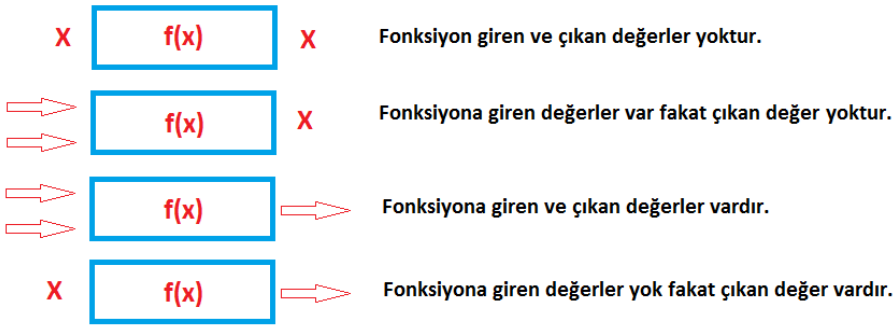
BİLGİSAYAR PROGRAMLAMA-II- 4.ders

FONKSİYONLAR

Program yazarken fonksiyon kullanmanın bir çok faydası vardır. Bu faydası aşağıdaki şekilde özetlenebilir.

a) Program fonksiyonlar vasıtasıyla daha küçük parçalara bölündüğü için programın anlaşılırlığı artar.
b) Fonksiyona yazılan komutlar programın değişik yerlerinde tekrar yazılmak zorunda kalınmadığı için program daha az kodla oluşturulmuş olur.
c) Fonksiyona giren ve çıkan değerler kontrol altında tutulduğu için, programda oluşabilecek hataların önüne geçilmiş olur ve hataların tesbiti kolaylaşmış olur.
d) Programın akış diyağramı ve mantıksal yapısı fonksiyon kullanımı ile daha kolay oluşturulur.
e) Nesne tabanlı programlama teknikleri kullanılırken, fonksiyon yapıları kullanılacağı için alt yapı sağlanmış olur.

Fonksiyonları aynı matematikte fonksiyonlar gibi düşünebiliriz. Matematik de bir fonksiyona bir çok değer girer, fonksiyon içerisinde bazı işlemler yapılır ve sonuç olarak da bir tane değer üretilir. $y= 3x^2 + z$ şeklinde verilen bir fonksiyonda x ve z değerleri giren değerler, y değeri ise çıkan değerdir. x ve z değerleri bazı işlemlerden geçilir ve tek bir çıkış olan y değeri oluşturulmuş olur. Aynı şekilde programlamadaki fonksiyon ifadeleri ise giriş ve çıkış değerlerine bağlı olarak 4 şekilde gruplandırılabilir.



Fonksiyonun genel formatı şu şekildedir. Fonksiyona giren değişken değerleri parantezin içinde tanımlanır. Dışarıdan bu fonksiyona değerler gönderilirken, girişteki tanımlanan değişkenin tipleri ile aynı tipte olmalıdır. Fonksiyondan geri değer döndürülecekse fonksiyon içinde return kelimesi ile geri dönen değer gösterilir. Bu değer fonksiyonun adının başında bulunan tip ile aynı olmalıdır. Eğer bir fonksiyon geri değer göndermeyecekse fonksiyonun adının başına void ifadesi eklenmeidir. Bu ifadeden önce public ifadesi kullanılırsa fonksiyon programın her tarafından çağrılarak kullanılabilir.

```
public double FonksiyonunAdi(double GirenDegisken1, double GirenDegisken2)
{
      double FonksiyonIcindeKullanilanYerelDegisken= 0;
      .......
      İşlemler
      .......
      return GeriDonenDeger;
}
```

Fonksiyon Oluşturma

Fonksiyonlar dört farklı tipte oluşturulur.

A- Hiç bir dış değer almayan ve geri değer göndermeyen fonksiyonlar. Sadece işlem yapar	B- Dışarıdan değer almaz fakat geri değer gönderir
`public Void FonksiyonAdi()` `{`	`public geridönüştipi FonkAdi()` `{`

Yerel degişkenler *İşlemler* *}*	*Yerel degişkenler* *İşlemler* *Return geridönendeğer* *}*

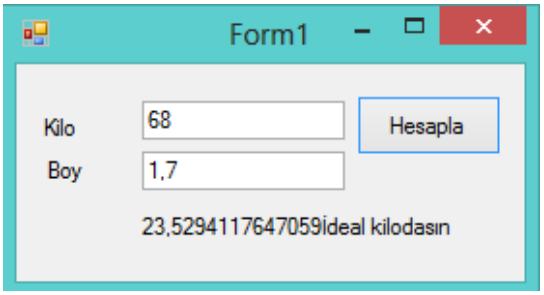
private void button1_Click(object sender, EventArgs e)
{
Hesapla();
}
 	
public void Hesapla()
{
int A=3, B=2, C;
C = A + B;
MessageBox.Show("Sonuc="+C);
}
	private void button1_Click(object sender, EventArgs e)
	{
	MessageBox.Show(Hesapla().ToString());
	}
	
	public int Hesapla()
	{
	int A=3, B=2, C;
	C = A + B;
	return C;
	}

C- Dışarıdan değer alır fakat dışarı değer göndermez	D- Dışarıdan hem değer alır hemde değer göderir.
public Void FonksiyonAdi(tip Degişken1, tip Degişken2) *{* *Yerel degişkenler* *İşlemler* *}*	*public geridönüştipi FonkAdi(tip Degişken1, tip Degişken2)* *{* *Yerel degişkenler* *İşlemler* *Return geridönendeğer* *}*

private void button1_Click(object sender, EventArgs e)
{
int X = 3, Y = 2;
Hesapla(X,Y);
}
 	
public void Hesapla(int A, int B)
{
int C;
C = A + B;
MessageBox.Show(C.ToString());
}
	private void button1_Click(object sender, EventArgs e)
	{
	int X = 3, Y = 2;
	MessageBox.Show(Hesapla(X, Y).ToString());
	}
	
	public int Hesapla(int A, int B)
	{
	int C;
	C = A + B;
	return C;
	}

Örnek

İdeal kilo hesabı yapan bir program yazın. Bu program üzerinde 4 tip fonksiyon kullanımını gösterin. İdel Kilo = Kilo / Boy^2 dir. Örnek : 68 kg/ 1.70 m = 23.52 çıkar. Bu sayı 18 den küçük ise kişi zayıf, 18-25 arasında ise ideal kiloda, 25-30 arasında ise hafif şişman ve 30 üzerinde çıkarsa obozite demektir.



1.fonksiyon kullanımı

```
private void button1_Click(object sender, EventArgs e)
{
            Hesapla();
}

public void Hesapla()
{
      double Boy=0, Kilo=0,Katsayi = 0;

      Boy=Convert.ToDouble (textBox2.Text)  ;
      Kilo = Convert.ToDouble (textBox1.Text);

      Katsayi = Kilo / (Boy * Boy);
      if (Katsayi < 18)
            label3.Text =Katsayi.ToString() + "Zayıfsın";
      else if(Katsayi >= 18 && Katsayi <= 25)
            label3.Text = Katsayi.ToString() + "İdeal kilodasın";
      else if (Katsayi > 25 && Katsayi < 30)
            label3.Text = Katsayi.ToString() + "Hafif Şişmansın";
      else if (Katsayi > 30)
      label3.Text = Katsayi.ToString() + "Şişmansın";
}
```

2. Tip Fonksiyon Kullanımı

```
private void button1_Click(object sender, EventArgs e)
{
      double Boy = 0, Kilo = 0, Katsayi = 0;

      Boy = Convert.ToDouble(textBox2.Text);
      Kilo = Convert.ToDouble(textBox1.Text);

      Hesapla(Boy,Kilo);
}

public void Hesapla(double Boy_degisken, double Kilo_degisken)
{
      double  Katsayi = 0;
      Katsayi = Kilo_degisken / (Boy_degisken * Boy_degisken);

      if (Katsayi < 18)
            label3.Text =Katsayi.ToString() + "Zayıfsın";
      else if(Katsayi >= 18 && Katsayi <= 25)
```

```
            label3.Text = Katsayi.ToString() + "İdeal kilodasın";
      else if (Katsayi > 25 && Katsayi < 30)
            label3.Text = Katsayi.ToString() + "Hafif Şişmansın";
      else if (Katsayi > 30)
            label3.Text = Katsayi.ToString() + "Şişmansın";
}
```

3. Tip Fonksiyon Kullanımı

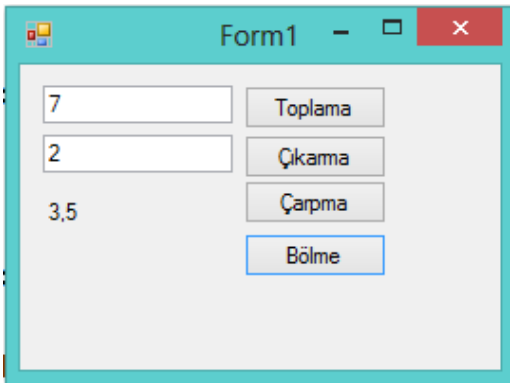
```
private void button1_Click(object sender, EventArgs e)
{
      double Boy = 0, Kilo = 0, Katsayi = 0;

      Boy = Convert.ToDouble(textBox2.Text);
      Kilo = Convert.ToDouble(textBox1.Text);

      double Katsayi_Degeri= Hesapla(Boy,Kilo);

      if (Katsayi_Degeri < 18)
            label3.Text = Katsayi_Degeri.ToString() + "Zayıfsın";
      else if (Katsayi_Degeri >= 18 && Katsayi_Degeri <= 25)
            label3.Text = Katsayi_Degeri.ToString() + "İdeal kilodasın";
      else if (Katsayi_Degeri > 25 && Katsayi_Degeri < 30)
            label3.Text = Katsayi_Degeri.ToString() + "Hafif Şişmansın";
      else if (Katsayi_Degeri > 30)
            label3.Text = Katsayi_Degeri.ToString() + "Şişmansın";
}

public double Hesapla(double Boy_degisken, double Kilo_degisken)
{
      double  Katsayi = 0;
      Katsayi = Kilo_degisken / (Boy_degisken * Boy_degisken);

      return Katsayi;
}
```

4. Tip Fonksiyon Kullanımı

```
private void button1_Click(object sender, EventArgs e)
{
      double Katsayi_Degeri = Hesapla();

      if (Katsayi_Degeri < 18)
            label3.Text = Katsayi_Degeri.ToString() + "Zayıfsın";
      else if (Katsayi_Degeri >= 18 && Katsayi_Degeri <= 25)
            label3.Text = Katsayi_Degeri.ToString() + "İdeal kilodasın";
      else if (Katsayi_Degeri > 25 && Katsayi_Degeri < 30)
            label3.Text = Katsayi_Degeri.ToString() + "Hafif Şişmansın";
      else if (Katsayi_Degeri > 30)
            label3.Text = Katsayi_Degeri.ToString() + "Şişmansın";
}

public double Hesapla()
{
      double Boy = 0, Kilo = 0, Katsayi = 0;

      Boy = Convert.ToDouble(textBox2.Text);
      Kilo = Convert.ToDouble(textBox1.Text);

      Katsayi = Kilo / (Boy * Boy);

      return Katsayi;
```

```
}
```

Örnek

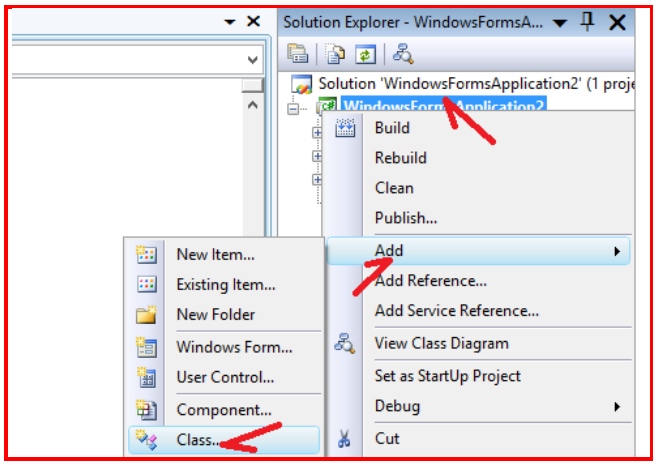


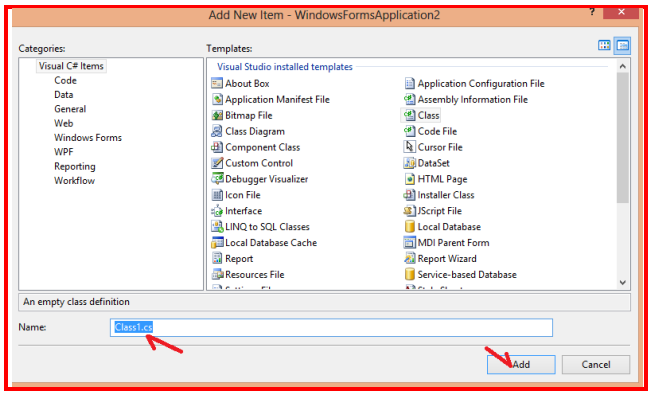

```
private void button1_Click(object sender, EventArgs e)
{
      topla();
}

private void button2_Click(object sender, EventArgs e)
{
      int Sayi1 = Convert.ToInt32(textBox1.Text);
      int Sayi2 = Convert.ToInt32(textBox2.Text);

      cikar(Sayi1, Sayi2);
}

private void button3_Click(object sender, EventArgs e)
{
      int Sayi1 = Convert.ToInt32(textBox1.Text);
      int Sayi2 = Convert.ToInt32(textBox2.Text);
      label1.Text = carpma(Sayi1, Sayi2).ToString();

}

private void button4_Click(object sender, EventArgs e)
{
      label1.Text = bolme().ToString();

}

//FONKSİYONLAR
public void topla()
{
      int Sayi1 = Convert.ToInt32(textBox1.Text);
      int Sayi2 = Convert.ToInt32(textBox2.Text);

      int sonuc = Sayi1 + Sayi2;
      label1.Text = sonuc.ToString();
}

public void cikar(int a, int b)
{
      int sonuc = a - b;
      label1.Text = sonuc.ToString();
}
```

```
public int carpma(int a, int b)
{
      return a * b;
}

public double bolme()
{
      int Sayi1 = Convert.ToInt32(textBox1.Text);
      int Sayi2 = Convert.ToInt32(textBox2.Text);

      double sonuc =(double) Sayi1 / (double) Sayi2;
      return  sonuc;
}
```

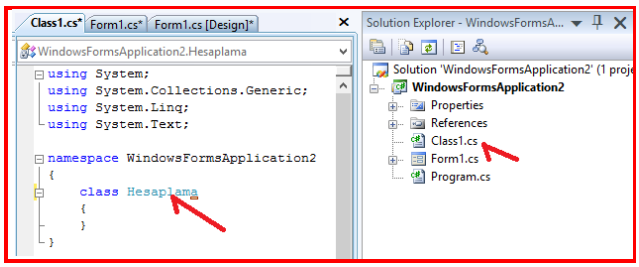
Sınıf (Nesne) Oluşturma

Projemizin içinde tekrar eden kodları bir nesne şeklinde (sınıf=class) şeklinde oluşturarak tekrar kullanabiliriz. Sınıf yapısını projenin her tarafından çağırıp kullanıma açabiliriz. Bunun için öncelikle solution penceresinden projemize sağ tuşa tıklarsak Add>Class yolunu takip edersek sınıf oluşturmak için pencere açılacaktır. Bu pencerede sınıf kodları için isim belirleyebiliriz.





Burada önemli olan Class1.cs isminden daha çok içerisindeki sınıf tanımlayan fonksiyonun adıdır. Bu ad bizim kodlarımızda kullanacağımız isimdir.



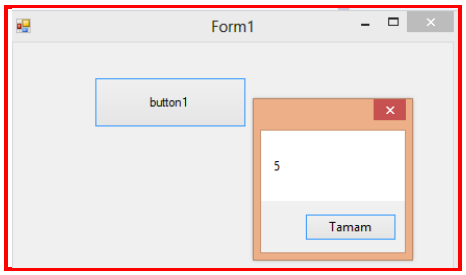
Şimdi bu hesaplama sınıfı içerisine iki tane fonksiyon (metod) yazalım.

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

namespace WindowsFormsApplication2
{
    class Hesaplama
    {
        public double Topla(double A, double B)
        {
            return A + B;
```

```
        }
        public double Cikar(double A, double B)
        {
            return A - B;
        }
    }
}
```

Sınıfı projemizde kullanırken şu şekilde tanımlarız.

```
private void button1_Click(object sender, EventArgs e)
{
Hesaplama Islem =new Hesaplama();
MessageBox.Show(Islem.Topla(2,3).ToString());
}
```



HAZIR FONKSİYONLAR

STRING FONKSİYONLARI (KÜTÜPHANESİ)

Çoğu web sitesinde olduğu gibi kullanıcıdan kullanıcı adını ve şifresini aldıktan sonra bu değerleri veri tabanından elde edilen değerlerle karşılaştırmak bir string işlemidir veyahut kullanıcı adı ve şifresini, yazacağınız sorgu cümlesinin(**SQL**) uygun yerine yerleştirmek yine bir string işlemidir ve çok önemlidir.

Birileri çok önceden bir uygulama yapmış ve uygulama verilerinin bir kısmını, bir **text** dökümanında veya bir **word** dökümanında veya herhangi bir yerde text biçiminde tutmuş.Sizde yeni bir yazılım gerçekleştireceksiniz fakat yazılımıza eski bilgileri de aktarmayı ihmal etmeyeceksiniz.(Her ne kadar eski program bilgileri veri tabanında tutuluyor olsada bazı bilgilerin bir **text** dosyasında veya bir **excel** dosyasında tutulması muhtemeldir ve karşılaşılan bir şeydir).Sizde dosyalara bir bakıyorsunuz ki adamlar kendilerine uygun bir biçimde verileri dosyalara kaydetmişler.Bu durumda size düşen bu text dosyalarından aldığınız verileri ayrıştırıp anlamlı verilere dönüştürmek.Yine bu durumda da yapacağınız işlemler genel itibariyle string işlemleridir.

Sitemizin üyelerine belli aralıklarla mail gönderdiğimizi düşünelim.Tabi mail gönderirken düz yazı değil,bir çerçeve oluşturup,çerçevenin sağ üst kısmına logomuzu koyup ,içeriğinin belli yerlerine veri tabanından çektiğimiz verileri yerleştirdip güzel görünümlü bir yazıyı göndeririz.

Örneğin “Sayın **Hüseyin Akkuş**,Şifreniz:**265901** .....” gibi uzayıp giden bir mail içeriğini hazırlamak için koyu olarak yazılmış verilerin veri tabanından çekilip diğer sabit string ifadelerle birleştirilmesi gerekmektedir.Bu durumda ne tür bir string işlemi gereklidir acaba?

Velhasıl kelam,bu saydıklarımda ve sayılacak daha bir ton şeyde string işlemleri vazgeçilmezdir.

String.Compare() – String ifadeleri karşılaştırmak

```
string str1 = "Dürdane";

string str2 = "Fikriye";


int result = String.Compare(str1, str2);
```

```
if (result < 0)
   Console.WriteLine("{0} < {1}", str1, str2);
else if (result > 0)
   Console.WriteLine("{0} > {1}", str1, str2);
else
   Console.WriteLine("{0} = {1}", str1, str2);
```

Yukarıdaki kod parçasında 2 string değer, **String** sınıfının **Compare()** metodu kullanılarak karşılaştırılmaktadır.

String.Compare() metodun, 1. parametre,2.parametreden küçük ise(alfabetik sıralamaya göre) negatif,büyük ise pozitif eşit ise 0 değerini döndürmektedir.

2 string'in eşit olup olmadığını karşılaştırmak için benzer şekilde

```
if (str1 == str2)
   Console.WriteLine("{0} = {1}", str1, str2);
```

karşılaştırmasını kullanabiliriz,fakat hangisinin hangisinden küçük olduğunu bulmak için **<** ve **>** işaretlerini bu karşılaştırmada kullanamayız.

Aşağıdaki kod parçasına bir bakalım;

```
string str1 = "Dürdane";
string str2 = "dürdane";

int result = String.Compare(str1, str2);

if (result < 0)
   Console.WriteLine("{0} < {1}", str1, str2);
else if (result > 0)
   Console.WriteLine("{0} > {1}", str1, str2);
else
   Console.WriteLine("{0} = {1}", str1, str2);
```

2 string ifade de aynıdır fakat 1. String ifadenin ilk harfi büyük olduğundan dolayı,1.string ikincisinden büyük olacak ve Compare metodu 0'dan büyük bir değer döndürecektir.Bazen öyle durumlar olur ki bu 2 string ifadenin eşit olmasını isteyebiliriz yani büyük küçük harf duyarlı olmamasını isteyebiliriz.Bu durumda Compare metodunun 3 parametresi işimizi görecektir.Zira eğer bu parametre **true** olur ise büyük küçük harf olup olmadığı dikkate alınmayacaktır.Aynı kod parçasını şe şekilde işletelim;

```
string str1 = "Dürdane";
string str2 = "dürdane";

int result = String.Compare(str1, str2, true); // Buraya DİKKAT!...
```

```
if (result < 0)
   Console.WriteLine("{0} < {1}", str1, str2);
else if (result > 0)
   Console.WriteLine("{0} > {1}", str1, str2);
else
   Console.WriteLine("{0} = {1}", str1, str2);
```

Bu durumda ekran görüntüsü (Dürdane = dürdane) olacaktır.

String.Format() – String biçimlendirmek

Bazen string ifadelerimizi formatlamak(biçimlendirmek) isteyebiliriz.Örneğin elimizde bir tarih varsa bunu anlamlı bir şekilde yazdırmak isteyebiliriz.Veya bir kordinat verilerini anlamlı bir biçimde yazdırmak isteyebiliriz.Şimdi aşağıdaki kod parçacıklarına bakalım;

```
int x = 3,
   y = 4;

// {0} yazan yere 3,
// {1} yazan yere 4 değeri gelecek ve
// coord değeri "3,4" olacaktır.
string coord = String.Format("{0},{1}", x, y);

// 2 string ifade ("Koordinat" ve "3,4") toplanıyor(birleştiriliyor).
Console.WriteLine("Koordinat:" + coord);

DateTime date = new DateTime(2008, 8, 23);
string dateText = String.Format("{0:d}", date);
Console.WriteLine(dateText);
dateText = String.Format("{0:D}", date);
Console.WriteLine(dateText);
```

İlk örnekte gerekli açıklama yapılmıştır.2 örnek için ise öncelikle **DateTime** türünde bir değişken tanımlanmıştır.Ardından **String** sınıfının **Format** fonksiyonu kullanılarak bu tarihe 2 çeşit biçim verilmiştir.İsterseniz yukarıdaki kod parçasının ekran çıktısına bir bakalım;

```
23.08.2008
23 Ağustos 2008 Cumartesi
```

Bu tarz biçimlendirmelerin bazılarını listeyelim;

{0:d}

23.08.2008

{0:D}

23 Ağustos 2008 Cumartesi

{0:f}

23 Ağustos 2008 Cumartesi 13:20

{0:F}

23 Ağustos 2008 Cumartesi 13:20:05

{0:t}

13:20

{0:T}

13:20:05

{0:y}

Ağustos 2008

Bir string ifadenin sol veya sağ yanına boşluk karakteri doldurmak istiyor iseniz yine String.Format() metodunu kullanabilirsiniz.Aşağıdaki örneğe göz atalım;

```
string str;


str = String.Format("-{0,15}-", "ASP.NET");
Console.WriteLine(str);


str = String.Format("-{0,-15}-", "ASP.NET");
Console.WriteLine(str);
```

Bu durumda birincisinde ASP.NET yazısının sol tarafına 15-7=8 tane boşluk karakteri eklenecek, ikincisinde ise sağ tarafına eklenecektir.

Şimdi ekran çıktısına bakalım;

```
-       ASP.NET-
-ASP.NET       -
```

Contains() - EndsWith() – StartsWith()

Contains metodu,bir string ifadenin diğer bir string ifade de içinde geçip geçmediğini bulur,geçiyor ise **true** geçmiyor ise **false** döndürür.

Örneğin,

```
string str1 = "Dürdane";
if(str1.Contains("dane") == true)
```

gibi bir karşılaştırma doğrudur.”Dürdane” kelimesi, “dane” kelimesini içermektedir ve yukarıdaki **if** koşulu true değerini döndürecektir.

Benzer şekilde **EndsWith()** ve **StartsWith()** metodlarıda aldıkları stringi,karşılaştırma yaptıkları stringin başında mı sonunda mı olduğu bilgisini döndürür.

```
string str1 = "Dürdane";
if (str1.StartsWith("Dür") == true)
{
   // "Dürdane" kelimesi "Dür" kelimesiyle başlamaktadır.
   // if bloğu işletilecektir.
}

if (str1.EndsWith("dane") == true)
{
   // "Dürdane" kelimesi "dane" kelimesiyle sonlanmaktadır.
   // if bloğu işletilecektir.
}
```

Split() – String’i dizi halinde parçalamak

```
string Metin =textBox1.Text ;

string[] DiziKelimeler = Metin.Split(' ');

foreach(string Kelime in DiziKelimeler)
{
       listBox1.Items.Add(Kelime);

}
```

Bir string içerisindeki kelimeleri bazı karakterleri kullanarak ayırmak istiyor isek bu durumda **Split()** metodunu kullanmamız gerekir**.”2,3”** koordinat bilgisinden **2** ve **3** sayılarını elde etmek istiyor isek Split() metodu harika metoddur.

Aşağıdaki kod parçacığında x,y,z değerleri string ifadeden parçalanıp elde edilmektedir.

```
// Koordinat bilgisi string olarak tutuluyor.
string coord = "2,3,5";

// Bu komut sonrasında 3 string ifadeden oluşan bir dizi elde edilecektir.
// xyz[0] = "2"
// xyz[1] = "3"
// xyz[2] = "5"
string[] xyz = coord.Split(',');
```

```
int x = int.Parse(xyz[0]);
int y = int.Parse(xyz[1]);
int z = int.Parse(xyz[2]);

// Ekran Çıktısı : 2,3,5
Console.WriteLine("{0},{1},{2}", x, y, z);
```

SubString() – String içindeki alt stringleri elde etmek

Bir string ifadenin içinde,4.karakterden başlayıp 10 karakter elde etmek istiyor isek bu durumda kullanacağımız metod **SubString()** metodudur.

Aşağıdaki kod parçalarını inceleyelim;

```
string text = "Visual Studio 2005";
Console.WriteLine(text.Substring(7,4));    // Ekran Çıktısı : "Stud"
Console.WriteLine(text.Substring(7));      // Ekran Çıktısı : "Studio 2005"
```

ToLower() – ToUpper() – ToLowerInvariant() - ToUpperInvariant()

Bir string ifadedeki bütün karakterleri küçük veya bütün karakterleri büyük yapmak istiyor iseniz bu fonksiyonlar işinizi görecektir.

Aşağıdaki kod parçasına ve ekran çıktısına bakalım;

```
string text = "Visual Studio 2005";
Console.WriteLine(text.ToLower());         // visual studio 2005
Console.WriteLine(text.ToLowerInvariant()); // visual studio 2005

Console.WriteLine(text.ToUpper());         // VİSUAL STUDİO 2005
Console.WriteLine(text.ToUpperInvariant()); // VISUAL STUDIO 2005
```

ToLower() ve ToUpper() metodları karakterleri olduğu gibi büyük veya küçük harfe çevirirken, ToLowerInvariant() ve ToUpperInvariant() metodları ise ilgili dile göre değişim göstermektedir. Uygulamayı gerçekleştirdiğim işletim sistemi ingilizce olduğundan dolayı küçük **'i'** karakterleri büyük harfe çevrildiğinde **'I'** haline dönüştürülüyor."**Invariant**" kullanılmayan metodlarda ise **'i'** harfleri olduğu gibi **'İ'** harfine dönüştürülüyor.

Trim() – TrimEnd() – TrimStart() – Boşlukları kaldıran fonksiyonlar

Trim metodları, string içindeki boşluklarla bir derdiniz var ise çok işinize yarayacaktır.

Aşağıdaki kod örneğini ve açıklamaları inceleyelim;

```
string text = "    Visual Studio 2005    ";

/*
```

```
* Trim()      : Text'in başındaki ve sonundaki boşlukları kaldırır
* TrimEnd()   : Text'in sonundaki boşlukları kaldırır.
* TrimStart() : Text'in başındaki boşlukları kaldırır.
*
*/
Console.WriteLine("-{0}-", text.Trim());       // -Visual Studio 2005-
Console.WriteLine("-{0}-", text.TrimEnd());    // -   Visual Studio 2005-
Console.WriteLine("-{0}-", text.TrimStart());  // -Visual Studio 2005   -
```

Replace() – Yer değiştirme fonksiyonu

Bir string içindeki bir değeri başka bir değerle değiştirmek istiyor iseniz Replace() metodunu kullanmanız gerekmektedir.

Aşağıdaki kod parçacığını inceleyelim;

```
string text = "Visual Studio 2005";

string text2 = text.Replace("sual", "SORU");
Console.WriteLine(text2);   // "Ekran Çıktısı : ViSORU Studio 2005"
```

StringBuilder() – String inşa eden sınıf

Birden fazla stringi birleştirmek istiyor iseniz **+** ile bunu yapabilirsiniz.(**str4 = str1+str2+str3**).Fakat bu yöntem performans açısından iyi değildir.Onun yerine **StringBuilder** sınıfını kullanmak yazılımınızı daha kaliteli hale getirecektir.

Aşağıdaki kod parçacığını inceleyelim;

```
StringBuilder builder = new StringBuilder();
// .Net dilleri dizi içinde tanımlanıyor
string[] diller = new string[] { "C#", "VB", "C++" };

builder.AppendLine(".Net Dilleri..");
builder.AppendLine();   // Boş bir satır ekleniyor.
for (int i = 0; i < diller.Length; i++)
{
   // Sırasıyla bütün diller yanyana ekleniyor.
   builder.Append(diller[i] + " ");
}
// 0.karakterden başlayarak "-->" ifadesi ekleniyor.
builder.Insert(0, "-->");
// Ekrana ToString() metodu ile yazdırılıyor.
Console.WriteLine(builder.ToString());
```

Makalemin en başında belirttiğim örneklerde birden fazla string'in birleştirilmesiyle ilgili örneklerin hepsi StringBuilder sınıfı kullanılarak yapılmalıdır.

Her ne kadar çözüm yolları çok basit olsa da incelediğimiz metodlar bir yazılımda sık sık kullanabileceğimiz metodlardır.Burada makalemi bitiyorum. Diğer makalelerimde görüşmek dileğiyle,hoşçakalın.

MATEMATİK (Math) FONKSİYONLARI (KÜTÜPHANESİ)

Normal şartlarda kütüphanesi eklenmiş olarak gelir.

Math.E; // e sayısını verir

Math.PI; // pi sayısını verir

Math.Sin(b); // b sayısının sin değerini alır

Math.Cos(b); // b sayısının Cos değerini alır

Math.Tan(b); // b sayısının Tan değerini alır

Math.Exp(b); // e^b demektir

Math.Pow(b,c); // b^c demektir

Math.Sqrt(b); // Karekök değerini alır daha fazla kök için $a^{(2/3)}$, Math.Pow(a,(2/3))

Math.Ceiling(b); // Ondalık sayıyı üste yuvarlar, b=10.3 , 11 çıkar

Math.Floor(b); // Ondalık sayıyı aşağıya yuvarlar, b=10.3 , 10 çıkar

Math.Round(b); // En yakın tamsayıya yuvarlar, b=10.3 , 10 çıkar, b=10.7 den 11 olur. b=10.49864 sayısı , Dikkat b=10.5 sayısını 10 yuvarlar.

Math.Min(b,c); //b ve c sayısından en küçük sayıyı verir. b=3 , c=4 ise sonuç 3 çıkar

Math.Max(b,c); //iki sayıdan en büyük olanını döndürür.

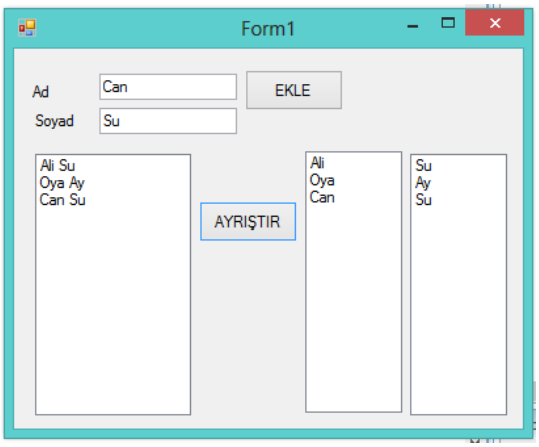
Math.Abs(b); // sayının mutlak değerini alır, yani tüm sayılar pozitif çıkar.

Math.Log10(b); // b sayısının 10 tabana göre logaritmasını alır. b=100 ise sonuç 2 çıkar $b=10^2$ => 2 çıkar.

Math.Log(b); // b sayısının In'ini almaktadır. e tabanına göre logaritmasını alır.

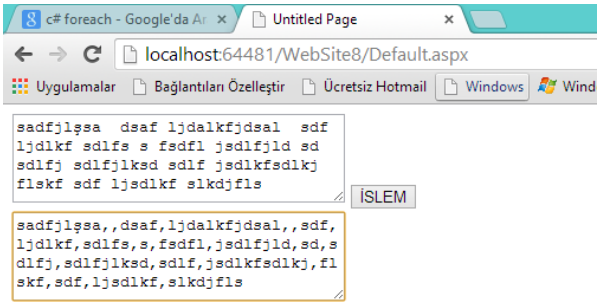
Math.Log(b,c); //c tabanında b sayısının logaritmasını alır. Örneğin b=8 ve c=2 ise sonuç 3 tür.

Örnek




```
private void button1_Click(object sender, EventArgs e)
{
	string Ad, Soyad;
	Ad = textBox1.Text;
	Soyad = textBox2.Text;

	listBox1.Items.Add(Ad + " " + Soyad);

}

private void button2_Click(object sender, EventArgs e)
{
	foreach (string Eleman in listBox1.Items)
	{
		string[] Dizi =Eleman.Split(' ');

		listBox2.Items.Add(Dizi[0]);
		listBox3.Items.Add(Dizi[1]);

	}
}
```

Örnek



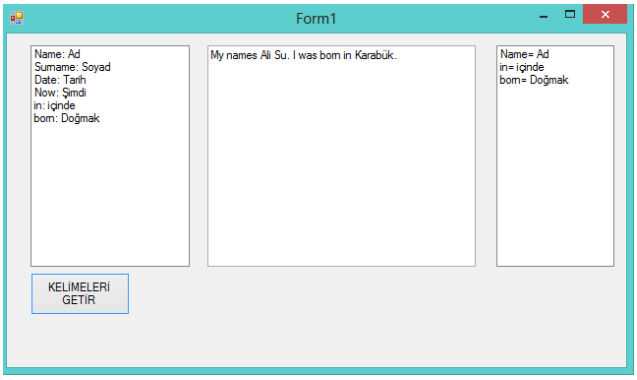

```
protected void Button1_Click(object sender, EventArgs e)
{
	string metin1 = TextBox1.Text;

	string [] Dizi = metin1.Split(new Char[] { ' ' });

	foreach (string kelime in Dizi)
	{
		TextBox2.Text = TextBox2.Text + "," + kelime;
```

```
    }
}
```

Örnek

```
protected void Button1_Click(object sender, EventArgs e)
{
    double b = 10.49864;
    int c = Convert.ToInt32(Math.Round(b));

    Response.Write(c);

}

Math.Round(b,c);  // b sayısının virgülden sonra C haneye kadar yuvarlatır.
                  b=10.234567 TL sayısını şu şekilde yapmalıyız. b=Math.Round(b,2);
                  şeklinde yazılmalıdır.
protected void Button1_Click(object sender, EventArgs e)
{
    double b = 10.4382323;
    double c = Math.Round(b,2);

    Response.Write(c);
}
```

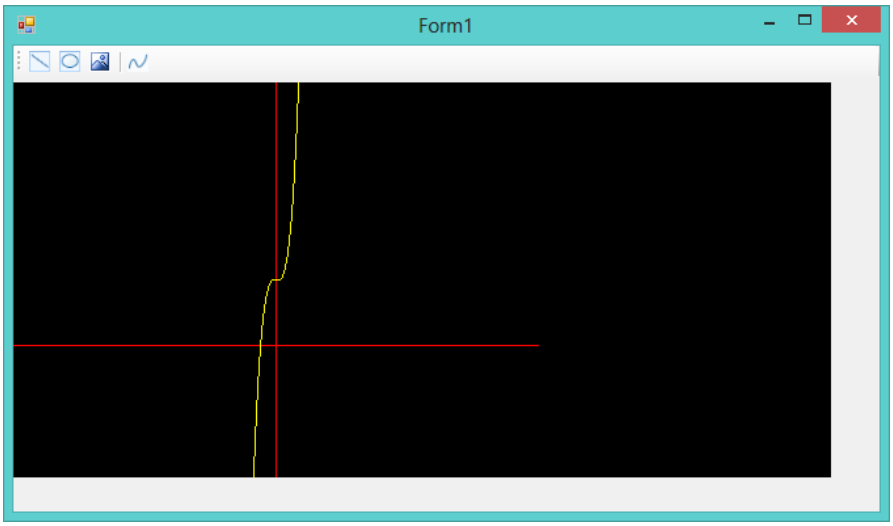
Örnek




```
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 WindowsFormsApplication22
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            for (int i = 0; i < listBox1.Items.Count; i++)
```

```
            {
                string OkunanSatir = listBox1.Items[i].ToString();

                string[] Dizi = OkunanSatir.Split(':');
                string Kelime = Dizi[0];
                string Tercume = Dizi[1];

                bool VarMi = textBox1.Text.Contains(Kelime.ToLower());

                if (VarMi == true)
                {
                    listBox2.Items.Add(Kelime+ "=" + Tercume);
                }

            }
        }
    }
}
```

Örnek




```
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 WindowsFormsApplication25
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        Graphics CizimAlani;

        Pen Kalem = new Pen(System.Drawing.Color.Yellow, 1);
        Pen Kalem2 = new Pen(System.Drawing.Color.Red, 1);
```

```
        private void btnLine_Click(object sender, EventArgs e)
        {
            CizimAlani .DrawLine(Kalem, 0, 0, 200, 100);
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            CizimAlani = pictureBox1.CreateGraphics();

        }

        private void btnGrafik_Click(object sender, EventArgs e)
        {
            int x, y, X, Y;


            CizimAlani.DrawLine(Kalem2, 0, 200, 400,  200);
            CizimAlani.DrawLine(Kalem2, 200, 0, 200, 400);


            for (x = -100; x < 100; x++)
            {
                y =-( 3 * x * x * x/100 + 50);

                X = x + 1;
                Y =-( 3 * X * X * X /100 + 50);

                CizimAlani.DrawLine(Kalem, x+200 ,y +200 , X+200, Y+200);

            }
        }
    }
}
```

TARİH ZAMAN FONKSİYONLARI

C# DateTime İşlemleri

Yazım tarihi 28 Eylül 2011 tarafından Ahmet Melih Tuna

Bu bölümde tarih ve zaman bilgileri üzerinde nasıl işlem yapıldığı konusunda bilgi verilecektir. Tarih ve zaman bilgileri üzerinde işlem yapılırken DateTime yapısından(structure) yararlanılmaktadır. Alışkanlıktan DateTime'dan bazen sınıf diye söz edilmektedir.

Today Özelliği

Tarih ve zaman bilgileri DateTime tipindeki değişkenlerde saklanmakta ve tarihsel bilgiler üzerinde işlem yapılırken DateTime yapısının metotlarından ve özelliklerden yararlanılmaktadır. Sistem tarihini elde etmek istiyorsanız DateTime structure'ın Today özelliğine bakabilirsiniz.

textBox1.Text = DateTime.Today.ToString();

Bu satır sayesinde TextBox'a sistem tarihi aktarılır. Bu şekilde elde edilen sistem tarihine ayrıca zaman başlangıcı eklenmektedir. Aşağıda verilen ekran görüntüsünü bu satırı bir metoda yazıp işlettikten sonra aldık.

Todayözelliğinden okunan tarih bilgisini TextBox yerine DateTime tipindeki değişkene aktarabilirsiniz. Bu amaçla DateTime tipinde bir değişken tanımladık.

```
DateTime Tarih = DateTime.Today;
```

DateTime tipindeki değişkenin içeriğini istediğiniz gibi kullanabilirsiniz. TextBox'a yalnızca tarihin aktarılmasını sağlamak için String sınıfının Substring() metodunu kullandık.

```
DateTime Tarih = DateTime.Today;
string Tarih_str = Tarih.ToString();
textBox1.Text = Tarih_str.Substring(0, 10);
```

Today özelliği sayesinde elde edilen tarih ve zaman bilgisinin yalnızca tarih kısmı ile ilgileniyorsanız DateTime yapısının ToShortDateString() metodundan yararlanabilirsiniz.

```
DateTime Tarih = DateTime.Today;
textBox1.Text= Tarih.ToShortDateString();
```

DateTime yapısının Today özelliği ile elde edilen tarih bilgisinin uzun halini kullanmak istiyorsanız ToShortDateString() yerine ToLongDateString() metodunu kullanmalısınız. Bu metodun nasıl kullanıldığını aşağıda görebilirsiniz.

```
DateTime Tarih = DateTime.Today;
textBox1.Text = Tarih.ToShortDateString();
textBox2.Text = Tarih.ToLongDateString();
```

ToShortDateString() ve ToLongDateString() metotları geriye string bilgi gönderdiği için bu bilgileri TextBox'a aktarmak için dönüştürme yapmaya gerek yoktur. Tarih bilgilerini string bilgiye dönüştürürken ToShortDateString ve ToLongDateString metotlarını kullandığınız gibi ToString() metodunu kullanabilirsiniz. DateTime nesnesinden kısa tarih bilgisini almak istiyorsanız "d" karakterini, uzun tarih elde etmek istiyorsanız "D" biçimleme karakterini ToString() metoduna parametre olarak vermelisiniz.

```
DateTime Tarih = DateTime.Today;
textBox1.Text = Tarih.ToString("d");
textBox2.Text = Tarih.ToString("D");
textBox3.Text = Tarih.ToString("m");
```

```
textBox4.Text = Tarih.ToString(“y”);
```

Benzer şekilde “m” format karakteri ile gün ve ay bilgisi, “y“ karakteri ile ay ve yıl bilgisi elde edilir. Bu satırlar bir metoda yazılıp işletilirse aşağıdaki gibi bir sonuç elde edilir.

Now Özelliği

Sistem saatinin hem tarih hem de zaman bilgisini elde etmek istiyorsanız DateTime yapısının Now özelliğine bakabilirsiniz. Now özelliğinin nasıl kullanıldığını anlatmak için aşağıda verilen satırı formun Load olayını temsil eden metoda yazdık.

```
textBox2.Text = DateTime.Now.ToString();
```

Bu satır sayesinde formdaki 2. TextBox’a hem sistem tarihi hem saati yazılır. Bu satırın işlevini aşağıda görebilirsiniz. Yukarıda yapıldığı gibi Substring() metodu ile DateTime tipindeki bilginin istediğiniz kısmını alıp başka amaçlarla kullanabilirsiniz.

Now özelliği sayesinde ile elde edilen sistem tarih ve saatinin yalnızca zaman kısmı ile ilgileniyorsanız DateTime tipindeki bilginin zaman kısmını ToShortTimeString() metodu ile elde edebilirsiniz. Bu metodun nasıl kullanıldığını aşağıda görebilirsiniz.

```
DateTime Zaman = DateTime.Now;
textBox1.Text = Zaman.ToShortTimeString();
```

Bu metot işletildiğinde sistem saatinin zaman bilgisi TextBox’a aktarılır. ToShortTimeString metodu ile elde edilen zaman bilgisinde saniye olmaz. Zaman bilgisinin saniye kısmı ile ilgileniyorsanız ToLongTimeString metodunu kullanmalısınız.

```
DateTime Zaman = DateTime.Now;
textBox1.Text= Zaman.ToShortTimeString();
textBox2.Text= Zaman.ToLongTimeString();
```

String Bilgiyi DateTime Tipine Dönüştürmek

Yukarıda DateTime yapısının Today ve Now özelliklerinin içeriklerini DateTime tipindeki değişkenlere aktardık. TextBox’lara girilen bilgileri DateTime tipine dönüştürebilirsiniz. Bu işlemi yaparken Convert sınıfının ToDataTime() metodunu kullanabilirsiniz.

```
DateTime Tarih, Saat;
Tarih = Convert.ToDateTime(“30.12.2007”);
```

```
Saat = Convert.ToDateTime(“12:45”);
```

Zaman bilgilerini String bilgiye dönüştürürken ToLongTimeString ve ToShortTimeString metotlarını kullandığınız gibi ToString() metodunu kullanabilirsiniz. ToString() metoduna “t” karakterini parametre olarak verirseniz DateTime bilgisinden kısa saat elde edilir. Benzer şekilde “T” karakteri ile uzun zaman bilgisi elde edilir.

```
DateTime Zaman = DateTime.Now;
textBox1.Text = Zaman.ToString(“t”);
textBox2.Text = Zaman.ToString(“T”);
textBox3.Text = Zaman.ToString(“f”);
textBox4.Text = Zaman.ToString(“F”);
textBox5.Text = Zaman.ToString(“g”);
```

Bu satırlar bir metoda yazılıp işletilirse aşağıdaki gibi bir sonuç alınır. ToString() metodu ile tarih ve zaman bilgileri biçimlenirken Format() metodunun biçim kodları kullanılmaktadır.

DayOfWeek Özelliği

Sistem tarihinden veya tarihsel bir bilgiden yararlanarak haftanın gününü öğrenmek istiyorsanız bu özelliğe bakabilirsiniz. DayOfWeek özelliğinin nasıl kullanıldığını göstermek için aşağıda verilen kodu hazırladık. Bu kodda “Tarih” adında bir değişken tanımlayıp bu değişkene Today özelliğinden yararlanarak sistem tarihini aktardık.

```
DateTime Tarih = DateTime.Today;
int gun = Convert.ToInt32(Tarih.DayOfWeek);
textBox1.Text = Tarih.ToString().Substring(0, 10);
textBox2.Text = Convert.ToString(gun);
```

Bir sonraki satırda ise sistem tarihini formdaki ilk TextBox’a aktardık. En son olarak “Tarih” değişkenin veya nesnesinin DayOfWeek özelliğinin içeriğini okuyup 2. TextBox’a aktardık. Bu kod işletildiğinde aşağıdaki gibi bir sonuç alınır.

Bu şekilde haftanın günü öğrenildikten sonra kullanıcı günün Türkçe adı konusunda bilgilendirilebilinir. Bunun için Türkçe gün adlarından meydana gelen bir dizi değişken tanımlanır. Bu işlemin nasıl yapıldığını aşağıda görebilirsiniz.

```
string[] Gunler={ “Pazar”, “Pazartesi”, “Salı”, “Çarşamba”, “Perşembe”,”Cuma”, “Cumartesi” };
DateTime Tarih = DateTime.Today;
textBox1.Text = Tarih.ToString().Substring(0, 10);
int gun = Convert.ToInt32(Tarih.DayOfWeek);
textBox2.Text = Convert.ToString(gun);
```

```
textBox3.Text = Convert.ToString(Gunler[gun]);
```

DayOfYear Özelliği

Elinizdeki tarih bilgisinin yılın kaçıncı günü olduğunu öğrenmek istiyorsanız DayOfYear özelliğine bakabilirsiniz. Bu özelliğin nasıl kullanıldığını aşağıda görebilirsiniz.

```
DateTime Tarih = DateTime.Today;
int gun = Tarih.DayOfYear;
textBox1.Text = Tarih.ToShortDateString();
textBox2.Text = Convert.ToString(gun);
```

Bu kod işletildiğinde geçerli tarih bilgisi ilk TextBox'a, bu tarih bilgisinin gün kısmı ise 2. TextBox'a aktarılır. Aşağıda verilen ekran görüntüsünü bu kodu işlettikten sonra aldık.

Bu örnekte DayOfYear özelliğinin önüne DateTime tipindeki değişkenin adını yazdık. Dolayısıyla yılbaşından bu güne kadar geçen gün sayısını öğrendik. DateTime tipindeki değişkene başka bir tarihi aktarıp yılbaşından o tarihe kadar geçen gün sayısını öğrenebilirsiniz. Bu özellikten yararlanarak 2 tarih arasındaki gün sayısını öğrenebilirsiniz. Bu işlemin nasıl yapıldığını anlatmak için forma 3 TextBox yerleştirdik.

Formdaki ilk TextBox'a geçerli sistem tarihini, 2. TextBox'a ise en son ödeme tarihini yazacağız. Ödemenin kaç gün geç yapıldığını bulup 3. TextBox'a aktaracağız. Bu amaçla Form1'in Load olayını temsil eden metoda aşağıda verilen satırları yazıp proje çalıştırıldığı zaman geçerli tarihin ilk TextBox'a yazılmasını sağlayacağız.

```
private void Form1_Load(object sender, EventArgs e)
{
DateTime Tarih = DateTime.Today;
textBox1.Text = Tarih.ToShortDateString();
}
```

Kullanıcı projeyi çalıştırıp son ödeme tarihini 2. TextBox'a girip formu tıkladığında ödemenin kaç gün geciktiğini öğrenip ilgili TextBox'a yazmak için aşağıda verilen kodu hazırladık. Bu kodda DayOfYear özelliğine 2 kez bakılmaktadır. İlk kullanımda yılbaşından son ödemeye kadar gün sayısı öğrenilmektedir.

```
private void Form1_Click(object sender, EventArgs e)
{
DateTime Bugun = DateTime.Today;
textBox1.Text = Bugun.ToShortDateString();
DateTime Son_gun = Convert.ToDateTime(textBox2.Text);
int ilk = Bugun.DayOfYear;
```

```
int son = Son_gun.DayOfYear;
textBox3.Text = Convert.ToString(ilk-son);
}
```

DayOfYear özelliğinin 2. kez kullanıldığı satırda ise yılbaşından bugüne kadarki gün sayısı öğrenilmektedir. Bu 2 tarih arasındaki gün farkı gecikmedir.

Day, Month ve Year Özellikleri

Sistem tarihinin gün bilgisini öğrenmek için Day özelliğini kullanabilirsiniz. Geçerli tarihin ay bilgisini öğrenmek istiyorsanız Month özelliğini, yıl bilgisini öğrenmek için ise Year özelliğine bakabilirsiniz. Bu özelliklerin nasıl kullanıldığını aşağıda görebilirsiniz.

```
DateTime Tarih = DateTime.Today;
textBox1.Text = Tarih.ToString().Substring(0,10);
textBox2.Text = Tarih.Day.ToString();
textBox3.Text = Tarih.Month.ToString() ;
textBox4.Text = Tarih.Year.ToString();
```

Bu satırlar bir metoda yazılıp işletilirse aşağıdaki gibi bir sonuç elde edilir. Bu sayede tarih bilgisi gün, ay ve yıl şeklinde parçalanmış oldu. Tarih bilgisinden ay adını öğrenmek istiyorsanız Visual Basic kökenli MonthName() metodunu kullanabilirsiniz. Bu metot parametre olarak ay bilgisini alıp geriye ay adını göndermektedir.

```
DateTime Tarih = DateTime.Today;
int ay = Tarih.Month;
textBox4.Text = Microsoft.VisualBasic.DateAndTime.MonthName(ay, false);
```

MonthName() metoduna 2. parametre olarak true vermeniz halinde ay adının kısa hali elde edilir. Ayın Türkçe adını Visual Basic'in MonthName metodu sayesinde öğrenmek yerine Türkçe ay adlarını bir dizi değişkene aktarıp bu dizi değişkenden yararlanabilirsiniz.

DateSerial() Metodu

Gün, ay ve yıl bilgilerinden yola çıkıp tarih bilgisi oluşturmak istiyorsanız Visual Basic'in DateSerial() metodundan yararlanabilirsiniz. Bu metot yıl, ay ve gün bilgilerini parametre olarak alıp geriye tarih bilgisini göndermektedir.

```
DateSerial(Yıl, Ay, Gün)
```

```
int gun=24;
```

```
int ay =03;
int yil=2004;
DateTime Tarih;
Tarih = Microsoft.VisualBasic.DateAndTime.DateSerial(yil, ay, gun);
textBox1.Text = Convert.ToString(Tarih);
```

Visual Basic'in DateSerial() metodu sayesinde elde edilen tarih bilgisine C# derleyicisi otomatik olarak zaman başlangıcını eklemektedir. Bu nedenle bu satırlar bir metoda yazılıp işletilirse aşağıdaki gibi bir sonuç elde edilir.

Substring() veya ToShortDateString() metodundan yararlanıp tarihe eklenen zaman başlangıcını silebilirsiniz. Yukarıda verilen koddaki son satır aşağıdaki gibi değiştirilirse TextBox'a yalnızca tarih bilgisi aktarılır.

```
textBox1.Text = Convert.ToString(Tarih).Substring(0,10) ;
```

DateAdd() Metodu

Bazen geçerli tarihe belli sayıda gün, tarih veya yıl eklenmek istenir. Aynı şekilde zaman bilgisine saat, dakika eklemek gerekebilir. Visual Basic'in DateAdd() metodu bu işlemler yapılmak istendiğinde kullanılabilir.

DateAdd(Artım Tipi, Artım Miktarı, Tarih Bilgisi)

Bu metot 3 parametreye sahiptir. İlk parametre tarih veya zaman bilgisinin artım şekli belirtilmektedir. Tarih bilgisinin gün değeri arttırılmak isteniyorsa ilk parametrede Microsoft.VisualBasic.DateInterval.Day seçeneği kullanılmalıdır.

```
DateTime Tarih = DateTime.Now;
textBox1.Text = Tarih.ToShortDateString();
object Tarih1 = Microsoft.VisualBasic.DateAndTime.DateAdd(
Microsoft.VisualBasic.DateInterval.Day, 10, Tarih);
textBox2.Text = Convert.ToString(Tarih1).Substring(0, 10);
```

Bu satırlarda önce DateTime tipinde bir değişken tanımladık ve Now() metodu sayesinde sistem tarihi ve saatini bu değişkene aktardık. Formdaki ilk TextBox'a sistem tarihini, ardından DateAdd() metodu ile sistem tarihine 10 gün ekleyip 2. TextBox'a aktardık.

DateAdd() metoduna 3. parametre olarak verilen tarihsel bilgiye bir veya birden fazla ay eklemek istemiş olsaydık 1. parametrede DateInterval.Month seçeneğini kullanırdık.

AddDays(), AddMonths() ve AddYears() Metotları

Değişik şekilde elde edilmiş geçerli tarih bilgisine gün ekleyip veya çıkarıp başka bir tarih bilgisini elde ederken DateTime yapısının bu metotlarını kullanabilirsiniz. Bu metotlar gün, ay veya yıl eklenmek veya çıkarılmak istenen tarih bilgisinin yanına yazılmaktadır.

```
DateTime Bugun, Sonra, Once;
Bugun = DateTime.Now;
textBox1.Text = Convert.ToString(Bugun).Substring(0, 10);
Sonra = Bugun.AddDays(5);
textBox2.Text = Convert.ToString(Sonra).Substring(0, 10);
Once = Bugun.AddDays(-10);
textBox3.Text = Convert.ToString(Once).Substring(0, 10);
```

Bu kod işletilirse formdaki ilk TextBox'a geçerli sistem tarihi ve 2. TextBox'a ise 5 gün sonrasının tarihi yazılır. AddDays() metoduna eksi(-) bir değeri parametre olarak verirseniz tarih bilgisine gün eklenmeyip çıkarılır.

Tarih bilgisine ay eklemek veya çıkarmak istiyorsanız AddMonths(), yıl eklemek veya çıkarmak istiyorsanız AddYears() metotlarını kullanabilirsiniz. Bu metotların nasıl kullanıldığını aşağıda görebilirsiniz.

```
DateTime Bugun, Tarih1, Tarih2;
Bugun = DateTime.Now;
Tarih1 = Bugun.AddMonths(2);
Tarih2 = Bugun.AddYears(1);
```

DaysInMonth() ve IsLeapYear() Metotları

Herhangi bir yılın herhangi bir ayının kaç çektiğini öğrenmek istiyorsanız DaysInMonth() metodundan yararlanabilirsiniz. Bu metodun nasıl kullanıldığını anlatmak için forma 3 TextBox yerleştirip aşağıda verilen kodu hazırladık.

```
int yil = Convert.ToInt32(textBox1.Text);
int ay = Convert.ToInt32(textBox2.Text);
int gun = DateTime.DaysInMonth(yil, ay);
textBox3.Text = gun.ToString();
```

DaysInMonthmetodu 2 parametreye sahiptir. İlk parametrede yıl, 2. parametrede ise gün sayısı öğrenilmek istenen ay verilmektedir. Formdaki ilk TextBox'a yılı, 2. TextBox'a gün sayısını öğrenmek istediğimiz ayı yazıp bu kodu işletince aşağıdaki gibi bir sonuç elde ettik.

4 yılda bir şubat ayı 29 çekmekte ve dolayısıyla o yılda 366 gün olmaktadır. Belirtilen yılın 366 güne sahip olup olmadığını öğrenmek istiyorsanız IsLeapYear() metodunu kullanabilirsiniz. Bu metot parametre olarak aldığı tarih bilgisinin yılı 366 gün çekiyorsa geriye true, değilse false gönderir. Bu metodun nasıl kullanıldığını aşağıda görebilirsiniz.

```
int yil = Convert.ToInt32(textBox1.Text);
bool artik = DateTime.IsLeapYear(yil);
if (artik == true)
MessageBox.Show("Bu yıl şubat 29 çekiyor");
```

Hour, Minute ve Second Özellikleri

Bu özelliklerden yararlanarak zaman bilgisinin saat, dakika ve saniye bileşenlerini öğrenebilirsiniz. Bu özellikler hakkında bilgi vermek için aşağıda verilen kodu hazırladık.

```
DateTime Zaman = DateTime.Now;
textBox1.Text = Zaman.ToString().Substring(11, 8);
textBox2.Text = Zaman.Hour.ToString();
textBox3.Text = Zaman.Minute.ToString();
textBox4.Text = Zaman.Second.ToString();
```

Bu metotta önce DateTime tipinde bir değişken tanımladık ve bu değişkene Now özelliğinden yararlanarak sistem tarih ve zaman bilgisini aktardık. Saat, dakika ve saniye bilgilerini Visual Basic'in TimeSerial() metoduna parametre olarak verip zaman bilgisi elde edebilirsiniz. Bu metodun nasıl kullanıldığını anlatmak için forma 4 TextBox yerleştirip aşağıda verilen kodu hazırladık.

```
int saat, dakika, saniye;
DateTime Zaman = DateTime.Now;
saat = Convert.ToInt32(textBox1.Text);
dakika = Convert.ToInt32(textBox2.Text);
saniye = Convert.ToInt32(textBox3.Text);
Zaman=Microsoft.VisualBasic.DateAndTime.TimeSerial(saat, dakika, saniye);
textBox4.Text = Convert.ToString(Zaman.ToLongTimeString());
```

Kullanıcı zaman bilgisini saat, dakika ve saniye cinsinden TextBox'lara girecek. Girilen bu bilgiler TimeSerial() metodu ile birleştirilip formdaki en son TextBox'a aktarılacaktır. Zaman bilgisinin mili saniye kısmını elde etmek istiyorsanız zaman nesnesinin Milisecond özelliğinden yararlanabilirsiniz.

AddHours(), AddMinutes() ve AddSeconds() Metotları

Zaman bilgisine saat, dakika veya saniye ekleme ve çıkarma gereğini duyduğunuzda bu metotları kullanabilirsiniz. Bu metotlar zaman bilgisini içeren değişken ile birlikte kullanılır ve eklenecek veya çıkarılacak saat, dakika ve saniyeyi parametre olarak alırlar.

Zaman Bilgisi.AddHours(Artım-Eksiltme Değeri)

Bu metotların nasıl kullanıldığını anlatmak için forma 3 TextBox yerleştirip aşağıda verilen satırları hazırladık. Bu satırlarda DateTime tipinde 3 değişken tanımlayıp ve Now özelliği ile sistem saatini "simdi" adını verilen değişkene aktardık. Sistem saatine saat eklemek için AddHours ve 30 dakika çıkarmak için AddMinutes metodunu kullandık.

```
DateTime Simdi, Zaman1, Zaman2;
Simdi = DateTime.Now;
Zaman1 = Simdi.AddHours(1);
Zaman2 = Simdi.AddMinutes(-30);
```

Format() Metodu İle Tarih ve Zaman Bilgilerini Biçimlemek

Tarih ve zaman bilgilerini biçimlerken Format() metodundan yararlanabilirsiniz. Aşağıda verilen kod işletilirse sistem tarihi belirlenen formata dönüştürülüp TextBox'lara aktarılır.

```
DateTime Tarih = DateTime.Today;
string str1 = String.Format("{0:d}", Tarih);
string str2 = String.Format("{0:D}", Tarih);
textBox1.Text = str1;
textBox2.Text = str2;
```

Format() metodu 2 parametreye gerek duymaktadır. İlk parametrede biçimleme yapılırken kullanılacak biçim kodu verilmektedir. 2. parametrede ise biçimlenecek veya dönüştürülecek tarih, zaman ve sayısal bilgi verilmektedir.

Bu örnekte DateTime tipinden kısa tarih elde etmek için d'yi, uzun tarihi elde etmek için ise D'yi kullandık. DateTime tipindeki değişkene tarih ile birlikte zaman bilgisini aktarırken "t" ve "T" biçim kodları ile zaman bilgisini kısa ve uzun olarak gösterebilirsiniz. "f" ve "F" biçim kodu ile hem tarih hem de zaman bilgisi elde edilmektedir.

```
DateTime Tarih = DateTime.Now;
string str1 = String.Format("{0:t}", Tarih);
string str2 = String.Format("{0:T}", Tarih);
```

```
string str3 = String.Format("{0:F}", Tarih);
```

Yalnızca sistem saatine gerek duyuyorsanız TimeOfDay() metodunu kullanabilirsiniz. Tarih ve zaman bilgilerini formatlarken kullanılan bazı kodlar aşağıda tablo halinde verildi.

Parse Metodu İle String Bilgileri Tarih ve Saate Çevirmek

Kullanıcının TextBox'a veya başka bir nesne aracılığıyla girmiş olduğu string bilgiyi tarih ve zaman olarak değerlendirmek istiyorsanız DateTime yapısının Parse() metodundan yararlanabilirsiniz. Bu metodun nasıl kullanıldığını aşağıda görebilirsiniz.

```
DateTime Tarih = DateTime.Parse(textBox1.Text);
```

TextBox'a girilen bilgi tarihsel değilse bu satır işletildiğinde hata meydana gelir. Geçersiz tarih girilip Parse() metodu ile dönüştürme yapılırken meydana gelecek hatalardan dolayı programın kırılmasını engellemek için aşağıdaki gibi try-catch bloğu hazırlanabilir.

```
DateTime Tarih;
try
{
Tarih = DateTime.Parse(textBox1.Text);
}
catch
{
MessageBox.Show("Girilen tarih geçersiz");
}
```

Bu satırları yazarken kullandığımız bilgisayarda mevcut bölgesel ayarlara göre tarih formatı gün-ay-yıl şeklindeydi. Dolayısıyla TextBox'a Ay-Gün-Yıl formatına uygun tarih girildiğinde Parse() metodu bu şekli ile kullanıldığında dönüştürme işlemini yapamaz. Parse() metoduna parametre olarak verilecek tarih bilgisinin formatını 2. parametrede belirtebilirsiniz. Bu konuda bilgi vermek için forma 2. bir TextBox yerleştirip aşağıda verilen kodu yazdık.

```
System.Globalization.CultureInfo ulke;
ulke = new System.Globalization.CultureInfo("en-US");
DateTime Tarih;
try
{
```

```
Tarih = DateTime.Parse(textBox1.Text, ulke);
textBox2.Text = Tarih.ToShortDateString();
}
catch
{
MessageBox.Show("Girilen tarih geçersiz");
}
```

Bu kod sayesinde formdaki ilk TextBox'a ay-gün-yıl formatında girilmiş tarih bilgisi DateTime yapısının Parse() metodu ile DateTime tipindeki değişkene aktarılır. Ardından bu değişkenin içeriği String'e çevrilip 2. TextBox'a aktarılmaktadır.

Ticks Özelliği

DateTime yapısının bu özelliğinden yararlanılarak bir işlemin ne kadar sürdüğünü öğrenebilirsiniz. Bu özellik hakkında bilgi vermek için aşağıda verilen kodu hazırladık.

```
long Tick_sayi = DateTime.Now.Ticks;
textBox1.Text = Tick_sayi.ToString();
```

Ticksözelliği 1 Ocak 0001'den içinde bulunulan ana kadar geçen süreyi tick sayısı cinsinden içermektedir. Bu özelliğin içeriği her 100 nano saniyede 1 artmaktadır.

Şimdi 1 Ocak 0001'den bu güne kadar geçen süreyi saniye, dakika, saat, gün ve yıl ile ifade edeceğiz. Bu amaçla forma 6 TextBox yerleştirip aşağıda verilen kodu hazırladık.

```
long Saniye, Dakika, Saat, Gun, Yil;
long Tick_sayi = DateTime.Now.Ticks;
textBox1.Text = Tick_sayi.ToString();
Saniye = Tick_sayi / 10000000;
textBox2.Text = Saniye.ToString();
Dakika = Saniye / 60;
textBox3.Text = Dakika.ToString();
Saat = Dakika / 60;
textBox4.Text = Saat.ToString();
Gun = Saat / 24;
textBox5.Text = Gun.ToString();
Yil = Gun / 365;
textBox6.Text = Yil.ToString();
```

Ticksözelliğinin içeriği her 100 nano saniyede bir arttığı için içeriğini 10.000.000'e bölerek 1 Ocak 0001'den bugüne kadar geçen süreyi saniye olarak ifade ettik. Devamında saniyeyi dakika, saat, gün ve yıla çevirdik. Bu kod işletildiğinde aşağıdaki gibi bir sonuç alınır.

Şimdi ise bir işlemin ne kadar sürdüğünü Ticks özelliğinden yararlanarak bulacağız. Bu amaçla bir for döngüsü hazırlayıp işlemin başında ve sonunda Ticks özelliğine bakıp geçen süreyi hesapladık. Kullandığımız bilgisayarda for döngüsünü 6 saniyede tamamlandı. Ticks özelliğinden yararlanıp bekleme durumları meydana getirebilirsiniz.

```
long Tick_ilk, Tick_son, Tick_fark;
long Sayi = 0;
Tick_ilk = DateTime.Now.Ticks;
for (int i = 0; i < 1000000000; i++)
{
Sayi = Sayi + i;
}
Tick_son = DateTime.Now.Ticks;
Tick_fark = Tick_son – Tick_ilk;
long Saniye = Tick_fark / 10000000;
textBox1.Text = Saniye.ToString();
```

TimeSpan Yapısı

Ticks özelliğinin içeriğini kullanırken ve tarihsel bilgiler üzerinde işlem yaparken TimeSpan adlı yapıdan yararlanabilirsiniz. Bu konuda bilgi vermek için aşağıda verilen kodu hazırladık. Bu örnek sayesinde 2 tarih arasındaki farkı gün birimi ile ifade edebileceğiz. Bu amaçla forma 2 TextBox yerleştirip aşağıda verilen odu hazırladık.

```
DateTime Tarih = DateTime.Parse(textBox1.Text);
DateTime Bugun = DateTime.Today;
textBox2.Text = Bugun.ToString("d");
```

Şimdi ise bir TimeSpan nesnesi hazırlayacağız. TimeSpan yapısının aşırı yüklenmiş 4 yapıcı metodu bulunmaktadır. İlk olarak Ticks bilgisini parametre olarak alan yapıcı metodu kullanacağız. Bu amaçla DateTime yapısının Ticks özelliğinden yararlanarak kullanıcının formdaki ilk TextBox'a girmiş olduğu tarihten yola çıkarak bir Tick değeri elde ettik.

```
DateTime Tarih, Bugun;
Tarih = DateTime.Parse(textBox1.Text);
Bugun = DateTime.Today;
```

```
textBox2.Text = Bugun.ToString(“d”);
long ilk_tick, Bugun_tick;
ilk_tick = Tarih.Ticks;
Bugun_tick = Bugun.Ticks;
System.TimeSpan Zaman;
Zaman = new TimeSpan(Bugun_tick -ilk_tick);
textBox3.Text = Zaman.Days.ToString();
```

DateTime sınıfının Now özelliğinden yararlanıp elde ettiğimiz geçerli tarihten de ikinci bir Tick değeri elde ettik. Bu 2 tick değerinin arasındaki farkı TimeSpan'ın yapıcı metoduna parametre olarak verip iki tarih arasındaki farkı TimeSpan olarak elde ettik. TimeSpan nesnesinden gün bilgisi elde ederken Days özelliğinden yararlandık. Formdaki ilk TextBox'a bilgi girilip bu kod işletildiğinde aşağıdaki gibi bir sonuç alınır.

Bu örneğe göre TimeSpan'dan yararlanarak tarih ve zaman bilgilerini değişik şekilde ifade etmek mümkündür. Bu 2 tarih arasındaki farkı saat birimi ile ifade etmek istemiş olsaydık TimeSpan nesnesinin TotalHours özelliğinden yararlanırdık. TimeSpan yapısı ayrıca TotalMinutes, TotalSeconds ve TotalMiliSeconds özelliklerine sahiptir.

Yukarıda verilen ve 2 tarih arasındaki gün sayısını bulma işleminde yararlandığımız TimeSpan nesnesini şimdi farklı bir teknikle hazırlayacağız. Yukarıda TimeSpan nesnesini hazırlarken parametre olarak iki tarih arasındaki zaman farkını Tick birimi ile vermiştik. Aşağıda ise TimeSpan nesnesini hazırlarken DateTime sınıfının Subtract() metodunu kullandık. Bu metot geriye TimeSpan nesnesini göndermektedir.

```
DateTime Tarih, Bugun;
Tarih = DateTime.Parse(textBox1.Text);
Bugun = DateTime.Today;
textBox2.Text = Bugun.ToString(“d”);
System.TimeSpan Zaman = Bugun.Subtract(Tarih);
textBox3.Text = Zaman.TotalDays.ToString();
```

Şimdi ise forma 3 TextBox ve 2 düğme yerleştirip kullanıcının "Baslat" adı verilen düğmeyi tıklaması ile "Durdur" düğmesini tıklaması arasında geçen süreyi mili saniye olarak hesaplayıp 3. TextBox'a aktaracağız. Baslat düğmesi için hazırlanan kodu aşağıda verdik.

```
private void Baslat_Click(object sender, EventArgs e)
{
DateTime Zaman = DateTime.Now;
textBox1.Text = Zaman.ToLongTimeString();
}
```

Bu kod sayesinde o anki zaman bilgisi ilk TextBox'a aktarılır. Kullanıcı "Durdur" adı verilen 2. düğmeyi tıklayıncaya kadar aradan geçen süreyi hesaplamak üzere aşağıdaki kodu yazdık.

```
private void Durdur_Click(object sender, EventArgs e)
{
DateTime Zaman = DateTime.Now;
textBox2.Text = Zaman.ToLongTimeString();
DateTime saat = DateTime.Parse(textBox1.Text);
DateTime simdi = DateTime.Now;
long ilk_tick, son_tick;
ilk_tick = saat.Ticks;
son_tick = simdi.Ticks;
System.TimeSpan fark;
fark = new TimeSpan(son_tick – ilk_tick);
textBox3.Text = fark.Milliseconds.ToString();
}
```

Tarihsel Bilgi Kontrolü

Daha önceki konulardan bildiğiniz gibi yıl, ay ve gün bilgileri birleştirilip tarihsel bilgi elde etmek mümkündür. Bazen kullanıcının girdiği bilgi tarihsel tipe uygun olmayabilir. Bu şartlarda programın kırılmasını engellemek için hata kontrolü yapılmalıdır.

Buradaki amacımız sizlere daha sonra sözü edilecek metotların işlevlerini hatırlatmak olduğu için tarihsel bilgi kontrolünü kendimiz yapacağız. Bu amaçla projeye bir CS dosyası ekledik ve bu CS dosyasında yandaki gibi bir Class hazırladık.

Öncelikle bu metotta kendisine parametre olarak verilen bilgiyi uzunluk açısından kontrol ettik. Bu metodu gönderilen bilgi 8 karakterden az veya 10 karakterden fazla ise kullanıcıya "Hata Var" mesajını verdik. Bu metodu çağırmak için "Aktar" adı verilen düğmenin Click olayını temsil eden metodu aşağıdaki gibi düzenledik.

```
private void Aktar_Click(object sender, EventArgs e)
{
Kontrol_sinifi nesne;
nesne = new Kontrol_sinifi();
textBox1.Text = nesne.Tarih_kontrol(textBox1.Text);
}
```

Ancak kullanıcı TextBox'a "111.1.2004" gibi bilgi yazarsa bu metot işe yaramaz ve "Tarih= Convert.ToDateTime(textBox1.Text)" satırı hata verir. Bu örnekte tarihsel bilgilerin uygunluğunu araştırırken .NET Framework ile gelen sınıfları kullanmak istemediğimiz için yukarıda verilen Tarih_kontrol() metodunu aşağıdaki gibi geliştirdik.

```
public string Tarih_kontrol(string tarih)
{
int uzunluk = tarih.Length;
if (uzunluk > 10 || uzunluk < 8)
return "Hata var";
string gun = Microsoft.VisualBasic.Strings.Left(tarih, 2);
if (Convert.ToInt16(gun) > 31)
return "Hata var";
return tarih;
}
```

Tarihsel bilginin gününü kontrol ederken sol taraftan 2 haneyi alıp 31'den büyük olup olmadığını araştırdık. Kullanıcı TextBox'a gün bilgisini 1 hane olarak girerse(1.12.2006 gibi) bu kodun işe yaramayacağını belirtmek isteriz. Şimdi ise tarih bilgisinin ay kısmını kontrol edeceğiz. Yukarıda verilen koda bu amaçla eklemeler yaptık. Kullanıcının gün, ay ve yıl bilgilerini "/" ile birbirinden ayırmak istemesi halinde bu kod hata verir.

```
public string Tarih_kontrol(string tarih)
{
int Konum1 = 0;
int Konum2 = 0;
int uzunluk = tarih.Length;
if (uzunluk > 10 || uzunluk < 8)
return "Hata var";
string gun = Microsoft.VisualBasic.Strings.Left(tarih, 2);
if (Convert.ToInt16(gun) > 31)
return "Hata var";
Konum1 = tarih.IndexOf('.', 1) +1;
Konum2 = tarih.IndexOf('.', Konum1 +1)+1;
if (Convert.ToInt16(tarih.Substring(Konum1, (Konum2-Konum1)-1)) > 12)
return "Hata Var";
return tarih;
}
```

Format Kodu

İşlevi

Örnek

D	Kısa tarih	27.03.2006
D	Uzun tarih	27 Mart 2006 Pazartesi
F	Uzun tarih ve kısa zaman birlikte	27 Mart 2006 Pazartesi 15:24
F	Uzun tarih ve uzun zaman birlikte	27 Mart 2006Pazartesi 15:24:43
Dd	Sadece tarihin günü	27
Ddd	Günün kısa adı	Pzt
Dddd	Günün tam adı	Pazartesi
M	Ay bilgisi	27 Mart
MM	Ay tek basamaklı ise 01 şeklinde	03
MMM	Ayın kısa adı	Mar
MMMM	Ayın tam adı	Mart
Yy	Yıl bilgisi 2 haneli olarak gösterilir	06
Yyyy	Yıl bilgisi 4 haneli olarak gösterilir	2006
T	Zaman bilgisinin saati gösterilir	15:33
Mm	Zaman bilgisinin dakikası gösterilir	33

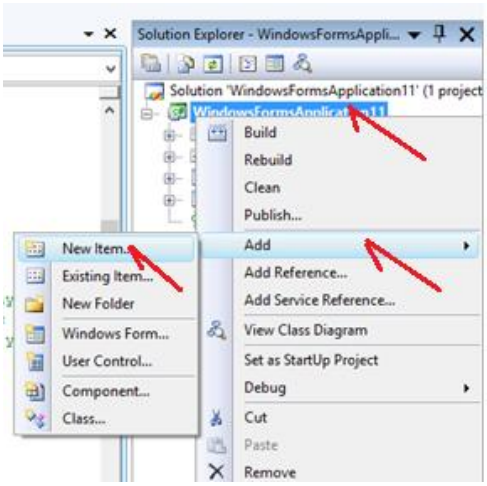
FORMLAR ARASI BİLGİ TAŞIMA

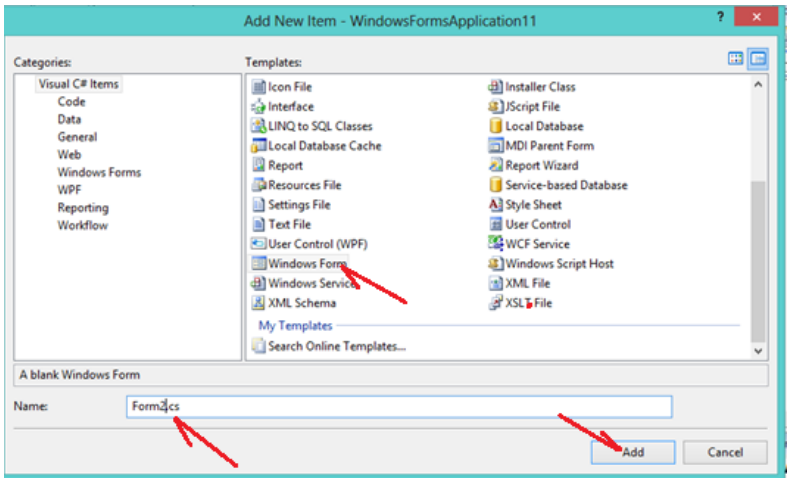
Masaüstü program hazırlarken projemizde bir çok form kullanabiliriz. Bu formlar geçişin nasıl olacağı ve önceki formlardaki bilgilerin diğer formlara nasıl taşınacağına dair aşağıdaki örneği inceleyin.

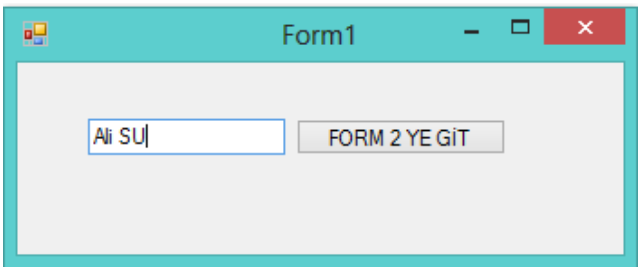
Örnek

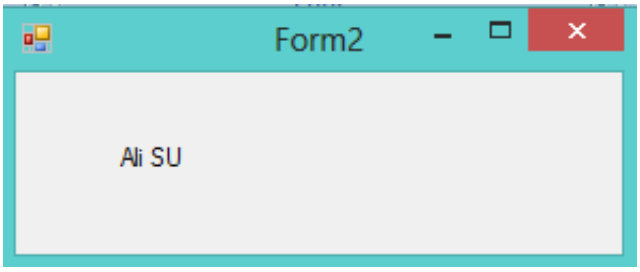
Form1 üzerine 1 buton ve textbox ekleyin. Texbox yazılan isim butona tıklanınca başka bir form açılsın ve orada bulunan labelda görüntülensin.

Öncelikle projemize yeni bir form eklememiz gerekir. Bunun için “Solution Explorer” dan proje başlığı üzerine sağ tuşa tıklayıp yeni form u ekleyelim.









Form1 kodları

```
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;

using System.Collections;

namespace WindowsFormsApplication11
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            string AdSoyad = textBox1.Text; //Yerel degişkene bilgi textbox dan
alınıyor.

            Form2 YeniForm2 = new Form2(); //Projemize tasarım esnasında
oluşturduğumuz Form2 nin aynısında bir nesne (kopyasını) üretiyoruz. Bu YeniForm2,
Form2 nin tüm özelliklerini taşır. Çalışma esnasında artık bu yeni nesne
kullanılacak.
            YeniForm2.GelenBilgi = AdSoyad; //Form2 kodları arasında (From2.cs)
"GelenBilgi" public ifadesi kullanarak global tanımlamıştık. Dolayısıyla oradaki
tanımlama nedeniyle burada bu komutu görmektedir.
```

```
            YeniForm2.Show();                //Yeni Form2 açılıyor. YeniFrom2 yi
bilgiler degişkene yüklendikten sonra görüntülenmeli. yoksa değişkendeki bilgileri
göremeyiz.

            this.Hide(); //Bulunduğumuz form olan Form1 kapatıyoruz.

        }
    }
}
```

Form2 nin kodları

```
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 WindowsFormsApplication11
{
    public partial class Form2 : Form
    {
        public Form2()
        {
            InitializeComponent();
        }

        public string GelenBilgi = null; //Bu degişkenin Form1 de görülebilmesi
için başında "public" ifadesinin olması gerekir.

        private void Form2_Load(object sender, EventArgs e)
        {
            label1.Text = GelenBilgi; //"GelenBilgi" değişkenine bilgi Form1 de
aktarılmıştı. public değişken (global) olduğu için oradan buraya gelişte bilgi
kaybolmamış oluyor.
        }
    }
}
```