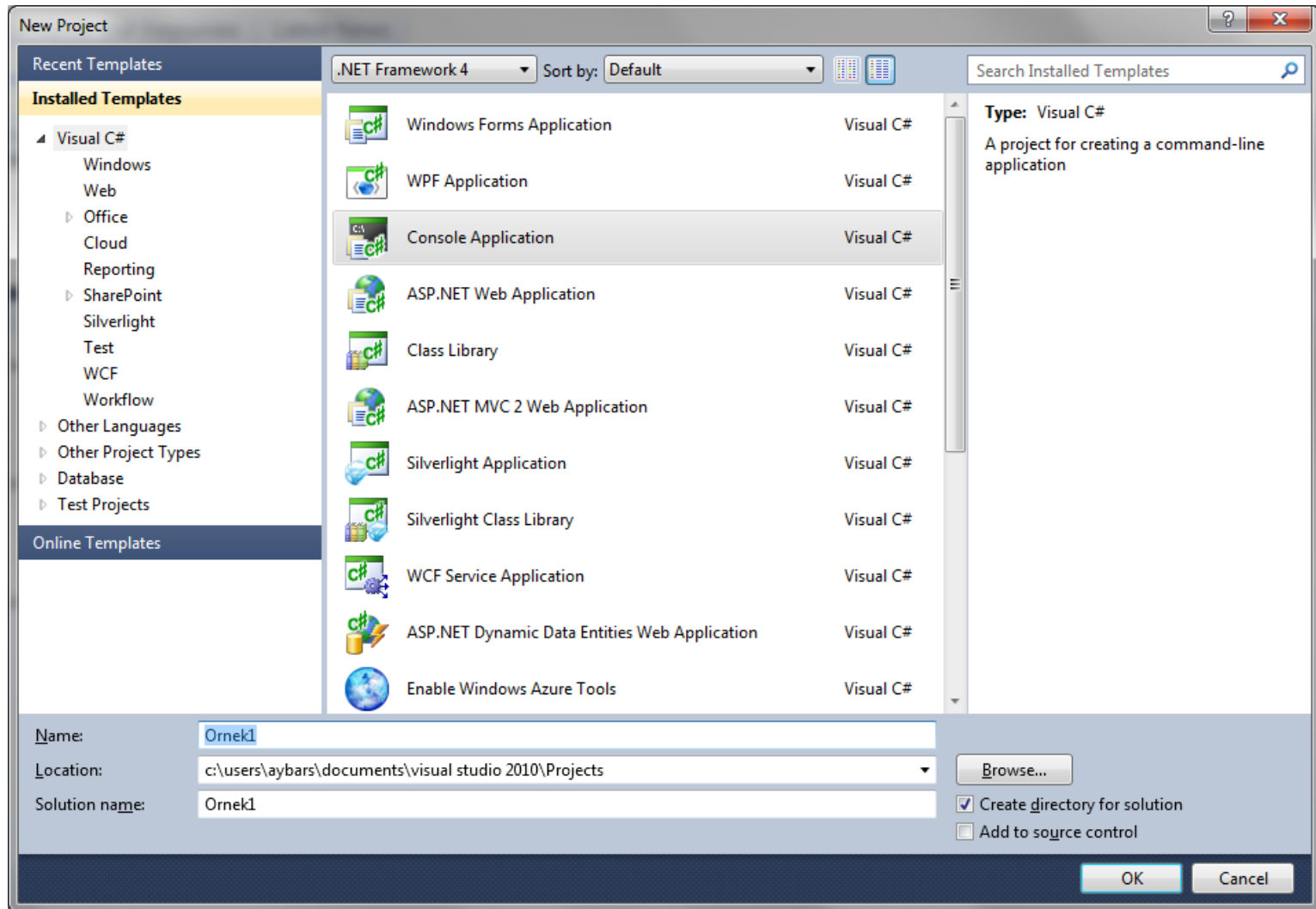
Yeni proje
oluşturulması



Yeni Proje Ekranı



Konsol Uygulaması Oluşturulması ve İsim Verilmesi



Ornek 1: Konsol Uygulaması Oluşturmak İçin İşlem Sırası

- "New Project" Düğmesi ile yeni bir proje açılır.
- Konsol Uygulaması yapacağımız için "Console Application" simgesi seçilir.
- (Visual C# şablonundan) Projeye verilmek istenen isim "Name" kısmına yazılır ve "Location" kısmında Projenin dosyalarının tutulacağı yer belirtilir. Burada Proje ismi olarak "Ornek1" yazılmıştır.
- "OK" düğmesine basılır. Karşımıza sonraki sayfadaki kod gelir.

İlk Program

Programlar, F5 tuşu, menüden Debug -> Start Debugging veya Start Debugging düğmesi  ile çalıştırılır.

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

```
namespace Ornek1
```

```
{
```

```
    class Program
```

```
    {
```

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

```
        {
```

```
        }
```

```
    }
```

```
}
```



Program başlangıç noktası
Kod buraya yazılacaktır

İlk Program ile Konsol'a Merhaba Yazdırılması

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

```
namespace Ornek1
```

```
{
```

```
    class Program
```

```
    {
```

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

```
        {
```

```
            Console.WriteLine("Merhaba");
```

```
        }
```

```
    }
```

```
}
```

Program bittiği
için Konsol
penceresi
kapanır!



Konsol ekranının bekletilmesi

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

```
namespace ConsoleApplication1
```

```
{
```

```
    class Program
```

```
    {
```

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

```
        {
```

```
            Console.WriteLine("Merhaba");
```

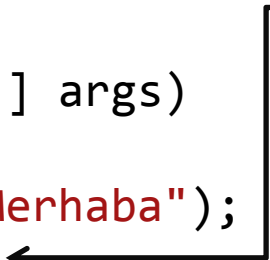
```
            Console.ReadKey();
```

```
        }
```

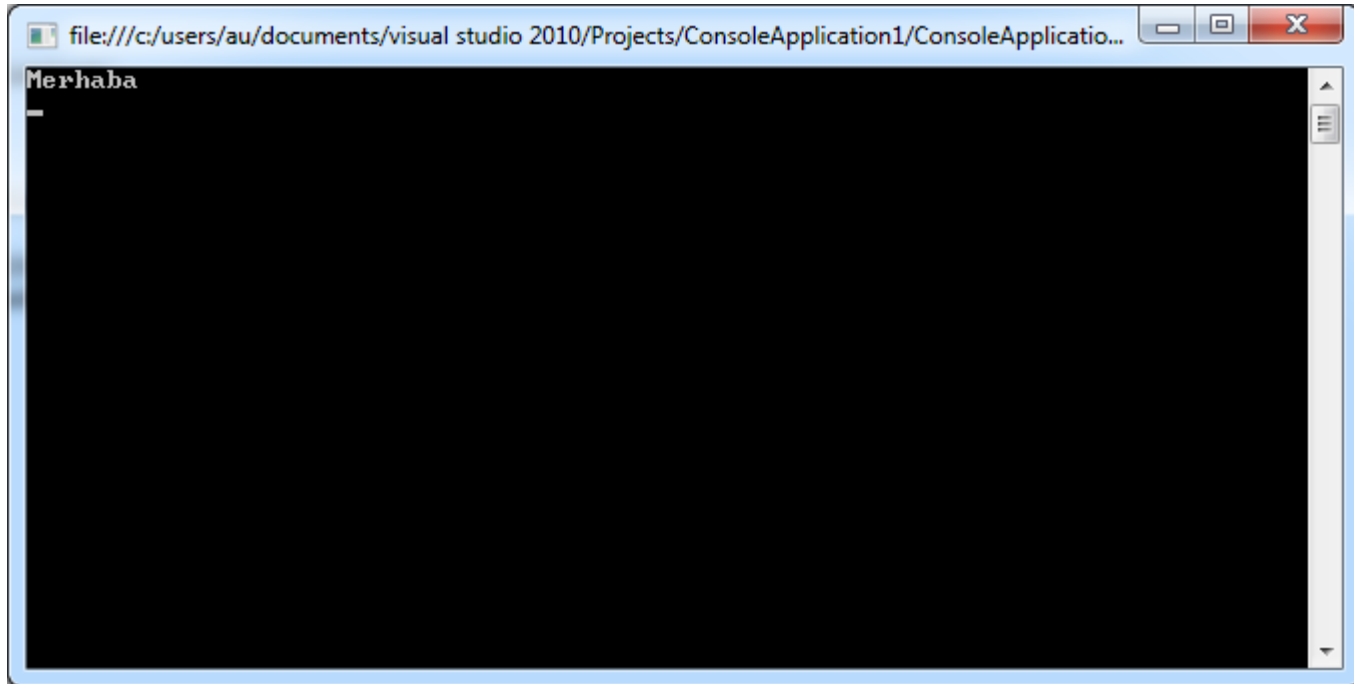
```
    }
```

```
}
```

Bir tuşa basana
kadar bekletir

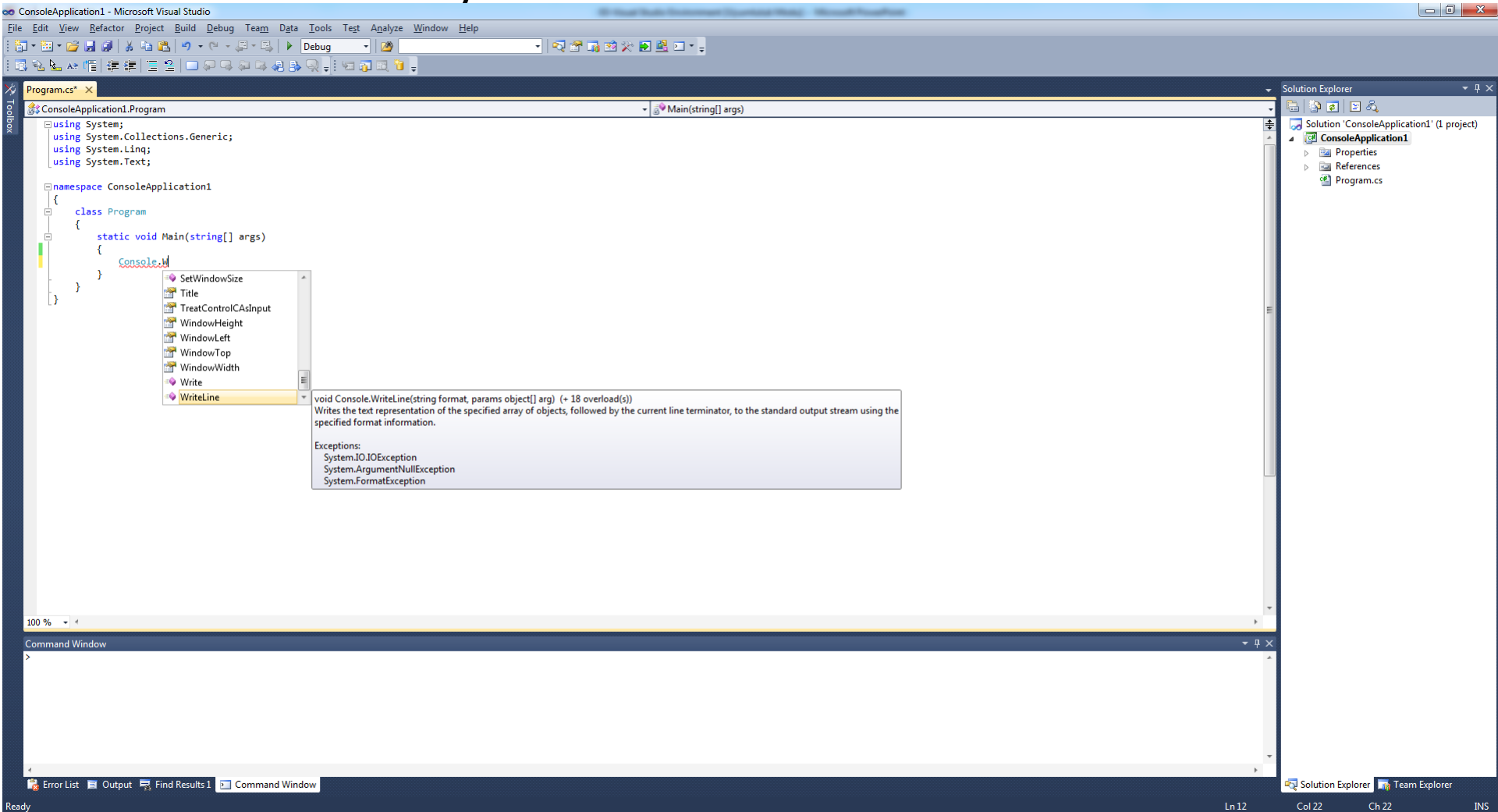


Konsol Ekranı

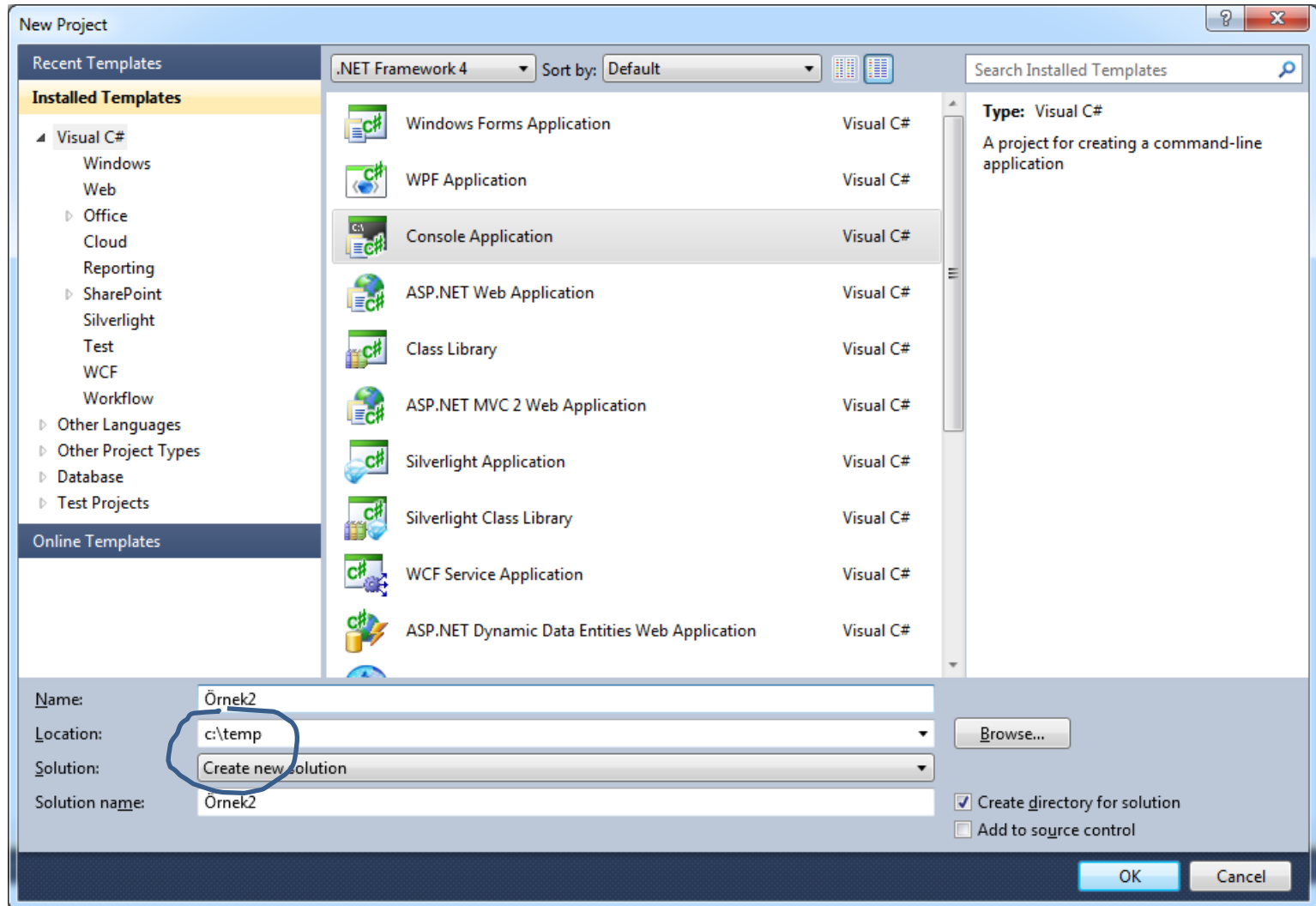


Intellisense (Autocompletion)

Programcının yazmakta olduğu sembol ve değişken isimlerini, fonksiyonları otomatik olarak tamamlar.

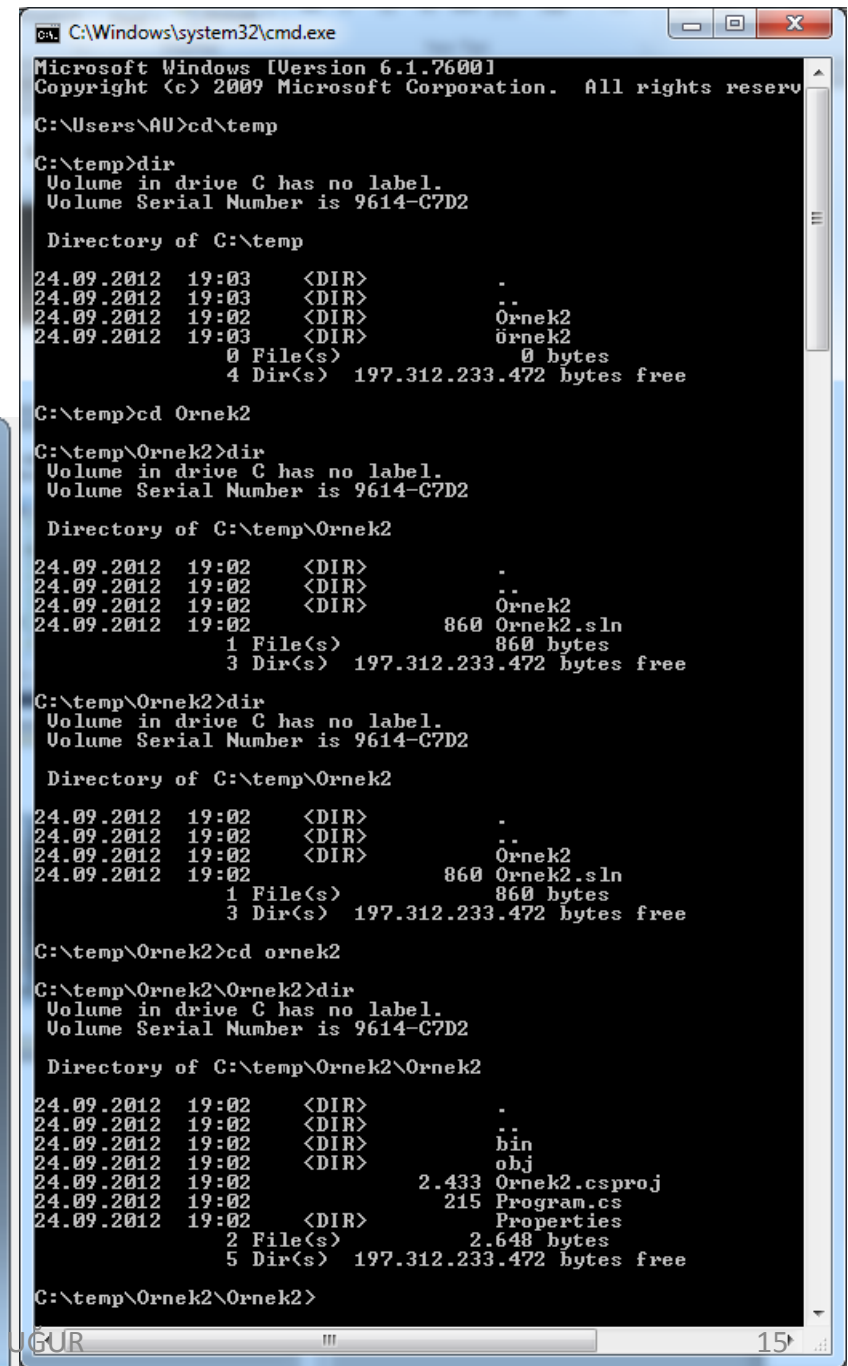
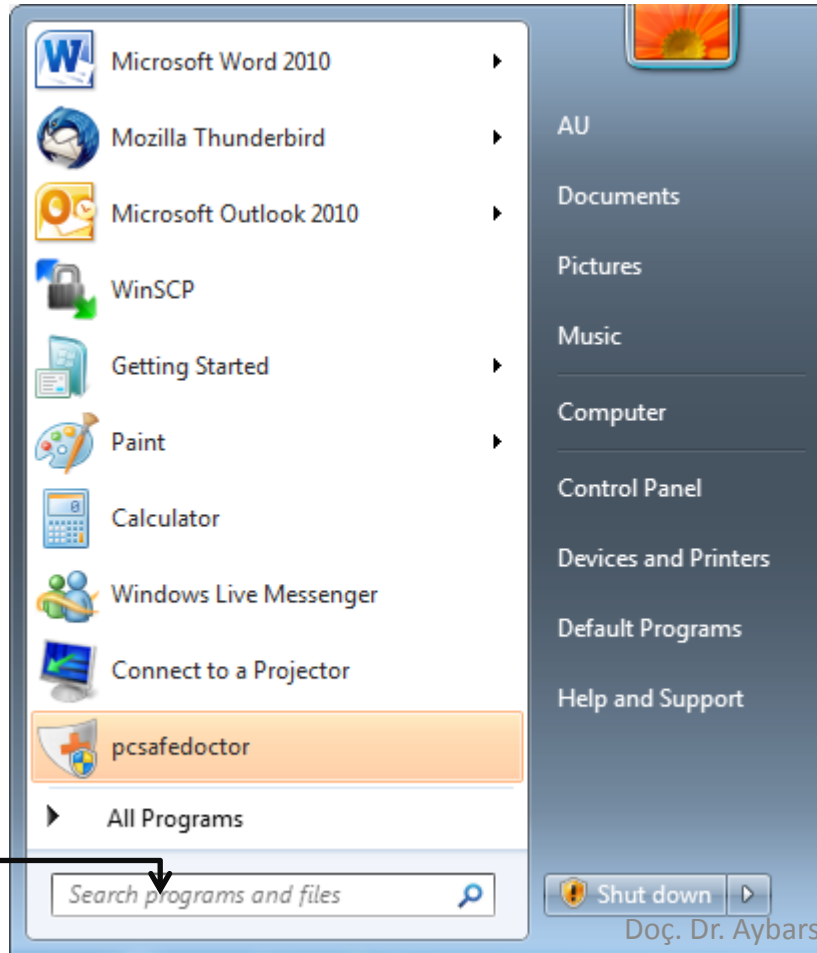


Örnek 2 : File -> New -> Project ile yeni bir proje açılabilir. Bilinen bir klasöre anlamlı bir isimler kaydetmek yararlıdır.



Programın Yeri :

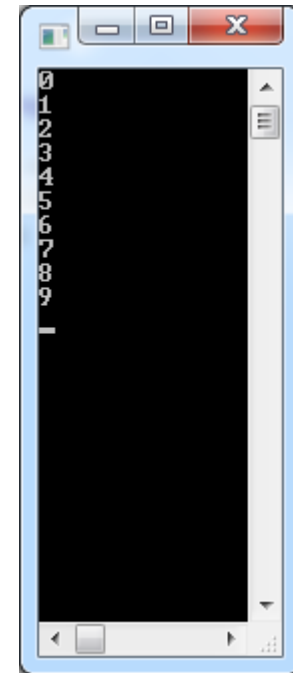
cmd ile komut satırı açılıp programa ulaşılabilir.



İlk 10 sayıyı yazdıran program

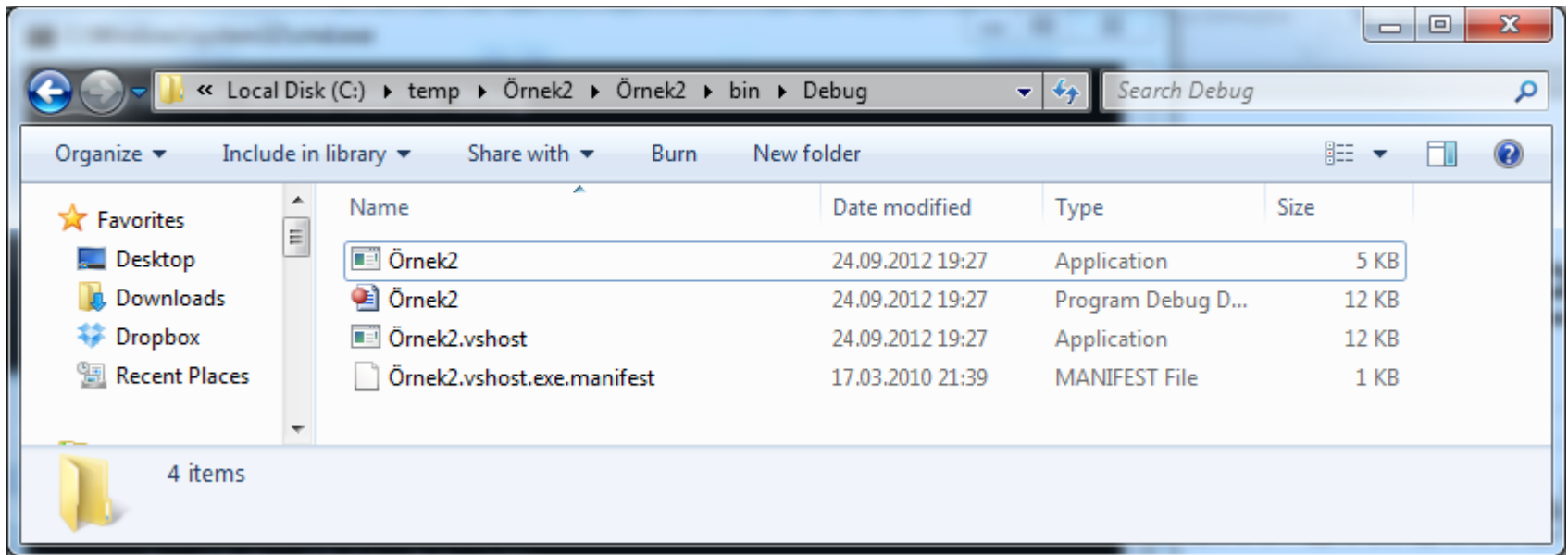
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Örnek2
{
    class Program
    {
        static void Main(string[] args)
        {
            for (int i = 0; i < 10; ++i)
                Console.WriteLine(i);
            Console.ReadKey();
        }
    }
}
```

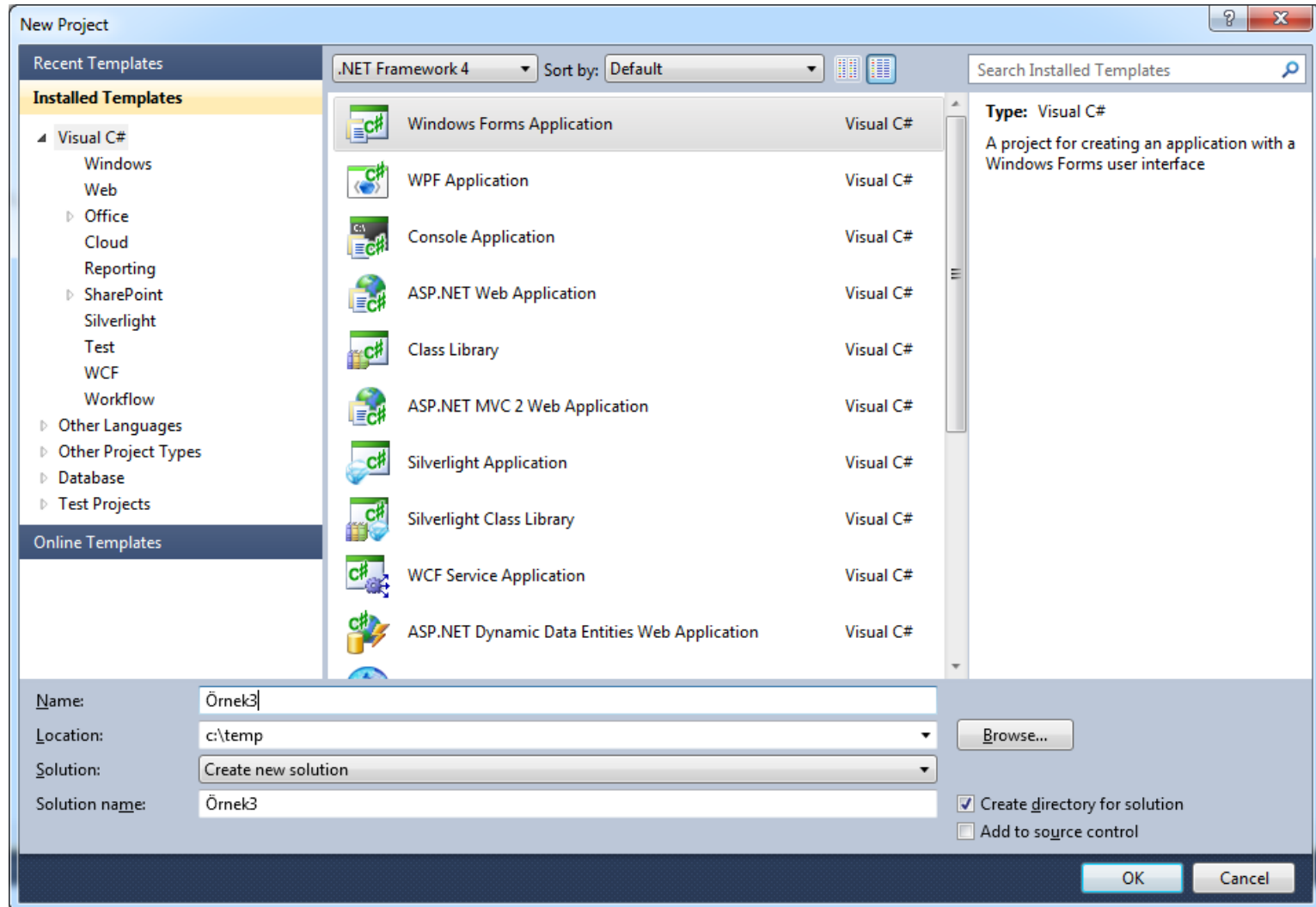


Örnek2.exe

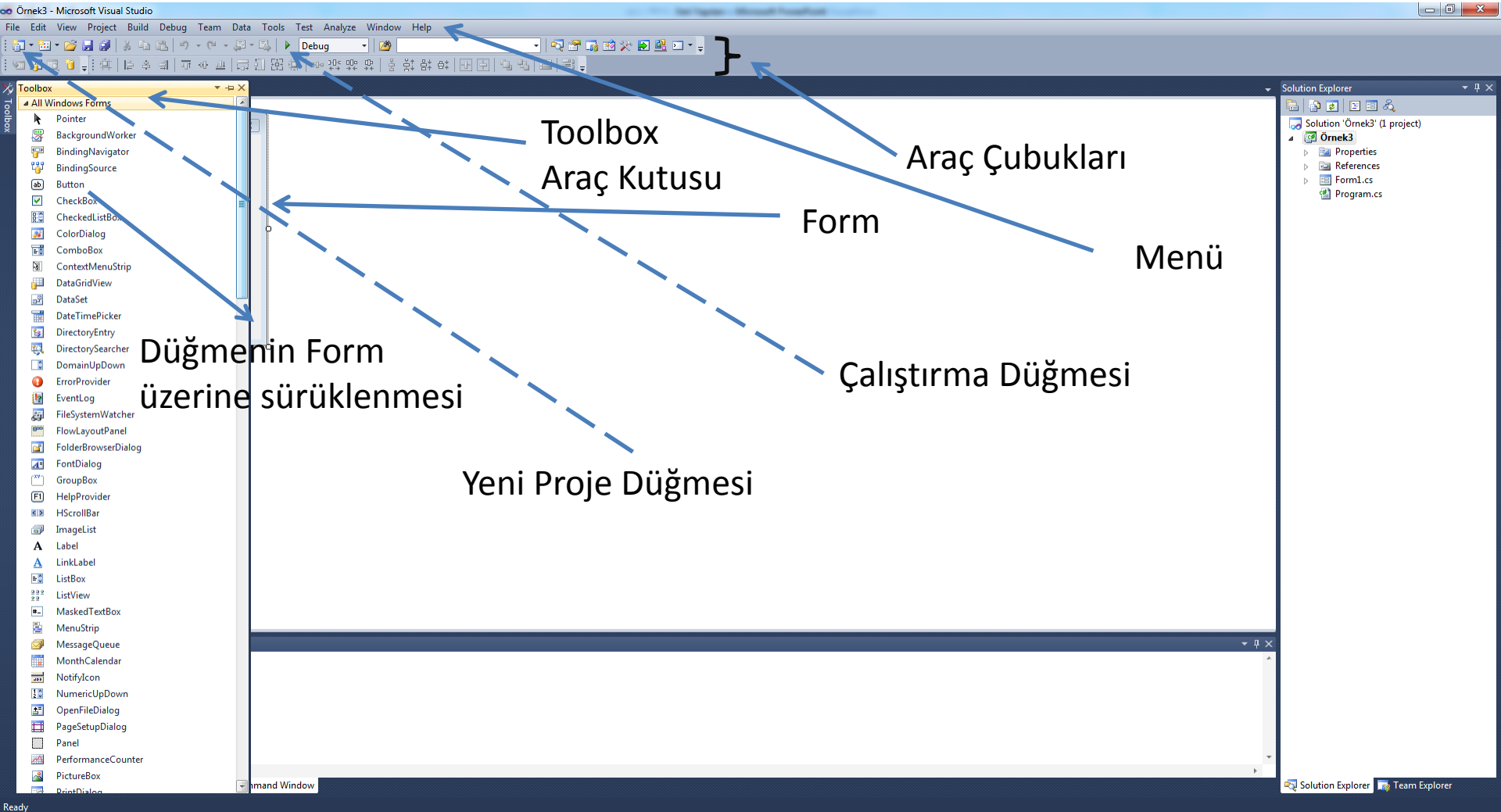
Program çalıştırıldıktan sonra Örnek2.exe oluşur. Daha sonra ilgili uygulama, Visual Studio dışından (veya başka bilgisayara taşınarak) da işletilebilir.



Örnek 3: Form Uygulaması



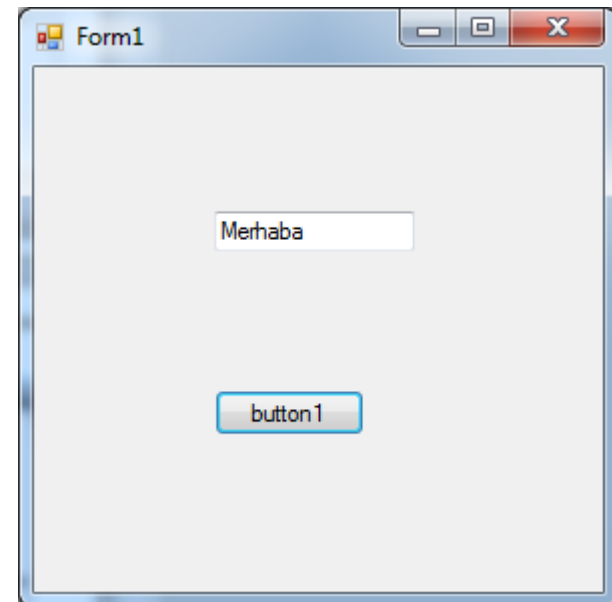
Toolbox içerisinde Button ve Textbox kontrolleri sürüklenerek Form üzerine bırakılır.



Form Uygulamasının Yazılması ve İşletimi

- Button1'e çift tıklanır.
- `textBox1.Text = "Merhaba";` yazılır.
- Program çalıştırılır.

- Form ekrana çıkar.
- Düğmeye basıldığında,
metin kutusuna **Merhaba**
yazar.



Ekranlar Arasında Geçiş

- View - Designer ve View – Code ile Tasarım ekranı ve kod arasında geçiş yapılabilir.

C# VERİ TİPLERİ

Veri Tipi Anlamı

int	tamsayı (32 bit) (-2,147,483,648..2,147,483,647)
long	uzun tamsayı (64 bit)
short	kısa tamsayı (16 bit) (-32768..32767)
float	kayan noktalı sayı (kns) (32 bit) (1,5E-45..3,4E+38)
double	çift duyarlıklı kns (64 bit) (5E-324..1.7E+308)
byte	8 bit işaretli tamsayı (8 bit) (0..255)
bool	true/false değerleri
char	karakter (16 bit)
uint	işaretsiz tamsayı (32 bit) >4 milyar
ulong	(64 bit) (0..18,446,744,073,709,551,615)
ushort	(16 bit) (0..65535)
decimal	(128 bit) (1E-28..7,9E+28) (hatasız)

SINIF ADI	KısaAdı	Tanımı
<i>System.Object</i>	<i>object</i>	Base class for all CTS types
<i>System.String</i>	<i>string</i>	String
<i>System.SByte</i>	<i>sbyte</i>	Signed 8-bit byte
<i>System.Byte</i>	<i>byte</i>	Unsigned 8-bit byte
<i>System.Int16</i>	<i>short</i>	Signed 16-bit value
<i>System.UInt16</i>	<i>ushort</i>	Unsigned 16-bit value
<i>System.Int32</i>	<i>int</i>	Signed 32-bit value
<i>System.UInt32</i>	<i>uint</i>	Unsigned 32-bit value
<i>System.Int64</i>	<i>long</i>	Signed 64-bit value
<i>System.UInt64</i>	<i>ulong</i>	Unsigned 64-bit value
<i>System.Char</i>	<i>char</i>	16-bit Unicode character
<i>System.Single</i>	<i>float</i>	IEEE 32-bit float
<i>System.Double</i>	<i>double</i>	IEEE 64-bit float
<i>System.Boolean</i>	<i>bool</i>	Boolean value (<i>true/false</i>)
<i>System.Decimal</i>	<i>decimal</i>	128-bit data type exact to 28 or 29 digits—mainly used for financial applications where a great degree of accuracy is required

Predefined Types

Value Types

- ◆ All are predefined structs

Signed	sbyte, short, int, long
Unsigned	byte, ushort, uint, ulong
Character	char
Floating point	float, double, decimal
Logical	bool

Predefined Types

Integral Types

C# Type	System Type	Size (bytes)	Signed?
sbyte	System.Sbyte	1	Yes
short	System.Int16	2	Yes
int	System.Int32	4	Yes
long	System.Int64	8	Yes
byte	System.Byte	1	No
ushort	System.UInt16	2	No
uint	System.UInt32	4	No
ulong	System.UInt64	8	No

DEĞİŞKENLER

Tip değişken ismi

```
int sayi1; // int veri tipidir.
```

```
/* sayi1 değişkeninin tamsayı tipinde olduğunu  
belirtir */
```

```
.....
```

```
sayi1=5;
```

```
float sayi=5.7f;
```

VERİ TİPİ DÖNÜŞÜMLERİ

```
int i=10;
```

```
float f;
```

```
f=i;
```

```
double sayi;
```

```
int karekok = (int) Math.Sqrt(sayi);
```

YAZDIRMA KOMUTU :

Console.WriteLine

```
Console.WriteLine("Not Ortalaması = " + ort);
```

```
Console.WriteLine("Şubat {0} veya {1} gündür", 28,29);
```

```
Console.WriteLine("Sayı\tKaresi");
```

```
Console.WriteLine("{0}\t{1}",5,5*5);
```

```
Console.WriteLine("{0,8}{1,10}",7,7*7);
```

```
Console.WriteLine("10/3 = {0:###}",10.0/3.0);
```

OPERATÖRLER - I

Hesaplamalarda kullanılan operatörler :

Aritmetik : +, -, *, /, % (Mod, kalan), ++, --

Mantıksal : &&, ||, !, &, |,

İlişkisel :

== (eşittir)

!= (eşit değildir)

>, <, >=, <=

OPERATÖRLER - II

Atama Operatörü : değişken = deyim;

```
int x = 5;
```

```
double sayi = -3.5;
```

```
int a, b, c;
```

```
a=b=c=100; // Atama zinciri
```

Bileşik Atamalar : +=, -=, *=, /=,

```
x-=10; // x = x - 10 ile eşdeğer
```

YAPISAL PROGRAMLAMA

Yapısal Programlamada üç tür denetim yeterlidir:

- Sıra (Sequence)
- Seçim (Selection)
- Tekrar (Repetition)

Sıralı işletim ?

PROGRAM DENETİM YAPILARI

- SEÇİM YAPILARI
 - if
 - if/else
 - switch
- TEKRAR YAPILARI (Döngüler)
 - while
 - do/while
 - for
 - foreach

SEÇİM YAPILARI : IF

if (koşul) ifade

```
if(notu>=60) Console.WriteLine("Geçti");
```

Koşul : bool veri tipindedir. true veya false olabilir.

İfade bloğu :

```
if(notu>=60)  
{ Console.WriteLine("Geçti"); sayac++; }
```

SEÇİM YAPILARI : IF/ELSE

if (koşul) ifade;

else ifade;

if(notu>=60)

 Console.WriteLine("Geçti");

else

 Console.WriteLine("Kaldı");

KÜMELENMİŞ (İÇİÇE) IF'LER (NESTED IF)

Verilen bir sayının işaretini (negatif, pozitif veya 0) bulan C# kod parçası :

```
if (i==0) Console.WriteLine("İşaretsiz");  
else  
    if(i<0) Console.WriteLine("Negatif");  
    else Console.WriteLine("Pozitif");
```

CONDITIONAL OPERATOR (?:)

```
Console.WriteLine(notu>=60?"Geçti":"Kaldı");
```

Eşdeğer ifade :

```
if (notu>=60)  
    Console.WriteLine("Geçti");  
else  
    Console.WriteLine("Kaldı");
```

IF-ELSE-IF MERDİVENİ (CASCADING IF)

if(koşul) ifade

else if (koşul) ifade

else if (koşul) ifade

.....

else ifade;

if (notu>=90)

str = "A";

else if (notu>=80)

str = "B";

else if (notu>=70)

str = "C";

else if (notu>=60)

str = "D";

else str = "E";

Console.WriteLine(str);

SEÇİM YAPILARI : SWITCH

```
switch (deyim) {  
    case sabit1:  
        ifade1;  
        break;  
    case sabit2:  
        ifade2; break;  
    ....  
    default :  
        ifade;  
        break;  
}
```

```
switch(sayi) {  
    case 0 :  
        Console.WriteLine(" Sayı 0");  
        break;  
    case 1 :  
        Console.WriteLine(" Sayı 1");  
        break;  
    case 2 :  
        Console.WriteLine(" Sayı 2");  
        break;  
    default : Console.WriteLine("X");  
        break;  
}
```

DÖNGÜLER : FOR

- `for(int sayac=0; sayac<10; ++sayac)`
`for(başlangıç; devam koşulu; kontrol değişkeni değişimi)`

```
for(int sayac=0; sayac<10; ++sayac)
{
    ifadeler;
}
```

foreach döngüsü için diziler bölümüne bakınız.

DÖNGÜLER : WHILE ve DO-WHILE

while(koşul) ifade

```
do {  
    ifadeler  
} while (koşul);
```

```
int sayac=1, toplam=0;  
while(sayac<10) { toplam+=sayac; sayac++; };
```


DÖNGÜDEN ÇIKMAK : BREAK

```
for(int x=1; x<10; ++x)
{
    toplam+=x;
    if (x==5) break;
}
```

```
Console.WriteLine("1 ile 5 arasındaki sayıların  
    toplamı = "+toplam);
```

BREAK sadece en içteki döngüden çıkar.

DÖNGÜDE ERKEN TEKRAR : CONTINUE

1 ile 100 arasındaki tek sayıları yazan program :

```
for(int i=0; i<100; ++i)
{
    if ( (i%2)!=1) continue;
    Console.WriteLine(i);
}
```

DİZİLER

Dizi (array), aynı tipteki değişkenler topluluğudur.

```
int[] arr = new int[10];
```

```
int[] dizi = {5,7,12,2,9,8,14,21,-6,5};
```

0	1	2	3	4	5	6	7	8	9
5	7	12	2	9	8	14	21	-6	5

```
double[] sayilar; sayilar = new double[5];
```

-3.4	12.5	27.0	1.1	25.33
------	------	------	-----	-------

DİZİ KULLANIM ÖRNEKLERİ

```
dizi[2]++;
```

```
dizi[3]=dizi[1]+dizi[2];
```

```
Console.WriteLine(dizi[5]);
```

Dizi elemanlarının toplamını bulduran programı yazınız : dizi.Length kullanınız!

FOREACH

foreach döngüsü, özellikle bir koleksiyonun tüm elemanları üzerinde işlemler yapılacaksa yararlıdır :

```
int toplam=0;
```

```
foreach(int i in dizi) toplam+=i;
```

foreach döngüsü “break” kullanılarak daha erken de bitirilebilir.

İKİ BOYUTLU DİZİLER (MATRİSLER)

- $M \times N$
- M satır, N sütun

		tablo		
		0	1	2
M=4	0	2	15	9
	1	21	33	8
	2	3	17	61
	3	89	3	5
		N=3		

- Oluşturulması : `int[,] tablo = new int[4,3];`
- Kullanımı : `tablo[0,2]`

ÇOK BOYUTLU DİZİLER

- Tip [,...,,] isim = new tip[büyüklik1,..., büyüklikN]

Örnekler

- double[,,,] mdizi = new double[4,10,2]; // boyutu 3
- float[,,,,] dizi4d = new float[5,5,5,5]; // boyutu 4

DÜZENSİZ DİZİLER (jagged array)

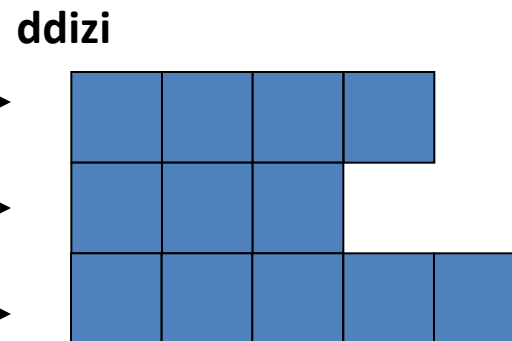
- Her biri farklı uzunluktaki dizilerin oluşturduğu dizidir.
- İki boyutlu dizilerde dikdörtgensel olmayan matrisler elde etmek için kullanılabilir : Her satırı farklı uzunlukta olabilen matris.

```
int[][] ddizi = new int[3][];
```

```
ddizi[0] = new int[4];
```

```
ddizi[1] = new int[3];
```

```
ddizi[2] = new int[5];
```



STRING'LER ve KARAKTER DİZİLERİ

- Karakter dizisi : `char[] harfdizi = {'T','e','s','t'};`

- String : Karakter dizisinden farklıdır.

```
string str1 = "Merhaba";
```

```
string str2 = new string(harfdizi);
```