

Algoritma, bir problemin çözümünde izlenen adımların tümüdür.

Kodlama, bir algoritmayı ifade etmek amacıyla bilgisayara yapmak istediğimiz işi anlatmamızdır.

Programlama dili, belli standartlar içinde yazılım geliştirmeye yarayan bilgisayar dilidir.

Derleyici, yazmış olduğumuz kodları makinenin anlayacağı kodlara dönüştüren birimlerdir. Her programlama dilinin kendine has derleyicisi bulunmaktadır. C#'ın derleyicisi ise Visual Studio'dur.

Değişken, bilgisayar ve matematik biliminde sembolik bir ifade veya bir niceliği (miktarı) ifade etmek için kullanılan yapılardır.

Sembolik değerlerler ülke, şehir, kişi, öğrenci bilgileri olabilir.

Değişken türleri; string, int, double, bool, char, float, decimal, byte, sbyte, short, ushort

Değişkenler, değişken_türü değişken_adı olarak yazılır.

Değişkenler adlandırılırken, araya boşluk koyulmaz

Değişkenler adlandırılırken, sayı ve sembol ile başlamaz.

Örnek

```
static void Main(string[] args)
{
    Console.Write("Salih");
    Console.Write("Toraman");
    Console.Read();
}
```

SalihToraman

Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("Salih");
    Console.WriteLine("Toraman");
    Console.ReadLine();
}
```

Salih

Toraman

string, metin veya karakter türünde değer alan değişken türüdür.

Bu değişkene ait değerler çift tırnak sembolü içine yazılır.

string deger;

Herhangi bir değişkene değer ataması yapmak için = operatörü kullanılır.

deger = "merhaba";

Örnek

```
static void Main(string[] args)
{
    string sehir;
    sehir = "İstanbul";
    Console.Write(sehir);
    Console.Read();
}
```

İstanbul

Örnek

```
static void Main(string[] args)
{
    string sehir;
    sehir = "İstanbul";
    Console.Write("sehir");
    Console.Read();
}
```

sehir

Örnek

```
static void Main(string[] args)
{
    string sehir, ilce;
    sehir = "İstanbul";
    ilce = "Bagcilar";
    Console.Write(sehir + " " + ilce);
    Console.Read();
}
```

İstanbul Bagcilar

int, aritmetik tam sayılar üzerinde işlemler yapan değişken türüdür. Mesela öğrencilerin sınav notlarının hesaplanmasında, aritmetik 4 işlemde alışveriş de müşterinin ödemesi gereken toplam tutarın hesaplanmasında gibi pek çok yerde **int** kullanılır.

Kapasiteler

Tür	Boyut	Kapasite	Örnek
byte	1 bayt	0, ..., 255 (tam sayı)	byte a=5;
sbyte	1 bayt	-128, ..., 127 (tam sayı)	sbyte a=5;
short	2 bayt	-32768, ..., 32767 (tam sayı)	short a=5;
ushort	2 bayt	0, ..., 65535 (tam sayı)	ushort a=5;
int	4 bayt	-2147483648, ..., 2147483647 (tam sayı)	int a=5;
uint	4 bayt	0, ..., 4294967295 (tam sayı)	uint a=5;
long	8 bayt	-9223372036854775808, ..., 9223372036854775807 (tam sayı)	long a=5;
ulong	8 bayt	0, ..., 18446744073709551615 (tam sayı)	ulong a=5;
float	4 bayt	$\pm 1.5 \cdot 10^{-45}$, ..., $\pm 3.4 \cdot 10^{38}$ (reel sayı)	float a=5F; veya float a=5f;
double	8 bayt	$\pm 5.0 \cdot 10^{-324}$, ..., $\pm 1.7 \cdot 10^{308}$ (reel sayı)	double a=5; veya double a=5d; veya double a=5D;
decimal	16 bayt	$\pm 1.5 \cdot 10^{-28}$, ..., $\pm 7.9 \cdot 10^{28}$ (reel sayı)	decimal a=5M; veya decimal a=5m;



Örnek

```
static void Main(string[] args)
{
    int sayi1, sayi2, toplam;
    sayi1 = 12;
    sayi2 = 25;
    toplam = sayi1 + sayi2;
    Console.Write(toplam);
    Console.Read();
}
```

Örnek

```
static void Main(string[] args)
{
    int s1, s2, toplam, carpim, bolum, fark;
    Console.WriteLine("***** Aritmetik İşlemler *****");
    Console.WriteLine();
    s1 = 20;
    s2 = 5;
    toplam = s1 + s2;
    fark = s1 - s2;
    carpim = s1 * s2;
    bolum = s1 / s2;
    Console.WriteLine("Toplam: " + toplam);
    Console.WriteLine("Fark: " + fark);
    Console.WriteLine("Çarpım: " + carpim);
    Console.WriteLine("Bölüm: " + bolum);
    Console.WriteLine();
    Console.WriteLine("***** Aritmetik İşlemler *****");
    Console.Read();

    ***** Aritmetik İşlemler *****
```

Toplam: 25

Fark: 15

Çarpım: 100

Bölüm: 4

******* Aritmetik İşlemler *******

Not: O iki kod arada boşluk olması için yazıldı.

```
Console.WriteLine("Toplam: " + toplam);
```

bu kod şu anlama gelir; ekrana **Toplam:** yazdır + **toplam değişkenini** getir.

Sonuç ise yukarıda gördüğün gibi gelir: Toplam: 25

Örnek

```
static void Main(string[] args)
{
    string ad, soyad, bolum, ders;
    int s1, s2, s3, ort;
    //String değişkenlerin atamaları
    ad = "Salih";
    soyad = "Toraman";
    bolum = "Yazılım Mühendisliği";
    ders = "Algoritma Programlama";
    //Int değişkenlerin atamaları
    s1 = 65;
    s2 = 70;
    ort = (s1 + s2) / 2;
    //Yazdırma Komutları
    Console.WriteLine("***** Öğrenci Bilgi Sistemi *****");
    Console.WriteLine();
    Console.WriteLine("Öğrenci Adı Soyadı: " + ad + " " + soyad);
    Console.WriteLine("Bölüm: " + bolum);
    Console.WriteLine("Ders: " + ders);
    Console.WriteLine();
    Console.WriteLine("Sınav1: " + s1);
    Console.WriteLine("Sınav2: " + s2);
    Console.WriteLine("Ortalamanız: " + ort);
    Console.Read();
}
***** Öğrenci Bilgi Sistemi *****
```

Öğrenci Adı Soyadı: Salih Toraman

Bölüm: Yazılım Mühendisliği

Ders: Algoritma Programlama

Sınav1: 65

Sınav2: 70

Ortalamanız: 67

Double Değişkenler, yalnızca tam sayılar değil ondalıklı sayılar üzerinde de işlem yapan değişken türüdür.

double sayi;

sayi = 24.9;

Örnek

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

```
{
```

```
double sayi;
```

```
sayi = 24.9;
```

```
Console.WriteLine(sayi);
```

```
Console.Read();
```

```
}
```

24,9

Örnek

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

```
{
```

```
double s1, s2, ort;
```

```
s1 = 78;
```

```
s2 = 85;
```

```
ort=(s1+s2) / 2;
```

```
Console.WriteLine(ort);
```

```
Console.Read();
```

```
}
```

81,5

ReadLine(), kullanıcıdan alınan metinsel ifadeyi hafızada tutar.

Tür: string

ReadKey(), klavyeden basılan tuşun bilgisini verir.

Read(), girilen parametrenin sadece ilk karakterinin ascii karşılığını verir.

Örnek

```
static void Main(string[] args)
{
    string sehir;
    Console.WriteLine("Lütfen şehrinizi girin: ");
    sehir=Console.ReadLine();
    Console.WriteLine("Girdiğiniz şehir: " + sehir);
    Console.Read();
}
```

Lütfen şehrinizi girin: Bursa yaz ve enter'a bas

Girdiğiniz şehir: Bursa çıkar karşına.

Örnek

```
static void Main(string[] args)
{
    string sehir, ilce;
    Console.WriteLine("Lütfen şehrinizi girin: ");
    sehir=Console.ReadLine();
    Console.WriteLine("Lütfen ilçeyi girin: ");
    ilce = Console.ReadLine();
    Console.WriteLine("Girdiğiniz şehir ve ilçe: " + sehir + "-" + ilce);
    Console.Read();
}
```

Lütfen şehrinizi girin: Bursa yaz enter'a bas aşağıdaki soru gelir.

Lütfen ilçeyi girin: Yıldırım yaz enter'a bas aşağıdaki sonuç gelir.

Girdiğiniz şehir ve ilçe: Bursa-Yıldırım

Int Dönüşümler

Dönüşümler için anahtar kelime: **Convert**

Örnek

```
static void Main(string[] args)
{
    int sayi;
    Console.Write("Sayıyı Giriniz: ");
    sayi = Convert.ToInt16(Console.ReadLine());
    Console.Write(sayi);
    Console.Read();
}
```

Sayıyı Giriniz: 20 yaz enter'a bas

20 gelir.

Not: sayi = Convert.ToInt16(Console.ReadLine()); bu şu demek parantez içindeki ifadeyi ToInt16 aralığına dönüştür. Parantez içindeki ifade ise benim klavyeden girdiğim değeri tutar. ToInt16 metod olduğu için yazdıktan sonra **parantez açmayı** asla unutma.

Örnek

```
static void Main(string[] args)
{
    int s1, s2, toplam;
    Console.Write("Sayı 1: ");
    s1 = Convert.ToInt16(Console.ReadLine());
    Console.Write("Sayı 2: ");
    s2 = Convert.ToInt16(Console.ReadLine());
    toplam = s1 + s2;
    Console.Write(toplam);
    Console.Read();
}
```

Sayı 1: 23 yaz enter'a bas aşağıdaki soru gelir.

Sayı 2: 25 yaz enter'a bas aşağıdaki sonuç gelir.

Soru: Klavyeden kısa ve uzun kenarı girilen dikdörtgenin ala ve çevresini hesaplayan kodu yazınız.

Çözüm:

```
static void Main(string[] args)
{
    int kısa, uzun, alan, cevre;
    Console.Write("Kısa kenar: ");
    kısa = Convert.ToInt16(Console.ReadLine());
    Console.Write("Uzun kenar: ");
    uzun = Convert.ToInt16(Console.ReadLine());
    alan = kısa * uzun;
    cevre = kısa + kısa + uzun + uzun;
    Console.WriteLine("Alan: " + alan);
    Console.WriteLine("Çevre: " + cevre);
    Console.Read();
}
```

Kısa kenar: 10 yaz enter'a bas aşağıdaki soru gelir.

Uzun kenar: 20 yaz enter'a bas aşağıdaki sonuçlar gelir.

Alan: 200

Çevre: 60

Soru: Klavyeden girilen 2 sayıya aritmetik 4 işlem uygulayan kodu yazın.

```
static void Main(string[] args)
{
    int sayi1, sayi2, toplam, fark, carpim, bolum;
    Console.Write("Sayı 1: ");
    sayi1 = Convert.ToInt16(Console.ReadLine());
    Console.Write("Sayı 2: ");
    sayi2 = Convert.ToInt16(Console.ReadLine());
    toplam = sayi1 + sayi2;
    fark = sayi1 - sayi2;
    carpim = sayi1 * sayi2;
    bolum = sayi1 / sayi2;
    Console.WriteLine("Toplam: " + toplam);
    Console.WriteLine("Fark: " + fark);
    Console.WriteLine("Çarpım: " + carpim);
    Console.WriteLine("Bölüm: " + bolum);
    Console.Read();
}
```

Sayı 1: 10 yaz enter'a bas aşağıdaki soru gelir.

Sayı 2: 20 yaz enter'a bas aşağıdaki sonuçlar gelir.

Toplam: 30

Fark: -10

Çarpım: 200

Bölüm: 0

Double Dönüşümler, klavyeden ondalıklı sayı girişi yapmak için kullanılır.

Dönüşüm komutu: Convert.ToDouble();

Örnek

```
static void Main(string[] args)
{
    //Klavyeden girilen ondalıklı sayıyı ekrana yazan kod bloğu
    double sayi1;
    Console.Write("Sayıyı Giriniz: ");
    sayi1 = Convert.ToDouble(Console.ReadLine());
    Console.Write("Dönüştürdüğünüz değer: " + sayi1);
    Console.Read();
}
```

Sayıyı Giriniz: 4,35 yaz enter'a bas aşağıdaki sonuç gelir.

Dönüştürdüğünüz değer: 4,35

Örnek

```
static void Main(string[] args)
{
    //Klavyeden girilen ondalıklı sayıyı toplayan ekrana yazan kod bloğu
    double sayi1, sayi2, toplam;
    Console.Write("Sayı 1'i Giriniz: ");
    sayi1 = Convert.ToDouble(Console.ReadLine());
    Console.Write("Sayı 2'yi Giriniz: ");
    sayi2 = Convert.ToDouble(Console.ReadLine());
    toplam = sayi1 + sayi2;
    Console.Write("Sonuç: " + toplam);
    Console.Read();
}
```

Sayı 1'i Giriniz: 1,35 yaz enter'a bas aşağıdaki soru gelir.

Sayı 2'yi Giriniz: 5,67 yaz enter'a bas aşağıdaki sonuç gelir.

Sonuç: 7,02

Char, tek karakter üzerinde işlemler gerçekleştiren değişken türüdür. String değişkenler istediğimiz kadar karakter üzerinde işlem yaparken, c değişkenlerde sadece 1 karakter üzerinde işlem yapılmaktadır. Karakterler tek tırnak sembolü arasına yazılır. Dönüşüm metodu **Convert.ToChar()** şeklindedir. Şifreleme ve şifre çözme işlemlerinde çokça kullanılır.

Örnek

```
static void Main(string[] args)
{
    char degisken;
    degisken = 'a';
    Console.WriteLine(degisken);
    Console.Read();
}
```

a

Örnek

```
static void Main(string[] args)
{
    char deger;
    deger = '6';
    Console.WriteLine(deger);
    Console.Read();
}
```

6

Örnek

```
static void Main(string[] args)
{
    char secim;
    Console.Write("Lütfen seçiminizi yapınız: ");
    secim=Convert.ToChar(Console.ReadLine());
    Console.Write("Seçiminiz: " + secim);
    Console.Read();
}
```

Lütfen seçiminizi yapınız: B yaz enter'a bas aşağıdaki sonuç gelir.
Seçiminiz: B

Byte, 0-255 arası tam sayılar üzerinde işlem yapan değişken türüdür.

Örnek

```
static void Main(string[] args)
{
    byte sayi;
    sayi = 256;
    Console.Write(sayi);
    Console.Read();
}
```

256 gelmez çünkü byte'da sınır 0-255 arasında olmalıdır.

Örnek

```
static void Main(string[] args)
{
    byte sayi1,sayi2,toplam ;
    sayi1 = 24;
    sayi2 = 36;
    toplam =Convert.ToByte(sayi1 + sayi2);
    Console.Write(toplam);
    Console.Read();
}
```

60

Örnek

```
static void Main(string[] args)
{
    byte s1, s2, carpim;
    Console.Write("1.Sayı: ");
    s1 = Convert.ToByte(Console.ReadLine());
    Console.Write("2.Sayı: ");
    s2= Convert.ToByte(Console.ReadLine());
    carpim = Convert.ToByte(s1 * s2);
    Console.Write(carpim);
    Console.Read();
}
```

1.Sayı: 10 yaz enter'a bas aşağıdaki soru gelir.

2.Sayı: 5 yaz enter'a bas aşağıdaki sonuç gelir.

50

Sbyte, -128 ile +128 arası tam sayılar üzerinde işlem yapan değişken türü.

S: signed

Örnek

```
static void Main(string[] args)
{
    sbyte sayi;
    sayi = 123;
    Console.Write(sayi);
    Console.Read();
}
```

123

Örnek

```
static void Main(string[] args)
{
    sbyte sayi2;
    Console.Write("Sayıyı Giriniz: ");
    sayi2 = Convert.ToSByte(Console.ReadLine());
    Console.Write(sayi2);
    Console.Read();
}
```

Sayıyı Giriniz: 50 yaz enter'a bas aşağıdaki sonuç gelir.

50

Örnek

```
static void Main(string[] args)
{
    sbyte s1, s2, toplam;
    Console.Write("Sayı1: ");
    s1 = Convert.ToSByte(Console.ReadLine());
    Console.Write("Sayı2: ");
    s2 = Convert.ToSByte(Console.ReadLine());
    toplam = Convert.ToSByte(s1 + s2);
    Console.Write(toplam);
    Console.Read();
}
```

Sayı1: 6 yaz enter'a bas aşağıdaki soru gelir.

Sayı2: 5 yaz enter'a bas aşağıdaki sonuç gelir.

11

Float, ondalıklı sayılar üzerinde işlem yapmak için kullanılan değişken türü. Boyut aralığı double değişkenden küçüktür.

Örnek

```
static void Main(string[] args)
{
    float sayi;
    sayi = 345.67f;
    Console.Write(sayi);
    Console.Read();
}
```

345,67

Not = Float değişkenlerde kod kısmına ondalıklı bir değişken gönderirken sonuna **f** harfi eklememiz gerekiyor.

Örnek

```
static void Main(string[] args)
{
    float s1, s2, toplam;
    Console.Write("Sayı 1: ");
    s1 = float.Parse(Console.ReadLine());
    Console.Write("Sayı 2: ");
    s2 = float.Parse(Console.ReadLine());
    toplam = s1 + s2;
    Console.Write(toplam);
    Console.Read();
}
```

Sayı 1: 3,44 yaz enter'a bas aşağıdaki soru gelir.

Sayı 2: 2,58 yaz enter'a bas aşağıdaki sonuç gelir.

6,02

Short, -32768 ile +32767 arasındaki tam sayılar üzerinde işlem yapar. ToInt16 aralığının karşılığıdır.

Örnek

```
static void Main(string[] args)
{
    short s1, s2, toplam;
    s1 = 650;
    s2 = 3456;
    toplam = (short) (s1 + s2);
    Console.Write(toplam);
    Console.Read();
}
```

4106

Örnek

```
static void Main(string[] args)
{
    short a, b, c;
    Console.Write("A: ");
    a = short.Parse(Console.ReadLine());
    Console.Write("B: ");
    b = short.Parse(Console.ReadLine());
    c = (short)(a + b);
    Console.Write(c);
    Console.Read();
}
```

A: 2 yaz enter'a bas aşağıdaki soru gelir.

B: 5 yaz enter'a bas aşağıdaki sonuç gelir.

7

Not: c'yi şu şekilde de yazabilirdin c = **Convert.ToInt16**(a+b);

Ushort, 0 ile 65535 arasındaki tam sayılar üzerinde işlem yapar.

Örnek

```
static void Main(string[] args)
{
    ushort sayi1, sayi2, toplam;
    sayi1 = 34;
    sayi2 = 56;
    toplam = (ushort)(sayi1 + sayi2);
    Console.Write(toplam);
    Console.Read();
}
```

90

Örnek

```
static void Main(string[] args)
{
    ushort kenar1, kenar2, alan, cevre;
    Console.Write("Kısa Kenarı Giriniz: ");
    kenar1 = ushort.Parse(Console.ReadLine());
    Console.Write("Uzun Kenarı Giriniz: ");
    kenar2 = ushort.Parse(Console.ReadLine());
    alan = (ushort)(kenar1 * kenar2);
    cevre = (ushort)(kenar1 + kenar1 + kenar2 + kenar2);
    Console.Write("Alan: " + alan + " " + "Çevre: " + cevre);
    Console.Read();
}
```

Kısa Kenarı Giriniz: 10 yaz enter'a bas aşağıdaki soru gelir.

Uzun Kenarı Giriniz: 15 yaz enter'a bas aşağıdaki sonuç gelir.

Alan: 150 Çevre: 50

Decimal, double değişkenlere göre daha geniş bir aralığı bulunan ve ondalıklı sayısal işlemlerde kullanılan değişken türüdür.

Örnek

```
static void Main(string[] args)
{
    decimal sayi;
    sayi = 4.56m;
    Console.Write(sayi);
    Console.Read();
}
```

4,56

Örnek

```
static void Main(string[] args)
{
    decimal sayi1, sayi2, fark;
    sayi1 = 20;
    sayi2 = 12;
    fark = sayi1 - sayi2;
    Console.Write(fark);
    Console.Read();
}
```

8

Örnek

```
static void Main(string[] args)
{
    decimal kenar, alan, cevre;
    Console.Write("Kenarı Giriniz: ");
    kenar = Convert.ToDecimal(Console.ReadLine());
    alan = kenar * kenar;
    cevre = kenar * 4;
    Console.Write("Alan: " + alan + " " + "Çevre: " + cevre);
    Console.Read();
}
```

Kenarı Giriniz: 8 yaz enter'a bas aşağıdaki sonuç gelir.

Alan: 64 Çevre: 32

Kenarı Giriniz: 4,15 yaz enter'a bas aşağıdaki sonuç gelir.

Alan: 17,2225 Çevre: 16,60

Bool, True-False şeklinde değer alan değişken türüdür.

Bool ifadesi boolean ifadesinin takma adıdır. (alias)

Evli mi bekar mı ?

Çalışıyor mu işsiz mi ?

Geçti mi kaldı mı ?

Örnek

```
static void Main(string[] args)
{
    bool durum;

    durum = true;

    Console.Write("Öğrenci sınavı geçti mi: " + durum);
    Console.Read();
}
```

Öğrenci sınavı geçti mi: True

Örnek

```
static void Main(string[] args)
{
    bool uyemi;

    Console.Write("Kullanıcı sisteme üye mi: ");

    uyemi = Convert.ToBoolean(Console.ReadLine());

    Console.Write("Kullanıcının sistem üyelik durumu: " + uyemi);
    Console.Read();
}
```

Kullanıcı sisteme üye mi: False yaz enter'a bas aşağıdaki sonuç gelir.

Kullanıcının sistem üyelik durumu: False

```
{  
    Console.WriteLine(byte.MaxValue);  
    255  
    Console.WriteLine(byte.MinValue);  
    0  
    Console.WriteLine(int.MaxValue);  
    2147483647  
    Console.WriteLine(int.MinValue);  
    -2147483648  
    Console.WriteLine(float.MaxValue);  
    3,402823E+38  
    Console.WriteLine(float.MinValue);  
    -3,402823E+38  
    Console.WriteLine(sbyte.MaxValue);  
    127  
    Console.WriteLine(sbyte.MinValue);  
    -128  
    Console.Read();  
}
```

Karar yapıları, herhangi bir şartın sağlanması durumunda yapılacak olan işlemlerin belirlendiği komut bloklarıdır.

2 temel komutu vardır: **if – else**

Syntax yapısı,

if (şart)

{

işlem 1

}

else

{

işlem 2

}

Operatörler:

= atama

= = eşit mi

> büyük mü

< küçük mü

>= büyük veya eşit mi

<= küçük veya eşit mi

!= eşit değilse

& ve

| veya

% mod

Örnek

```
static void Main(string[] args)
{
    string sehir;
    Console.Write("Şehir adı: ");
    sehir = Console.ReadLine();
    if (sehir == "Adana")
    {
        Console.Write("Doğru şehir");
    }
    else
    {
        Console.Write("Yanlış şehir");
    }
    Console.Read();
}
```

Şehir adı: Adana yaz enter'a bas **eğer doğruysa** aşağıdaki sonuç gelir.

Doğru şehir

Şehir adı: İstanbul yaz enter'a bas **değilse** aşağıdaki sonuç gelir.

Yanlış şehir

Örnek

```
static void Main(string[] args)
{
    int sayi;
    Console.Write("Sayıyı giriniz: ");
    sayi = Convert.ToInt16(Console.ReadLine());
    if (sayi == 23)
    {
        Console.Write("Sayı doğru girildi");
    }
    else
    {
        Console.Write("Sayı yanlış girildi");
    }
    Console.Read();
}
```

Sayıyı giriniz: 23 yaz enter'a bas **eğer doğruysa** aşağıdaki sonuç gelir.

Sayı doğru girildi

Örnek

```
static void Main(string[] args)
{
    int s1, s2, ort;

    Console.Write("Sınav 1: ");
    s1 = Convert.ToInt16(Console.ReadLine());
    Console.Write("Sınav 2: ");
    s2 = Convert.ToInt16(Console.ReadLine());
    ort = (s1 + s2) / 2;
    Console.Write("Ortalamanız: " + ort + " ");
    if (ort >= 65)
    {
        Console.Write("Geçtiniz");
    }
    else
    {
        Console.Write("Kaldınız");
    }
    Console.Read();
}
```

Sınav 1: 90 yaz enter'a bas aşağıdaki soru gelir.

Sınav 2: 70 yaz enter'a bas aşağıdaki sonuç gelir.

Ortalamanız: 80 Geçtiniz

Sınav 1: 35 yaz enter'a bas aşağıdaki soru gelir.

Sınav 2: 55 yaz enter'a bas aşağıdaki sonuç gelir.

Ortalamanız: 45 Kaldınız

Örnek

```
static void Main(string[] args)
{
    string kullanıcı, sifre;
    Console.Write("Kullanıcı Adınız: ");
    kullanıcı = Console.ReadLine();
    Console.Write("Şifreniz: ");
    sifre = Console.ReadLine();
    if (kullanıcı == "admin" & sifre == "123456")
    {
        Console.Write("Hoşgeliniz");
    }
    else
    {
        Console.Write("Hata");
    }
    Console.Read();
}
```

Kullanıcı Adınız: **admin** yaz enter'a bas aşağıdaki soru gelir.

Şifreniz: **123456** yaz enter'a bas aşağıdaki sonuç gelir.

Hoşgeliniz

Kullanıcı Adınız: **admin** yaz enter'a bas aşağıdaki soru gelir.

Şifreniz: **9999** yaz enter'a bas aşağıdaki sonuç gelir.

Hata

Not: Sadece şifre hatalı olmasına rağmen **& (ve)** olduğu için yanlış sonuç verdi.

Örnek

```
static void Main(string[] args)
{
    string kullanıcı, sifre;
    Console.Write("Kullanıcı Adınız: ");
    kullanıcı = Console.ReadLine();
    Console.Write("Şifreniz: ");
    sifre = Console.ReadLine();
    if (kullanıcı == "admin" | sifre == "123456")
    {
        Console.Write("Hoşgeliniz");
    }
    else
    {
        Console.Write("Hata");
    }
    Console.Read();
}
```

Kullanıcı Adınız: **admin** yaz enter'a bas aşağıdaki soru gelir.

Şifreniz: **7777** yaz enter'a bas aşağıdaki sonuç gelir.

Hoşgeliniz

Not: Şifre hatalı olmasına rağmen **| (veya)** olduğu için doğru sonuç verdi.

Örnek

```
static void Main(string[] args)
{
    int s1, s2, s3, ort;
    Console.Write("Sınav 1: ");
    s1 = Convert.ToInt16(Console.ReadLine());
    Console.Write("Sınav 2: ");
    s2 = Convert.ToInt16(Console.ReadLine());
    Console.Write("Sınav 3: ");
    s3 = Convert.ToInt16(Console.ReadLine());
    ort = (s1 + s2 + s3) / 3;
    Console.Write("Ortalamanız: " + ort + " ");
    if (ort <= 49)
    {
        Console.Write("Durum: Vasat");
    }
    if (ort >= 50 & ort <= 65)
    {
        Console.Write("Durum: Orta");
    }
    if (ort >= 66 & ort <= 79)
    {
        Console.Write("Durum: İyi");
    }
    if (ort >= 80)
    {
        Console.Write("Durum: Çok iyi");
    }
    Console.Read();
}
```

Sınav 1: 15 yaz enter'a bas aşağıdaki soru gelir.

Sınav 2: 32 yaz enter'a bas aşağıdaki soru gelir.

Sınav 3: 20 yaz enter'a bas aşağıdaki sonuç gelir.

Ortalamanız: 22 Durum: Vasat

Sınav 1: 65 yaz enter'a bas aşağıdaki soru gelir.

Sınav 2: 45 yaz enter'a bas aşağıdaki soru gelir.

Sınav 3: 90 yaz enter'a bas aşağıdaki sonuç gelir.

Ortalamanız: 66 Durum: İyi

Örnek

```
static void Main(string[] args)
{
    char karakter;
    Console.Write("Karakteri giriniz: ");
    karakter = Convert.ToChar(Console.ReadLine());
    if (karakter != 'a')
    {
        Console.Write("a harfi girmediniz tebrikler");
    }
    else
    {
        Console.Write("a harfi girdiniz maalesef hata");
    }
    Console.Read();
}
```

Karakteri giriniz: s yaz enter'a bas aşağıdaki sonuç gelir.

a harfi girmediniz tebrikler

Karakteri giriniz: a yaz enter'a bas aşağıdaki sonuç gelir.

a harfi girdiniz maalesef hata

Bilgi yarışması projesi

Kurallar;

Toplam soru sayısı: 3

Her soruda şık sayısı: 4

Diğer soruya geçebilmek için doğru cevap vermek gerekiyor.

Yanlış cevap verilince yarışma sona erer.

3 sorunun tamamı doğru cevaplanırsa oyun biter.

Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("Turkcell Bilgi Yarışmasına Hoşgeldiniz");

    Console.WriteLine();

    Console.WriteLine("-----");

    Console.WriteLine();

    int soru = 1;

    string cevap;

    if (soru == 1)
    {
        Console.WriteLine("Türkiye'nin başkenti neresidir ?");

        Console.WriteLine();

        Console.WriteLine("A) İstanbul");

        Console.WriteLine("B) Ankara");

        Console.WriteLine("C) İzmir");

        Console.WriteLine("D) Bursa");

        Console.WriteLine();

        Console.Write("Cevabınız: ");

        cevap = Console.ReadLine(); (klavyeden girdiğimiz değeri cevap değişkenine atar)
```

```
if (cevap == "b" || cevap == "B")
{
    soru = soru + 1;
}

else
{
    Console.WriteLine("Cevap yanlış toplam puanınız: 0");
}

}

if (soru == 2)
{
    Console.WriteLine("Cumhuriyet kaç yılında ilan edilmiştir ?");
    Console.WriteLine();
    Console.WriteLine("A) 1920");
    Console.WriteLine("B) 1921");
    Console.WriteLine("C) 1922");
    Console.WriteLine("D) 1923");
    Console.WriteLine();
    Console.Write("Cevabınız: ");
    cevap = Console.ReadLine(); (klavyeden girdiğimiz değeri cevap değişkenine atar)
    if (cevap == "d" || cevap == "D")
    {
        soru = soru + 1;
    }
}
```

else

```
{  
Console.Write("Cevabınız yanlış toplam puanını 50");  
}  
}
```

if (soru == 3)

```
{  
Console.WriteLine("İstanbul hangi coğrafi bölgemizedir ?");  
Console.WriteLine();  
Console.WriteLine("A) Marmara");  
Console.WriteLine("B) Akdeniz");  
Console.WriteLine("C) Karadeniz");  
Console.WriteLine("D) Ege");  
Console.WriteLine();  
Console.Write("Cevabınız: ");  
cevap = Console.ReadLine(); (klavyeden girdiğimiz değeri cevap değişkenine atar)  
if (cevap == "a" || cevap == "A")
```

```
{  
Console.Write("Tebrikler yarışma bitti, bütün soruları doğru bildiniz.");  
}
```

else

```
{  
Console.Write("Cevabınız yanlış toplam puanını 50");  
}  
Console.Read();  
}
```

Turkcell Bilgi Yarışmasına Hoşgeldiniz

Türkiye'nin başkenti neresidir ?

A) İstanbul

B) Ankara

C) İzmir

D) Bursa

Cevabınız: B yaz enter'a bas aşağıdaki soru gelir.

Cumhuriyet kaç yılında ilan edilmiştir ?

A) 1920

B) 1921

C) 1922

D) 1923

Cevabınız: D yaz enter'a bas aşağıdaki soru gelir.

İstanbul hangi coğrafi bölgemizedir ?

A) Marmara

B) Akdeniz

C) Karadeniz

D) Ege

Cevabınız: A yaz enter'a bas aşağıdaki sonuç gelir.

Tebrikler yarışma bitti, bütün soruları doğru bildiniz.

Switch Case, dallanmanın fazla olduğu durumlarda kullanılan karar yapısı alt başlığıdır.

Şehir – plakalar

Ülkeler – başkentler

Switch (değer)

```
{  
Case1: işlemler  
Break;  
Case2: işlemler  
Break;  
....  
Default: işlemler  
Break;  
}
```

Örnek

```
static void Main(string[] args)  
{  
    byte plaka;  
    Console.Write("Lütfen plakayı giriniz: ");  
    plaka = byte.Parse(Console.ReadLine());  
    switch (plaka)  
    {  
        case 1: Console.Write("Merhaba Adana"); break;  
        case 2: Console.Write("Merhaba Adıyaman"); break;  
        case 3: Console.Write("Merhaba Afyon"); break;  
        default: Console.Write("Henüz bu şehir bilgisi girilmedi"); break;  
    }  
    Console.Read();  
}
```

Lütfen plakayı giriniz: 2 yaz enter'a bas aşağıdaki sonuç gelir.

Merhaba Adıyaman

Lütfen plakayı giriniz: 44 yaz enter'a bas aşağıdaki sonuç gelir.

Henüz bu şehir bilgisi girilmedi

Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("String Değişkenler ile Switch Case Mevsim Uygulaması");
    Console.WriteLine();
    string mevsim;
    Console.Write("Lütfen mevsimi giriniz: ");
    mevsim = Console.ReadLine(); (klavyeden girdiğimiz değeri cevap değişkenine atar)
    switch (mevsim)
    {
        case "yaz": Console.WriteLine("Haziran - Temmuz - Ağustos"); break;
        case "ilkbahar": Console.WriteLine("Mart - Nisan - Mayıs"); break;
        case "sonbahar": Console.WriteLine("Eylül - Ekim - Kasım"); break;
        case "Kış": Console.WriteLine("Aralık - Ocak - Şubat"); break;
        default: Console.WriteLine("hatalı mevsim girişi"); break;
    }
    Console.Read();
}
```

String Değişkenler ile Switch Case Mevsim Uygulaması

Lütfen mevsimi giriniz: **ilkbahar** yaz enter'a bas aşağıdaki sonuç gelir.

Mart - Nisan – Mayıs

Lütfen mevsimi giriniz: **A** yaz enter'a bas aşağıdaki sonuç gelir.

hatalı mevsim girişi

Döngü, belirtilen işlemlerin belli şartlar aralığında tekrar tekrar olmasıdır.

Ekrana 100 defa merhaba dünya yazdırmak.

1000 tane personel için numara atamak.

Döngü türleri; **for, while, do-while, foreach**

1. for döngüsünün syntax yapısı;

for(başlangıç;bitiş;miktar)

```
{  
işlemler  
}
```

Örnek

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

```
{
```

```
    int i;
```

```
    for (i = 1; i <= 10; i++)
```

```
    {
```

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

```
    }
```

```
    Console.Read();
```

```
}
```

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Merhaba Dünya

Örnek

```
static void Main(string[] args)
{
    int i;
    for (i = 1; i < 10; i++)
    {
        Console.WriteLine(i);
    }
    Console.Read();
}
```

1

2

3

4

5

6

7

8

9

Örnek

```
static void Main(string[] args)
{
    int i;
    for (i = 0; i < 10; i=i+2)
    {
        Console.WriteLine(i);
    }
    Console.Read();
}
```

0

2

4

6

8

Örnek

```
static void Main(string[] args)
{
    int sayi = 24 % 9;
    Console.Write(sayi);
    Console.Read();
}
6 (kalan)
```

Örnek

```
static void Main(string[] args)
{
    int sayi;

    Console.Write("Sayıyı girin: ");

    sayi = Convert.ToInt16(Console.ReadLine());

    if (sayi % 2 == 0)
    {
        Console.Write("Sayı çifttir");
    }

    Else
    {
        Console.Write("Sayı tektir");
    }

    Console.Read();
}
```

Sayıyı girin: **10** yaz enter'a bas aşağıdaki sonuç gelir.

Sayı çifttir

Sayıyı girin: **5** yaz enter'a bas aşağıdaki sonuç gelir.

Sayı tektir

Örnek (yalnızca 3'e bölünebilen sayıları getireceğin bir örnek)

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

```
{
```

```
    for (int i = 1; i <= 10; i++)
```

```
    {
```

```
        if (i % 3 == 0)
```

```
        {
```

```
            Console.WriteLine(i);
```

```
        }
```

```
    }
```

```
    Console.Read();
```

```
}
```

3

6

9

Örnek

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

```
{
```

```
    //tam bölenleri bulma
```

```
    int sayi;
```

```
    Console.Write("Sayıyı giriniz: ");
```

```
    sayi = Convert.ToInt16(Console.ReadLine());
```

```
    for (int i = 1; i <= sayi; i++)
```

```
    {
```

```
        if (sayi % i == 0)
```

```
        {
```

```
            Console.WriteLine(i);
```

```
        }
```

```
    }
```

```
    Console.Read();
```

```
}
```

Sayıyı giriniz: 10 yaz enter'a bas aşağıdaki sonuç gelir.

1

2

5

10

Örnek

```
static void Main(string[] args)
{
    int toplam = 0;
    // 1 2 3 4 5
    // 0 1 3 6 10 15 --> toplam=toplam + i
    for (int i = 1; i <= 5; i++)
    {
        toplam = toplam + i;
        Console.WriteLine(toplam);
    }
    Console.Read();
}
```

1
3
6
10
15

Örnek

```
static void Main(string[] args)
{
    int toplam = 0;
    // 1 2 3 4 5
    // 0 1 3 6 10 15 --> toplam=toplam + i
    for (int i = 1; i <= 10; i++)
    {
        toplam = toplam + i;
    }
    Console.WriteLine(toplam);
    Console.Read();
}
```

55

NOT = İki örnek arasındaki farka iyi bak, 1.sinde sayıları toplayarak yazdı
2.sinde ise direk sayıların toplamını yazdı.

Örnek

```
static void Main(string[] args)
{
    //5 faktöriyel
    int faktoriyel = 1;
    for (int i = 5; i >= 1; i--)
    {
        faktoriyel = faktoriyel * i;
        Console.WriteLine(faktoriyel);
    }
    Console.Read();
}
```

5
20
60
120
120

Soru: Klavyeden girilen sayının faktöriyelini while döngüsü ile hesaplayan kodu yazınız.

Çözüm:

```
static void Main(string[] args)
{
    int sayi ;
    int faktoriyel = 1 ;
    Console.Write("Sayıyı Girin: ") ;
    sayi = Convert.ToInt16(Console.ReadLine()) ;
    while (sayi >= 1)
    {
        faktoriyel = faktoriyel * sayi;
        sayi - - ;
    }
    Console.Write("Sonuç: " + faktoriyel) ;
    Console.Read() ;
}
```

Sayıyı Girin: 5 yaz enter'a bas aşağıdaki sonuç gelir.

Sonuç: 120

While döngüsü syntax yapısı;

While (şart)

```
{  
işlemler  
}
```

Örnek

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

```
{
```

```
    int sayac = 1;
```

```
    while (sayac <= 10)
```

```
    {
```

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

```
        sayac++;
```

```
    }
```

```
    Console.Read();
```

```
}
```

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Merhaba Turkcell Videolari

Örnek

```
static void Main(string[] args)
{
    int sayi = 1;
    while (sayi <= 5)
    {
        Console.WriteLine(sayi);
        sayi++;
    }
    Console.Read();
}
1
2
3
4
5
```

Örnek

```
static void Main(string[] args)
{
    int sayi = 1;
    int toplam = 0;
    while (sayi <= 5)
    {
        toplam = toplam + sayi;
        sayi++;
    }
    Console.Write(toplam);
    Console.Read();
}
15
```

do-while döngüsü syntax yapısı;

do

{

işlemler

}

while (şart);

Örnek

static void Main(string[] args)

{

int sayi = 1;

do

{

Console.WriteLine(sayi);

sayi++;

}

while (sayi <= 10);

Console.Read();

}

1

2

3

4

5

6

7

8

9

10

Dizi, aynı türdeki verilerin bir araya gelerek oluşturdukları kümelerdir.

Programlama literatüründe **array** olarak adlandırılmaktadır.

Dizilerde her eleman **data**, her elemanın konumu **index** olarak adlandırılır.

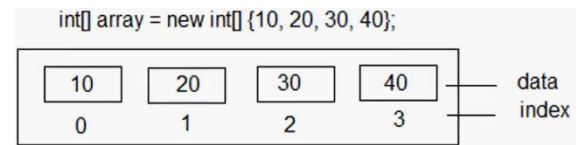
Dizilerde sayma işi 0'dan başlar.

Diziler ram bellekte tutulurlar.

Diziler tanımlama işlemleri **[]** sembolü ile yapılır.

Dizi tanımlamalarında elemanlar **{}** sembolü içine yazılır.

Dizi tanımlamalarında ilgili dizinin mutlaka bir değişken türü bulunur.



Örnek

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

```
{
```

```
    string[] sehirler = { "İstanbul", "Ankara", "İzmir", "Bursa" };
```

```
    Console.WriteLine("Dizimizin 2.Index Değeri: " + sehirler[2]);
```

```
    Console.Read();
```

```
}
```

Dizimizin 2.Index Değeri: İzmir

Örnek

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

```
{
```

```
    string[] sehirler = { "İstanbul", "Ankara", "İzmir", "Bursa" };
```

```
    Console.WriteLine(sehirler[0]);
```

```
    Console.Read();
```

```
}
```

İstanbul

Örnek

```
static void Main(string[] args)
{
    //Int değişkenler ile dizi kullanımları
    Console.WriteLine("Int değişkenler ile dizi kullanımları");
    Console.WriteLine();
    int[] sayilar = {1,2,3,4,5,6,7};
    Console.WriteLine(sayilar[4]);
    Console.Read();
}
```

Int değişkenler ile dizi kullanımları

5

Örnek

```
static void Main(string[] args)
{
    //Int değişkenler ile dizi kullanımları
    Console.WriteLine("Int değişkenler ile dizi kullanımları");
    Console.WriteLine();
    int[] sayilar = {1,2,3,4,5,6,7};
    for (int i = 0; i < 7; i++)
    {
        Console.WriteLine(sayilar[i]);
    }
    Console.Read();
}
```

Int değişkenler ile dizi kullanımları

1

2

3

4

5

6

7

Örnek

```
static void Main(string[] args)
{
    //dizilerle beraber karar yapısı kullanımı
    int[] sayilar = {10,20,30,40,50};
    for (int i = 0; i < sayilar.Length; i++)
    {
        Console.WriteLine(sayilar[i]);
    }
    Console.Read();
}
```

10
20
30
40
50

Örnek

```
static void Main(string[] args)
{
    //dizilerle beraber karar yapısı kullanımı
    int[] sayilar = {10,20,30,40,50};
    for (int i = 0; i < sayilar.Length; i++)
    {
        if (sayilar[i] % 20 == 0)
        {
            Console.WriteLine(sayilar[i]);
        }
    }
    Console.Read();
}
```

20
40

Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("Length Kullanımı");
    Console.WriteLine();
    string[] kisiler = { "Ali", "Veli", "Ayşe" };
    for (int i = 0; i < 5; i++)
    {
        Console.WriteLine(kisiler[i]);
    }
    Console.Read();
}
```

Length Kullanımı

Ali

Veli

Ayşe

Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("Length Kullanımı");
    Console.WriteLine();
    string[] kisiler = { "Ali", "Veli", "Ayşe", "Ahmet", "Eylül" };
    for (int i = 0; i < kisiler.Length; i++)
    {
        Console.WriteLine(kisiler[i]);
    }
    Console.Read();
}
```

Length Kullanımı

Ali

Veli

Ayşe

Dizilerde toplama işlemi;

Toplama işlemi += yaklaşımla gerçekleştirilir.

Başlangıçta ilk değeri 0 olan bir toplam değişkeni tanımlanır.

Toplam değişkeninin son değerinin üzerine ilgili indexte bulunan değer eklenir.

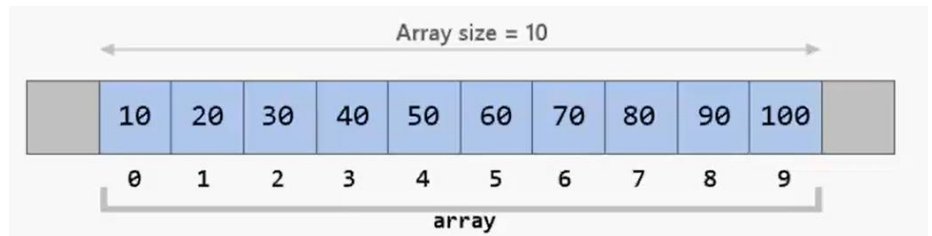
Başlangıç: toplam = 0

Adım 1: $0 + 10 = 10$

Adım 2: $10 + 20 = 30$

Adım 3: $30 + 30 = 60$

Adım 4: $60 + 40 = 100$



Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("Dizilerde toplama işlemi örneği");
    Console.WriteLine();
    int[] sayilar = { 10, 20, 30, 40, 50, 60, 70, 80, 90 };
    int toplam = 0;
    for (int i = 0; i < sayilar.Length; i++)
    {
        toplam += sayilar[i];
        Console.WriteLine(toplam);
    }
    Console.Read();
}
```

Dizilerde toplama işlemi örneği

10

30

60

100

150

210

280

360

450

Örnek

```
static void Main(string[] args)
{
    Console.WriteLine("Dizilerde toplama işlemi örneği");
    Console.WriteLine();
    int[] sayilar = { 10, 20, 30, 40, 50, 60, 70, 80, 90 };
    int toplam = 0;
    for (int i = 0; i < sayilar.Length; i++)
    {
        toplam += sayilar[i];
        Console.WriteLine(toplam);
    }
    Console.Read();
}
```

Dizilerde toplama işlemi örneği

Dizi metotları,

Short: diziyi küçükten büyüğe sıralar.

Reverse: diziyi tersten yazdırır.

Index of: aranan değerin index sırasını döndürür.

Max: dizideki en büyük elemanı verir.

Min: dizideki en küçük elemanı verir.

Length: dizinin boyutunu verir.

Örnek

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

```
{
```

```
//Short Metodu
```

```
int[] sayilar = { 20, 11, 16, 32 };
```

```
Array.Sort(sayilar);
```

```
for (int i = 0; i < sayilar.Length; i++)
```

```
{
```

```
    Console.WriteLine(sayilar[i]);
```

```
}
```

```
Console.Read();
```

```
}
```

11

16

20

32 (Şehirler küçükten büyüğe doğru sıralandı)

Örnek

```
static void Main(string[] args)
{
    //Reverse Metodu
    string[] sehirler = { "Malatya", "Kütahya", "Uşak", "Bursa" };
    Array.Reverse(sehirler);
    for (int i = 0; i < sehirler.Length; i++)
    {
        Console.WriteLine(sehirler[i]);
    }
    Console.Read();
}
```

Bursa

Uşak

Kütahya

Malatya

(Şehirler tersten yazıldı)

Örnek

```
static void Main(string[] args)
{
    string[] sehirler = { "Malatya", "Kütahya", "Uşak", "Bursa" };
    Array.Reverse(sehirler);
    for (int i = 0; i < sehirler.Length; i++)
    {
        Console.WriteLine(sehirler[i]);
    }
    Console.Read();
}
```

Bursa

Kütahya

Malatya

Uşak

(Şehirler alfabetik sıraya göre sıralandı)

Örnek

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

```
{
```

```
//Index Of Metodu
```

```
string[] kisiler = { "Buse", "Ali", "Eda", "Salih" };
```

```
int sir;
```

```
sira = Array.IndexOf(kisiler, "Eda");
```

```
Console.Write(sira);
```

```
Console.Read();
```

```
}
```

2

Örnek

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

```
{
```

```
//Index Of Metodu
```

```
string[] kisiler = { "Buse", "Ali", "Eda", "Salih" };
```

```
int sir;
```

```
Array.Sort(kisiler);
```

```
sira = Array.IndexOf(kisiler, "Salih");
```

```
Console.Write(sira);
```

```
Console.Read();
```

```
}
```

3

Örnek

```
static void Main(string[] args)
{
    //Min Max Metotları
    int[] sayilar = { 76, 43, 12, 56, 34 };
    Console.Write(sayilar.Min());
    Console.Read();
}
12
```

Örnek

```
static void Main(string[] args)
{
    //Min Max Metotları
    int[] sayilar = { 76, 43, 12, 56, 34 };
    Console.Write(sayilar.Max());
    Console.Read();
}
76
```

Foreach döngüsü, dizilerle beraber kullanılan döngü türüdür.

4 temel parametresi vardır.

1: değişken türü

2: değişken adı

3: in(içinden) komutu

4: dizi adı

Örnek

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

```
{
```

```
string[] sehirler = { "Ankara", "Adana", "Bursa", "İzmir" };
```

```
foreach (string s in sehirler)
```

```
{
```

```
    Console.WriteLine(s);
```

```
}
```

```
Console.Read();
```

```
}
```

Ankara

Adana

Bursa

İzmir

Örnek

```
static void Main(string[] args)
{
    //Foreach ve Aritmetik İşlemler
    int[] sayilar = { 23, 55, 32, 16, 89, 70 };
    int toplam = 0;
    foreach (int x in sayilar)
    {
        toplam = toplam + x;
    }
    Console.WriteLine("Toplam: " + toplam);
    Console.Read();
}
```

Toplam: 285

Örnek

```
static void Main(string[] args)
{
    int[] sayilar = { 34, 22, 11, 67, 89, 50 };
    foreach (int sayi in sayilar)
    {
        if (sayi % 2 == 0)
        {
            Console.WriteLine(sayi);
        }
    }
    Console.Read();
}
```

34

22

50

Dosya işlemleri, çeşitli belge türlerinin (metin belgesi, word, excel vs.) C# kodları üzerinden oluşturulmasıdır.

Dosya işlemleri sayesinde kodlar üzerinden metin belgesi oluşturabilir, yazabilir, silebilir veya değiştirebiliriz.

Dosya işlemleri aslında bir veri tabanıdır.

Kütüphane ekleme işlemleri, dosya işlemleri için kullanılacak kütüphane: System.OP

Not: Geriye değer döndürmeyecek dediğimizde **'void'** kullanırız.

“public static void “ ile metod oluşturuyoruz dikkat !!!!!

Örnek

public static void yazdir()

```
{  
    Console.Write("Bu bir metotdur");  
    Console.WriteLine();  
    Console.Write("Burası metodun bir başka satırıdır");  
    Console.WriteLine();  
}
```

```
static void Main(string[] args)  
{  
    yazdir();  
    Console.Read();  
}
```

Bu bir metotdur

Burası metodun bir başka satırıdır

Örnek

public static void toplamametodu()

```
{  
    int sayi1 = 24, sayi2 = 30;  
    int toplam = sayi1 + sayi2;  
    Console.Write(toplam);  
}
```

```
static void Main(string[] args)  
{  
    toplamametodu();  
    Console.Read();  
}
```

54

Örnek

public static void ardisiksayilar()

```
{  
    for (int i = 1; i <= 5; i++)  
    {  
        Console.Write(i + " ");  
    }  
}
```

```
static void Main(string[] args)  
{  
    ardisiksayilar();  
    Console.Read();  
}
```

1

2

3

4

5

//Void metotlarda parametre kullanımı

Örnek

```
public static void Metinyaz(string p)
```

```
{  
    Console.Write(p);  
}  
  
static void Main(string[] args)  
{  
    Metinyaz("Herkes merhaba")  
    Console.Read();  
}
```

Herkes merhaba

Örnek

```
public static void Metinyaz2(string parametre)
```

```
{  
    Console.Write(parametre);  
}  
  
static void Main(string[] args)  
{  
    Console.Write("Kelimeyi giriniz: ");  
    string kelime = Console.ReadLine();  
    Metinyaz2(kelime);  
    Console.Read();  
}
```

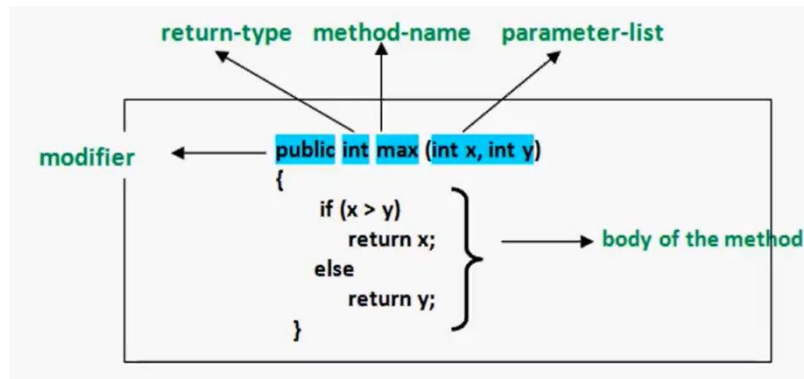
Kelimeyi giriniz: İstanbul yaz enter'a bas aşağıdaki sonuç gelir.
İstanbul

Geriye değer döndüren metotlar, geriye bir dönüş tipi olarak oluşturulan metotlardır.

Geriye değer döndürmeyen metotlarda **void** tanımlaması yapılır.

Geriye değer döndüren metotlarda ise **int**, **string** veya sınıfa ait ilgili değer türü yazılır.

Geriye dönecek değer **return** ifadesi ile belirlenir.



Örnek

public static int toplama()

```
{  
    int sayi1, sayi2, sonuc;  
    sayi1 = 25;  
    sayi2 = 35;  
    sonuc=sayi1+sayi2;  
    return sonuc;  
}  
  
static void Main(string[] args)  
{  
    Console.Write(toplama());  
    Console.Read();  
}
```

//Parametrelili geriye değeri döndüren metod

Örnek

```
public static int toplam(int s1,int s2)
{
    int sonuc;
    sonuc = s1 + s2;
    return sonuc;
}
static void Main(string[] args)
{
    Console.WriteLine("Toplam: " + toplam(10, 15));
    Console.ReadLine();
}
```

Toplam: 25

Örnek

```
public static int toplam(int s1, int s2)
{
    int sonuc;
    sonuc = s1 + s2;
    return sonuc;
}
static void Main(string[] args)
{
    int sayi1, sayi2;
    Console.WriteLine("1.Sayıyı giriniz: ");
    sayi1 =Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("2.Sayıyı giriniz: ");
    sayi2 = Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("Sonuç: " + toplam(sayi1, sayi2));
    Console.ReadLine();
}
```

1.Sayıyı giriniz: 22

2.Sayıyı giriniz: 44

Sonuç: 66

Sınıf, niteliklerimizi barındıran ve kod bloklarını bir arada tutan yapılardır.

Nesne, sınıfların niteliklerine ulaşabilmemiz için gereken komutlardır.

Nesne türetme işlemi, **SınıfAdı NesneAdı = new SınıfAdı();**

Önce Araba adlı sınıf oluşturup bu adımları yaptın

```
internal class Araba
```

```
{  
    public string marka;  
    public string model;  
    public int hiz;  
    public double motor;  
    public string renk;
```

Sonra bu adımları yaptın

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

```
{  
    Araba ar=new Araba();  
    ar.marka = "Marka x";  
    ar.hiz = 180;  
    ar.model = "2021";  
    ar.motor = 1.6;  
    ar.renk = "Beyaz";  
  
    Console.WriteLine("** Araba Tanıtım Kartı 1 **");  
    Console.WriteLine();  
    Console.WriteLine("Marka: " + ar.marka);  
    Console.WriteLine("Hız: " + ar.hiz);  
    Console.WriteLine("Model: " + ar.model);  
    Console.WriteLine("Motor: " + ar.motor);  
    Console.WriteLine("Renk: " + ar.renk);  
    Console.Read();  
}
```

**** Araba Tanıtım Kartı 1 ****

Marka: Marka x

Hız: 180

Model: 2021

Motor: 1,6

Renk: Beyaz

Kalıtım, projelerimizi gerçekleştirirken bir sınıfa ait değişkenlerin birkaçını ya da tamamını bir başka sınıf içinde kullanmaktır.

Kuşlar sınıfı → Kartal, şahin, papağan, baykuş → Papağan: sultan, cennet, jako, amazon

Önce Kuşlar sınıfını oluşturdun.

internal class Kuşlar

```
{  
    public string tur;  
    public string ses;  
    public int hiz;  
    public double agirlik;  
}
```

Sonra Papagan sınıfını oluşturdun ve Papagan sınıfına Kuşlar sınıfından bilgiler çekmek için Papagan sınıfının yanına Kuşlar'ı ekledin.

internal class Papagan:Kuşlar

```
{  
    public string isim;  
    public string renk;  
}
```

Sonuç:

```
static void Main(string[] args)  
{  
    Papagan p = new Papagan();  
    p.tur = "Papağan";  
    p.hiz = 50;  
    p.isim = "Sultan";  
    p.renk = "Sarı - Kırmızı";  
    p.ses = "Cik";  
    p.agirlik = 1650;  
  
    Console.WriteLine("Tür: " + p.tur);  
    Console.WriteLine("İsim: " + p.isim);  
    Console.WriteLine("Hız: " + p.hiz);  
    Console.WriteLine("Ağırlık: " + p.agirlik);  
    Console.WriteLine("Renk: " + p.renk);  
    Console.WriteLine("Ses: " + p.ses);  
    Console.Read();  
}
```

Tür: Papağan

İsim: Sultan

Hız: 50

Ağırlık: 1650

Renk: Sarı – Kırmızı

Ses: Cik

Çok biçimlilik, miras alma işleminden sonra herhangi bir alanda bulunan değerleri miras alınan sınıftaki haliyle değil de bizim istediğimiz formatta kullanım sağlamasıdır.

Virtual, kalıtım alınan sınıflarda içeriğin değiştirilebilmesi için kullanılan komutlardır.

Override, geçersiz kılma anlamına bu komut değiştirilecek metod yazılmadan önce metodun başına eklenir.

Önce Kuslar sınıfını oluşturdu.

internal class Kuslar

```
{
    public string tur;
    public string ses;
    public int hiz;
    public double agirlik;

    public virtual string sescikar()
    {
        return "buraya ses yazılacak";
    }
}
```

Sonra Papagan sınıfını oluşturdu ve Papagan sınıfına Kuslar sınıfından bilgiler çekmek için Papagan sınıfının yanına Kuslar'ı ekledi.

internal class Papagan:Kuslar

```
{
    public string isim;
    public string renk;

    public override string sescikar()
    {
        return "cik cik";
    }
}
```

Sonra Karga sınıfını oluşturdu ve Karga sınıfına Kuslar sınıfından bilgiler çekmek için Karga sınıfının yanına Kuslar'ı ekledi.

internal class Karga:Kuslar

```
{
    public string isim;
    public string renk;

    public override string sescikar()
    {
        return "gak gak";
    }
}
```

Not: Her bir kuş sesi farklı olduğu için her kuş sınıfında farklı bir ses kodladın.

Sonuç:

```
static void Main(string[] args)
{
    Papagan p = new Papagan();
    p.tur = "Papağan";
    p.hiz = 50;
    p.isim = "Sultan";
    p.renk = "Sarı - Kırmızı";
    p.agirlik = 1650;
    p.sescikar();

    Console.WriteLine("Tür: " + p.tur);
    Console.WriteLine("İsim: " + p.isim);
    Console.WriteLine("Hız: " + p.hiz);
    Console.WriteLine("Ağırlık: " + p.agirlik);
    Console.WriteLine("Renk: " + p.renk);
    Console.WriteLine("Ses: " + p.sescikar());

    Console.WriteLine();
    Karga k = new Karga();

    k.agirlik = 1350;
    k.hiz = 80;
    k.isim = "Alacakarga";
    k.renk = "Siyah";
    k.tur = "Karga";
    k.sescikar();

    Console.WriteLine("Ağırlık: " + k.agirlik);
    Console.WriteLine("Hız: " + k.hiz);
    Console.WriteLine("İsim: " + k.isim);
    Console.WriteLine("Renk: " + k.renk);
    Console.WriteLine("Tür: " + k.tur);
    Console.WriteLine("Ses: " + k.sescikar());
    Console.Read();
}
```

Tür: Papağan

İsim: Sultan

Hız: 50

Ağırlık: 1650

Renk: Sarı – Kırmızı

Ses: cik cik

Ağırlık: 1350

Hız: 80

İsim: Alacakarga

Renk: Siyah

Tür: Karga

Ses: gak gak

Hazır fonksiyonlar, dizilerde bulunan sort, reverse, min, max ve benzeri metotların matematiksel, metinsel veya tarih zaman işlemlerinde kullanılması için C# tarafında bize hazır olarak sunulan bazı fonksiyonlar / metotlardır.

Matematiksel fonksiyonlar;

Abs: mutlak

Ceiling: üst tabana yuvarla

Floor: alt tabana yuvarla

Sqrt: karekök

Pow: üs alma

Pi: pi sayısı

Örnek

```
static void Main(string[] args)
{
    //Math
    double sayi;
    Console.Write("Sayıyı giriniz: ");
    sayi = Convert.ToDouble(Console.ReadLine());
    Console.WriteLine("Mutlak değer: " + Math.Abs(sayi));
    Console.WriteLine("Üst taban: " + Math.Ceiling(sayi));
    Console.WriteLine("Alt taban: " + Math.Floor(sayi));
    Console.WriteLine("Karekök: " + Math.Sqrt(sayi));
    Console.Read();
}
```

Sayıyı giriniz: -3,68

Mutlak değer: 3,68

Üst taban: -3

Alt taban: -4

Karekök: NaN

String fonksiyonlar, metinsel ifadelerde işlemler yapabilmek için kullanılan fonksiyonlardır.

Concat: birleştirme

Length: uzunluk

Index of: metin arama

Starwith: ilgili parametre ile mi başlıyor

Trim: metnin başındaki ve sonundaki boşlukları kaldırma

To Upper: büyük harf

To Lower: küçük harf

Remove: istenilen karakterden itibaren siler

Replace: değiştirme

Substring: istenilen karakterden itibaren işlem yapar

Örnek

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

```
{  
    string metin1, metin2;  
    Console.Write("Metin 1'i giriniz: ");  
    metin1 = Console.ReadLine();  
    Console.Write("Metin 2'yi giriniz: ");  
    metin2 = Console.ReadLine();  
  
    Console.WriteLine("Concat ile birleştirme: " + string.Concat(metin1, metin2));  
    Console.WriteLine("Metin 1 için karakter sayısı: " + metin1.Length);  
    Console.WriteLine("Index Of Örneği: " + metin1.IndexOf("ay"));  
    Console.WriteLine("Startswith Örneği: " + metin1.StartsWith("Günaydın"));  
    Console.WriteLine("Büyük harf: " + metin1.ToUpper());  
    Console.WriteLine("Küçük harf: " + metin1.ToLower());  
    Console.WriteLine("Remove metodu: " + metin1.Remove(4));  
    Console.WriteLine("Replace fonksiyonu: " + metin1.Replace("a" , "A"));  
    Console.WriteLine("Substring fonksiyonu: " + metin1.Substring(4));  
    Console.ReadLine();  
}
```

Metin 1'i giriniz: Günaydın yaz enter'a bas aşağıdaki soru gelecek

Metin 2'yi giriniz: Bursa yaz enter'a bas aşağıdaki sonuçlar gelecek

Concat ile birleştirme: GünaydınBursa

Metin 1 için karakter sayısı: 8

Index Of Örneği: 3

Startswith Örneği: True

Büyük harf: GÜNAYDIN

Küçük harf: günaydın

Remove metodu: Güna

Replace fonksiyonu: GünAydın

Substring fonksiyonu: ydın

Tarih zaman (datetime) fonksiyonları, tarih ve zaman türünde yapılacak işlemler için bizlere hazır olarak sunulan fonksiyonlardır.

Anahtar kelime: **Datetime**

Kullanım şekli: **Datetime.Now** yaz sonra **day, month, year, hour, second** gibi ifadeler alır.

Timespan: 2 tarih arasındaki farkı hesaplar.

Örnek

```
static void Main(string[] args)
{
    //Datetime
    Console.WriteLine("Bugünün gün bilgisi: " + DateTime.Now.Day);
    Console.WriteLine("Bugünün ay bilgisi: " + DateTime.Now.Month);
    Console.WriteLine("Bugünün yıl bilgisi: " + DateTime.Now.Year);
    Console.WriteLine("Bugünün saat bilgisi: " + DateTime.Now.Hour);
    Console.WriteLine("Bugünün dakika bilgisi: " + DateTime.Now.Minute);
    Console.WriteLine("Bugünün saniye bilgisi: " + DateTime.Now.Second);
    Console.WriteLine("Bugünün kısa tarih bilgisi: " + DateTime.Now.ToShortDateString());
    Console.WriteLine("Bugünün kısa tarih bilgisi: " + DateTime.Now.ToLongDateString());
    Console.Read();
}
```

Bugünün gün bilgisi: 22

Bugünün ay bilgisi: 7

Bugünün yıl bilgisi: 2022

Bugünün saat bilgisi: 23

Bugünün dakika bilgisi: 29

Bugünün saniye bilgisi: 11

Bugünün kısa tarih bilgisi: 22.07.2022

Bugünün kısa tarih bilgisi: 22 Temmuz 2022 Cuma

Random, kod kısmında belirlenen aralıklarda rastgele olarak tam sayı üreten sınıftır.

Random nesne_adı = new Random();

Değer aralığı: **Next**

Örnek

```
static void Main(string[] args)
{
    //Random sınıfı uygulaması
    int sayi;
    Random r = new Random();
    sayi = r.Next(0,51);
    Console.Write(sayi);
    Console.Read();
}
```

24

Not: Random aralığında 1.sayı dahildir fakat 2.sayı dahil değildir yani 0 karşımıza gelebilir fakat 51 gelemez.

Örnek

```
static void Main(string[] args)
{
    Random rn = new Random();

    int sehir;
    string[] sehirler = { "Malatya", "Kütahya", "Bursa", "Uşak" };
    sehir = rn.Next(0, sehirler.Length);
    Console.Write(sehirler[sehir]);
    Console.Read();
}
```

Sehirler dizisi 0 1 2 3 numaralı dizilerden oluştuğu için yani dizinin toplam uzunluğu 4 tür ve sonuç olarak 0 1 2 3 sayısı gelebilirdi ama **Uşak** geldi yani 4 gelmiş demektir.

```
static void Main(string[] args)
{
    Random rastgele = new Random();
    int sayi;
    sayi = rastgele.Next(50);
    Console.WriteLine(sayi);
    Console.Read();
}
```

Örnek

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

```
{
```

```
    //Captcha
```

```
    int d1,d2,d3,d4;
```

```
    Random rnd = new Random();
```

```
    d1 = rnd.Next(0, 10);
```

```
    d2 = rnd.Next(0, 10);
```

```
    d3 = rnd.Next(0, 10);
```

```
    d4 = rnd.Next(0, 10);
```

```
    Console.WriteLine(d1);
```

```
    Console.WriteLine(d2);
```

```
    Console.WriteLine(d3);
```

```
    Console.WriteLine(d4);
```

```
    string[] karakterler = { "a", "A", "b", "B", "c", "C", "d", "D", "e", "E" };
```

```
    Console.Write(d1 + karakterler[d2] + d3 + karakterler[d4]);
```

```
    Console.Read();
```

```
}
```

8

9

4

3

8E4B

Dosya işlemleri, çeşitli belge türlerinin (metin belgesi, word, excel vs.)

C# kodları üzerinden oluşturulmasıdır.

Dosya işlemleri sayesinde kodlar üzerinden metin belgesi oluşturabilir, yazabilir, silebilir veya değiştirebiliriz.

Dosya işlemleri aslında bir veri tabanıdır.

Dosya işlemleri için kullanılacak kütüphane: **System.IO**

Örnek

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

```
    StreamWriter sw = new StreamWriter("C:\\Users\\toroman\\Desktop\\Deneme Belgesi.txt");
    Console.Read();
}
```

Sonuç olarak masaüstünde **“Deneme Belgesi”** adında metin belgesi oluştu

Örnek

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

```
    StreamWriter sw = new StreamWriter("C:\\Users\\toroman\\Desktop\\Deneme Belgesi.txt");
    sw.Write("Merhaba bu bir metin belgesidir");
    sw.Close();
    Console.Read();
}
```

Sonuç olarak masaüstünde **“Deneme Belgesi”** adında metin belgesi oluştu ve içine **“Merhaba bu bir metin belgesidir”** yazdırdık.

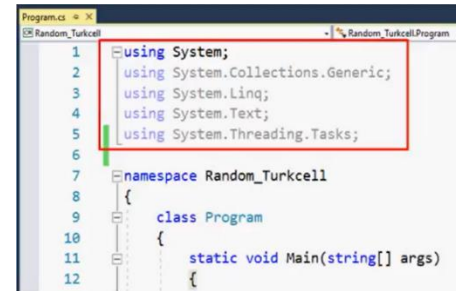
sw.Close(); komutu yazma işlemi tamamlandı artık kaydedebilirsin anlamı taşır.

Örnek

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

```
    StreamWriter sw = new StreamWriter(@"C:\Users\toroman\Desktop\Deneme Belgesi.txt");
    string metin;
    Console.Write("Metni giriniz: ");
    metin = Console.ReadLine();
    sw.Write(metin);
    sw.Close();
    Console.Read();
}
```

Sonuç olarak masaüstünde **“Deneme Belgesi”** adında metin belgesi oluştu ve ekran açıldığında **Salih Toraman** yazdım, sonuç olarak metin belgesine **Salih Toraman** yazıldı.



Örnek

```
static void Main(string[] args)
{
    Console.BackgroundColor = ConsoleColor.Blue;

    Console.WriteLine("Merhaba Dünya");

    Console.Read();
}
```

Merhaba Dünya (yazdirdığın şey mavi renkli olarak karşına çıkar)

Örnek

```
static void Main(string[] args)
{
    Console.BackgroundColor = ConsoleColor.Blue;

    Console.Clear();

    Console.WriteLine("Merhaba Dünya");

    Console.Read();
}
```

Merhaba Dünya yazar ve bütün arka plan mavi renkli olur.

Örnek

```
static void Main(string[] args)
{
    Console.BackgroundColor = ConsoleColor.Blue;

    Console.Clear();

    Console.ForegroundColor = ConsoleColor.Green;

    Console.WriteLine("Merhaba Dünya");

    Console.Read();
}
```

Merhaba Dünya yazı yeşil arka plan mavi olur.

2 Boyutlu diziler, matematikte matris olarak adlandırılırlar.

Tek boyutlu dizilerde olduğu gibi index değeri tek bir sayı değildir.

Index değerleri satır ve sütunlarla beraber anılır.

Örnek

```
static void Main(string[] args)
{
    int[,] sayilar = new int[2, 2];

    sayilar[0, 0] = 10;
    sayilar[0, 1] = 20;
    sayilar[1, 0] = 30;
    sayilar[1, 1] = 40;

    Console.Write(sayilar[0, 1]);
    Console.Read();
}
```

20

The diagram shows a 3x3 matrix with rows indexed 0, 1, 2 and columns indexed 0, 1, 2. Each cell contains a coordinate pair (row, column). A yellow box labeled 'Column Index' points to the column headers, and a yellow box labeled 'Row Index' points to the row headers.

	0	1	2
0	(0,0)	(0,1)	(0,2)
1	(1,0)	(1,1)	(1,2)
2	(2,0)	(2,1)	(2,2)

Yıldızlarla şekil oluşturma; döngüler, karar yapıları ve değişkenlerin bir arada kullanılabileceği, özellikle algoritmik mülakat ve üniversitelerin vize / final sorularında çokça karşılaşılan yapılardır.

Örnek

```
static void Main(string[] args)
{
    //Yıldızlarla şekil oluşturma
    for (int i = 1; i <= 5; i++)
    {
        Console.WriteLine("*");
    }
    Console.Read();
}
```

*
*
*
*
*

Hata yönetimi, kod kısmında yapılan işlemlerde uygulamanın hata vermesi yerine uyarı mesajları ile uygulamayı ayakta tutma işlemidir.

Try – Catch – Finally

```
Try
{
İşlemler
}
Catch
{
Hata
}
Finally
{
Çalıştır
}
```

Örnek

```
static void Main(string[] args)
{
    try
    {
        int sayi1, sayi2, sonuc;
        Console.Write("Sayı 1: ");
        sayi1 = Convert.ToInt16(Console.ReadLine());
        Console.Write("Sayı 2: ");
        sayi2 = Convert.ToInt16(Console.ReadLine());
        sonuc = sayi1 * sayi2;
        Console.Write("İşlem sonucu: " + sonuc);
    }
    catch
    {
        Console.Write("Hata var lütfen kontrol ediniz");
    }
    Console.Read();
}
```

Sayı 1: 5
Sayı 2: 10
İşlem sonucu: 50

Sayı 1: 5
Sayı 2: a
Hata var lütfen kontrol ediniz

Örnek

```
static void Main(string[] args)
{
    try
    {
        int sayi1, sayi2, sonuc;

        Console.Write("Sayı 1: ");

        sayi1 = Convert.ToInt16(Console.ReadLine());

        Console.Write("Sayı 2: ");

        sayi2 = Convert.ToInt16(Console.ReadLine());

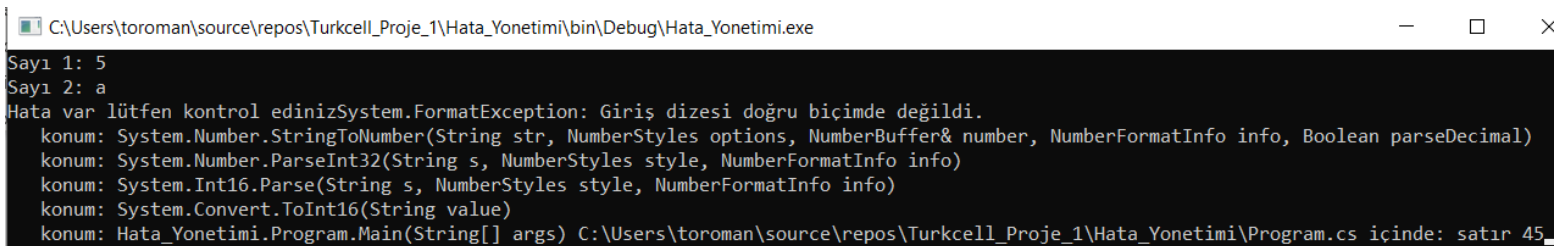
        sonuc = sayi1 * sayi2;

        Console.Write("İşlem sonucu: " + sonuc);

    }

    catch(Exception ex)
    {
        Console.Write("Hata var lütfen kontrol ediniz");
        Console.Write(ex);
    }

    Console.Read();
}
```



C:\Users\toroman\source\repos\Turkcell_Proje_1\Hata_Yonetimi\bin\Debug\Hata_Yonetimi.exe

Sayı 1: 5
Sayı 2: a
Hata var lütfen kontrol edinizSystem.FormatException: Giriş dizesi doğru biçimde değildi.
konum: System.Number.StringToNumber(String str, NumberStyles options, NumberBuffer& number, NumberFormatInfo info, Boolean parseDecimal)
konum: System.Number.ParseInt32(String s, NumberStyles style, NumberFormatInfo info)
konum: System.Int16.Parse(String s, NumberStyles style, NumberFormatInfo info)
konum: System.Convert.ToInt16(String value)
konum: Hata_Yonetimi.Program.Main(String[] args) C:\Users\toroman\source\repos\Turkcell_Proje_1\Hata_Yonetimi\Program.cs içinde: satır 45

“ **Exception ex** ” bize hata mesajını gönderdi ve hatanın nedenini açıkladı.

Invalid Cast Exception: Tür dönüştürme işlemlerinde tanımlanan değişkenlere göre farklı bir türde dönüşüm yapılmaya çalışılması durumunda karşımıza çıkan hata mesajıdır.

Index Out Of Range Exception: Dizide bulunmayan değerler ile karşılaşınca oluşan hata mesajıdır.

Divided By Zero: Sıfıra bölme işlemi yapılmak istenildiği zaman oluşan hata mesajıdır.

Format Exception: Sayısal bir alana sayısal olmayan bir değer girilmesi durumunda oluşan hata mesajıdır.

Over Flow Exception: Bir değişkenin aralıklarının dışına çıkılması durumunda karşılaşılan hata mesajıdır.

Argument Null Exception: Aritmetik bir alanın boş bırakılması durumunda karşımıza çıkan hata mesajıdır.

Arithmetic Exception: Matematiksel hatalarda oluşan hata mesajıdır.

Dosya İşlemleri

Folder browser dialog aracı, dosya giriş çıkış işlemlerinde konum seçmek için kullanılan araçtır.

Bu diyalog aracında dosya türleri gösterilmez yalnızca konum bilgisi verilir.

Open file dialog aracı, folder browser dialog aracından farklı olarak sadece klasörleri değil dosyaları da gösteren araçtır.

Bazı filtreleme yöntemleri ile sadece istenen türde dosyalar gösterilebilir.

Save file dialog aracı, yapı olarak open file dialog aracı ile benzerlik gösteren bir araçtır.

Amaç form üzerinden herhangi bir dosya türüne kayıt işlemi yapmaktır.

C# Form

Form, kullanıcılara geliştirme sürecinde arayüz tasarlama olanağı sağlayan Visual Studio geliştirme ortamlarından biridir.

C# Console geliştirme ortamında işlemler yalnızca siyah ekran üzerinden yapılmaktaydı.

C# Form ise geliştiricilere kullanıcılar için grafikler, metin giriş kutuları, veri tabanı araçları, hareketli nesneler ve görsel değeri yüksek daha pek çok arayüz nesnesi sunmaktadır.

C# Form uygulamaları aşağıdaki yerlerde kullanılabilir;

Bankacılık sektöründe

Sigorta ve Finansmanda

Ticari otomasyonlarda

Personel takip sistemlerinde

Ön muhasebe uygulamalarında

Masaüstü uygulamalarında (video kayıt, ses kayıt, video düzenleme vs.)

Veri tabanı kayıt uygulamalarında

Akıllı sistemlerde

Yapay zeka uygulamalarında

Parmak izi okuma sistemlerinde

Temel Araç Kullanımları

Araç, görsel programlamada arayüz oluşturabilmek için ihtiyacımız olan bileşenlerdir.

C# Form da araçlar araç kutusu (**toolbox**) üzerinden eklenir.

Bir web sitesinde kullanıcı adını girdiğimiz kutucuk, kutucuğun başındaki açıklama, sisteme giriş yapmak için tıkladığımız düğme gibi bileşenlerin her biri birer araçtır.

C# Form da kullanacağımız temel araçlar şunlardır;

Button

Label

Textbox

Combobox

Listbox

Maskedtextbox

PictureBox

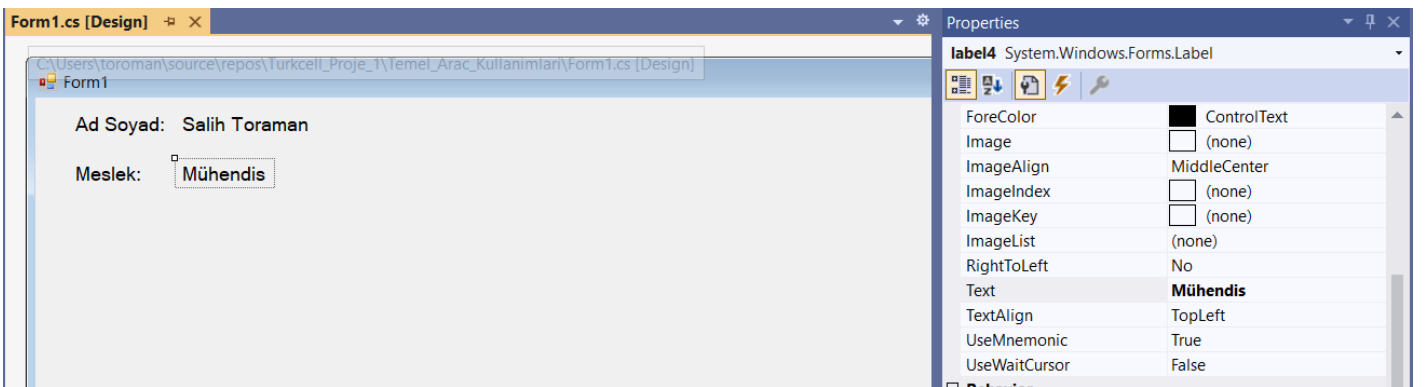
Checkbox

Groupbox

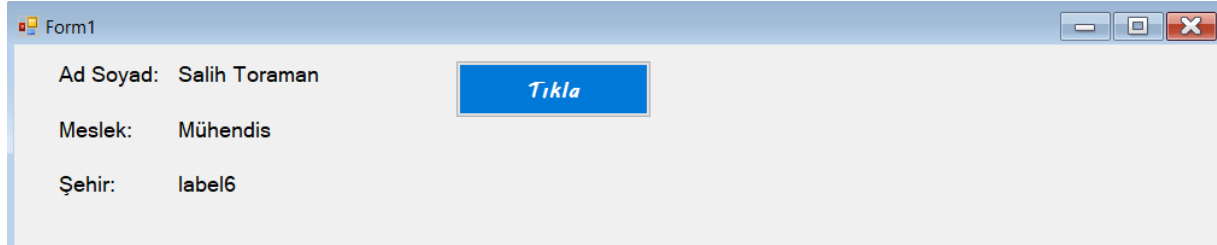
Panel

Label Aracı, formda özellikle açıklamalar için kullanılan ve etiket görevi gören araçtır.

Amaç ilgili bileşen hakkında bilgi vermektir.



Button aracı, olayları gerçekleştirmek için kullanılan tetikleme aracıdır. Bir lambayı açıp kapatmak için ihtiyacımız olan düğme aslında bir butondur. Örnek olarak 6 tane Label ve 1 tane Button oluşturduk.



“Tıkla” butonuna 2 kere tıkladıktan sonra backend ekranı gelir. (Aşağıdaki)

```
namespace Temel_Arac_Kullanimlari
{
    3 references
    public partial class Form1 : Form
    {
        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        1 reference
        private void button1_Click(object sender, EventArgs e)
        {
            label6.Text = "Bursa";
        }
    }
}
```

“Tıkla” butonuna tıkladığımızda **Bursa** yazması için backend tarafına kod yazdık.

Daha sonra kodumuzu çalıştırırız. (F5'e bas)

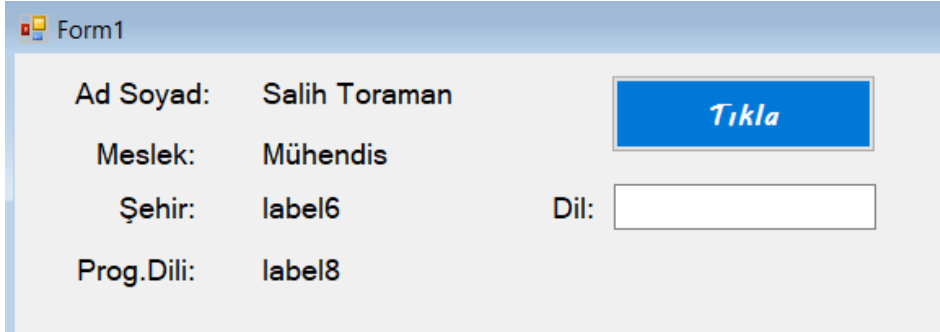


“Tıkla” butonuna tıkladıktan sonra şehir kısmında **“Bursa”** yazdı.

Textbox aracı, veri girişi yapmak için kullanılan araçtır.

Yapı olarak Label aracının özelliklerine çok benzer.

Örnek olarak 6 tane Label, 1 tane Button, 1 tane TextBox oluşturduk.



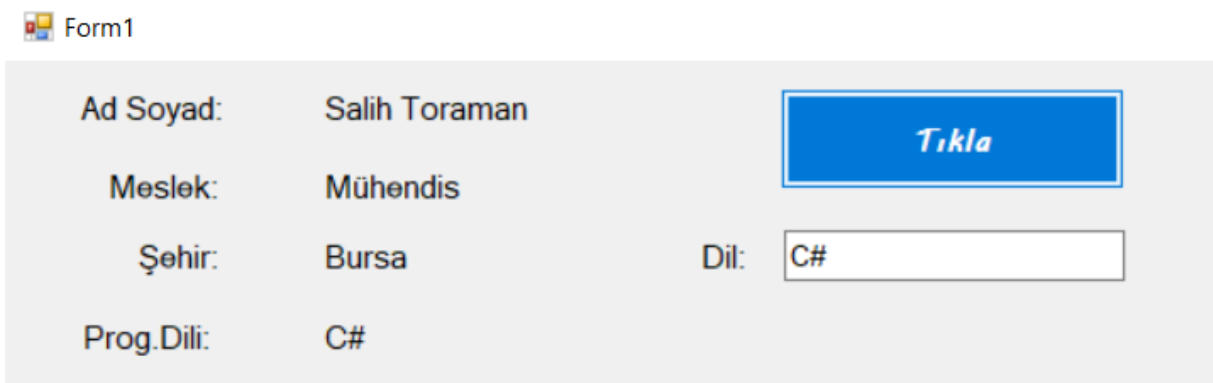
“Tıkla” butonuna 2 kere tıkladıktan sonra backend ekranı gelir. (Aşağıdaki)

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

    1 reference
    private void button1_Click(object sender, EventArgs e)
    {
        label6.Text = "Bursa";
        label8.Text = textBox1.Text;
    }
}
```

“Tıkla” butonuna tıkladığımızda **Bursa** yazması için ve Dil kısmına bir şey yazdığımız da label8’de görünmesi için backend tarafına kod yazdık.

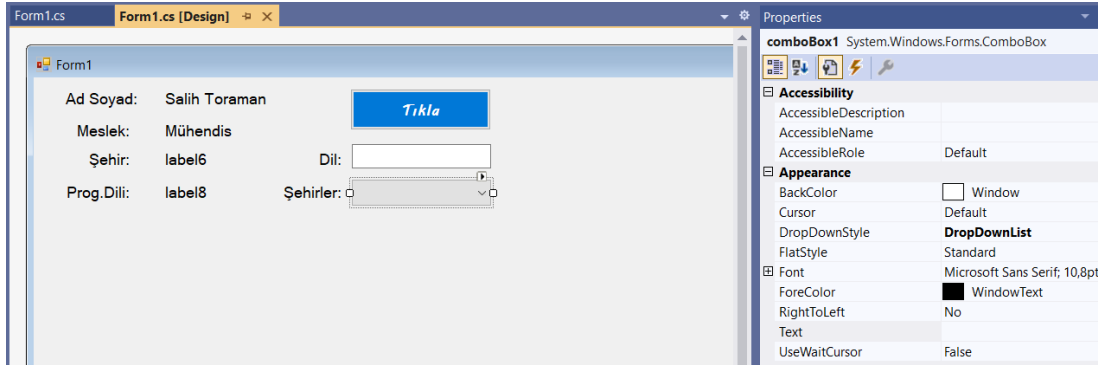
Daha sonra kodumuzu çalıştırırız. (F5’e bas)



Combobox aracı, çoklu seçim kutusu olarak kullanılan araçtır. Bir listede istenilen değerlerden herhangi birini seçtirmek için kullanılır.

Örnek olarak 8 tane Label, 1 tane Button, 1 tane TextBox ve 1 tane Combobox oluşturduk.

DropDownList seçeneği Şehirler kısmının içine yazı yazmamızı engeller sadece içerisindeki şehirleri seçmemizi sağlar.



“Tıkla” butonuna 2 kere tıkladıktan sonra backend ekranı gelir. (Aşağıdaki)

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

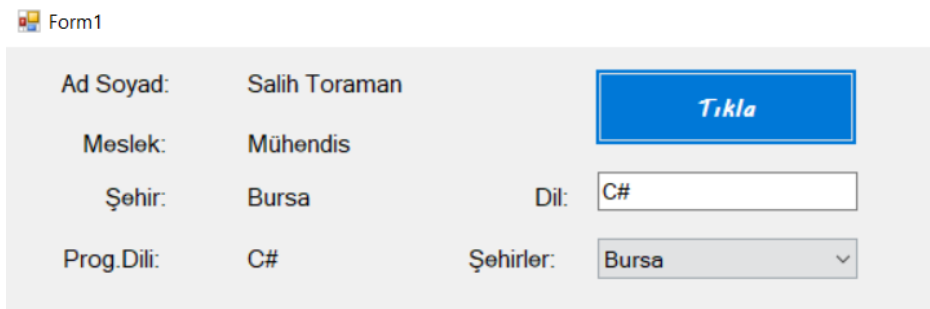
    1 reference
    private void button1_Click(object sender, EventArgs e)
    {
        //label6.Text = "Bursa";
        label6.Text = comboBox1.Text;
        label8.Text = textBox1.Text;
    }
}
```

“Tıkla” butonuna tıkladığımızda **Bursa** yazması için ve Dil kısmına bir şey yazdığımız da label8’de görünmesi için backend tarafına kod yazdık.

Daha sonra kodumuzu çalıştırırız. (F5’e bas)

Son olarak şehri seçeriz ve Dil kısmına istediğimiz dili yazarız ardından

“Tıkla” butonuna basarız.



Listbox aracı, verilerin listelenmesi için kullanılan araçtır.

Yapı olarak Combobox aracına benzer.

Listelenecek veriler >>**items.add**>> komutu kullanılarak Listbox aracına yansıtılır.

Örnek olarak 8 tane Label, 2 tane Button, 1 tane TextBox ve 1 tane Combobox ve 1 tane ListBox oluşturduk.

Program dillerini yukarıdaki ufak siyah seçeneğe tıkladıktan sonra **Edit Items** kısmına tıklayarak karşımıza çıkan ekrana ekliyoruz.

```
private void button1_Click(object sender, EventArgs e)
{
    //label6.Text = "Bursa";
    label6.Text = comboBox1.Text;
    label8.Text = textBox1.Text;
}
```

1 reference

```
private void button2_Click(object sender, EventArgs e)
{
    listBox1.Items.Add("Javascript");
    comboBox1.Items.Add("Kütahya");
    listBox1.Items.Add(textBox1.Text);
}
```

F5'e basarak kodumuzu çalıştırıyoruz

Dil kısmına textBox1 kısmına Oracle yazıyoruz ve Listele butonuna tıklıyoruz. Sonuç olarak şehirler kısmına Kütahya, dil kısmına Oracle eklendi.

MaskedTextBox aracı, maskeli metin kutusu anlamına gelir.

Textbox aracının bazı nitelikler alan formatıdır.

Telefon numarası ve TC kimlik numarası gibi formatlarda kullanılır.

(.)- MaskedTextBox


oluşturup türünü aşağıdaki gibi değiştirebilirsin.

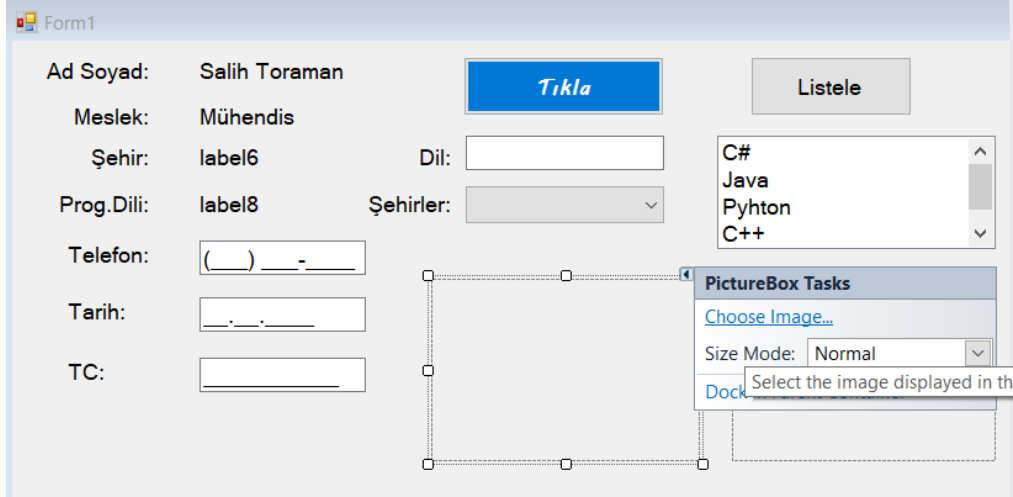
The screenshot shows a Windows Forms application window titled "Form1". The form contains several controls: a label "Ad Soyad:" with text "Salih Toraman", a label "Meslek:" with text "Mühendis", a label "Şehir:" with text "label6", a label "Dil:" with a text box, a label "Prog.Dili:" with text "label8", a label "Şehirler:" with a dropdown menu, a label "Telefon:" with a MaskedTextBox control showing "() -", a label "Tarih:" with a date picker, and a label "TC:" with a MaskedTextBox control. A blue button labeled "Tıkla" is next to the "Ad Soyad:" label. A dialog box titled "Input Mask" is open, showing a list of predefined mask descriptions and their data formats and validating types. The "Short date" mask is selected. The dialog box also has a "Mask:" field with "00/00/0000" and a "Preview:" field showing "00/00/0000".

Mask Description	Data Format	Validating Type
Numeric (5-digits)	12345	Int32
Phone number	(574) 555-0123	(none)
Phone number no area code	555-0123	(none)
Short date	12/11/2003	DateTime
Short date and time (US)	12/11/2003 11:20	DateTime
Social security number	000-00-1234	(none)
Time (European/Military)	23:20	DateTime
Time (US)	11:20	DateTime
Zip Code	98052-6399	(none)

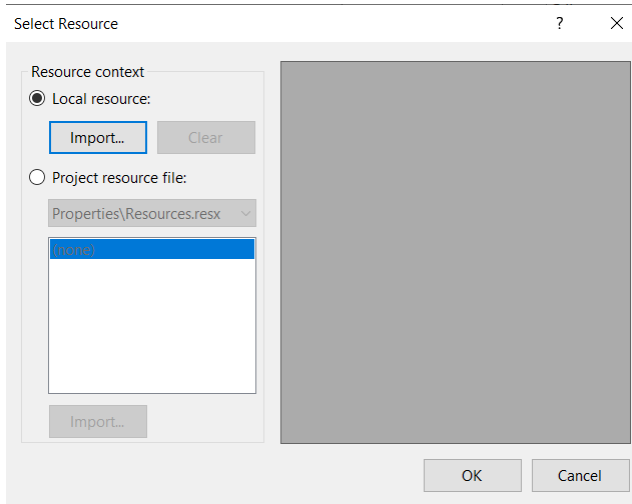
The screenshot shows the same Windows Forms application window "Form1". The controls are the same as in the previous screenshot. The "TC:" label now has a MaskedTextBox control showing "12345678900". The "Tıkla" button is still present. The "Listele" button is also visible. A list box on the right side of the form contains the following items: C#, Java, Python, C++, R.

PictureBox aracı, formda resim kutusu olarak kullanılan araçtır. Amaç forma resim dosyaları ekleyip görüntüleyebilmektir.

 PictureBox oluştur

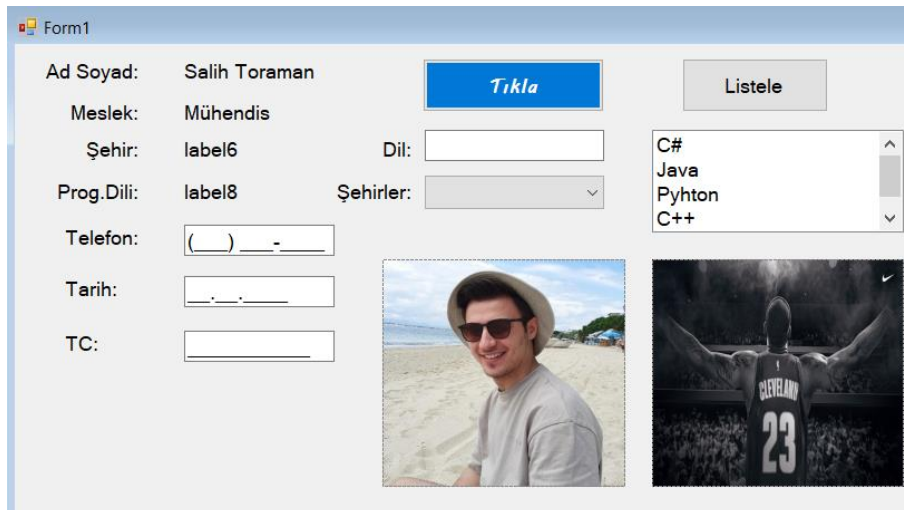


“Choose Image” kısmına tıklayıp resim seçip yükleyebilirsiniz.



“Local resource” kısmından istediğin resmi bilgisayarından seçebilirsin.

Sonuç:



GroupBox aracı, araçları gruplamak için kullanılan form aracıdır. Amaç özellikle birbiriyle ilişkili olan bileşenleri bir arada tutabilmektir.



GroupBox

oluşturdun



Resimleri **GroupBox**'ın içine aldın.

MessageBox, istenilen durumlarda kullanıcıya bir diyalog penceresi aracılığıyla mesaj vermek için kullanılan bileşendir.

4 ana parametre alır.

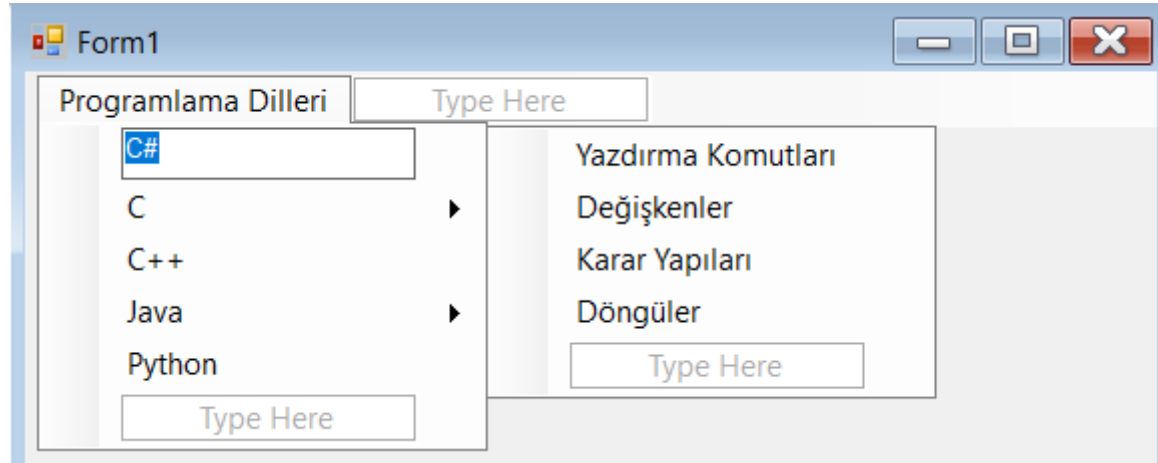
- 1- Mesaj
- 2- Başlık
- 3- Buton
- 4- İkon

MenuStrip Aracı, masaüstü programlarda yer alan üst menülerin oluşturulması için kullanılacak araçtır.

Menüler alt tarafa ve sağ tarafa doğru uzayabilir.



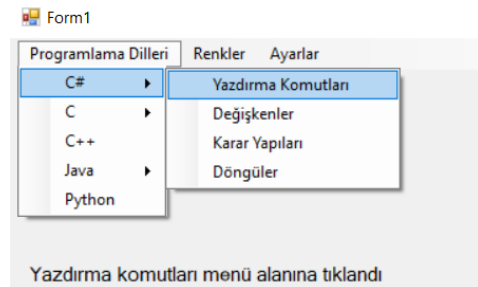
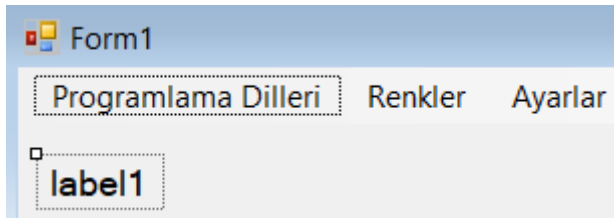
Toolbox'dan bir MenuStrip aldın ve aşağıdaki gibi doldurdun.



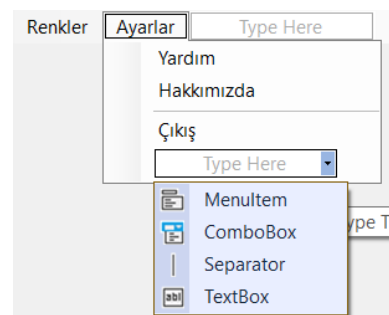
Yazdırma Komutlarına çift tıklayarak backend tarafına gerekli kodları yazıyoruz.

```
1 reference
private void yazdırmaKomutlarıToolStripMenuItem_Click(object sender, EventArgs e)
{
    label1.Text = "Yazdırma komutları menü alanına tıklandı";
}
```

Yazdırma Komutlarına tıkladığımızda yazdırmak istediğimiz metin label1'de görünür.



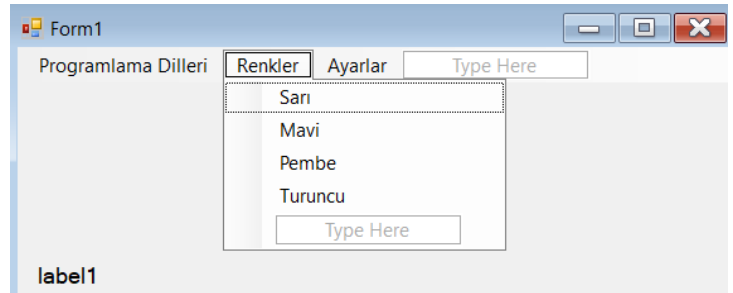
Siyah imlece tıklayıp **Separator**'ü seçerek Çıkış'ın altında bir çizgi oluştu ve onu yukarıya taşıdık.



Amacımız Renkler menüsünden istediğimiz rengi seçtiğimizde Form'un arka planının o renkte olması.

Sarı'nın üzerine çift tıklıyoruz ve backend ekranına gerekli kodu yazıyoruz.

Not: `this` komutu Form1 ile ilgili yapılacak özelliklerde kullanılan komuttur.



Gerekli kod backend tarafına her renk için yazıldı.

```
private void sarıToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.BackColor = Color.Yellow;
}
```

1 reference

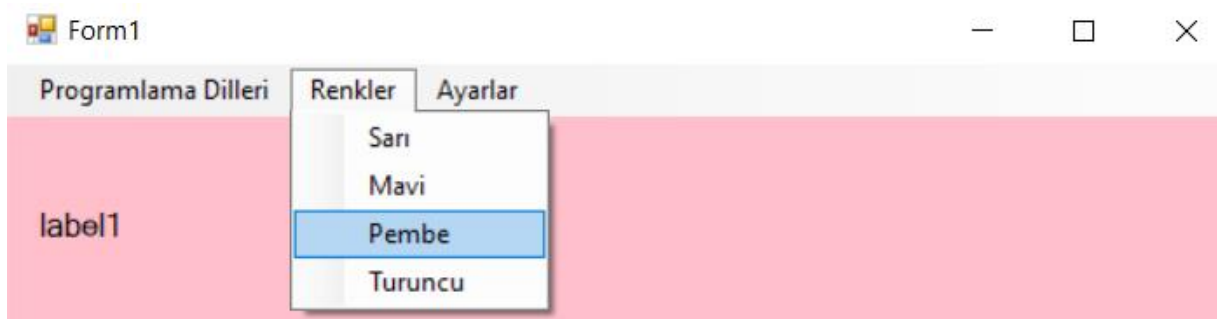
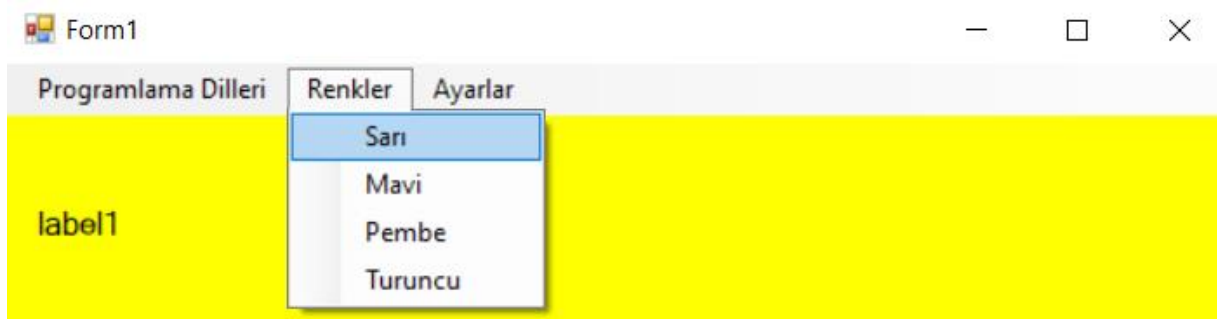
```
private void maviToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.BackColor = Color.Blue;
}
```

1 reference

```
private void pembeToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.BackColor = Color.Pink;
}
```


1 reference

```
private void turuncuToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.BackColor = Color.Orange;
}
```

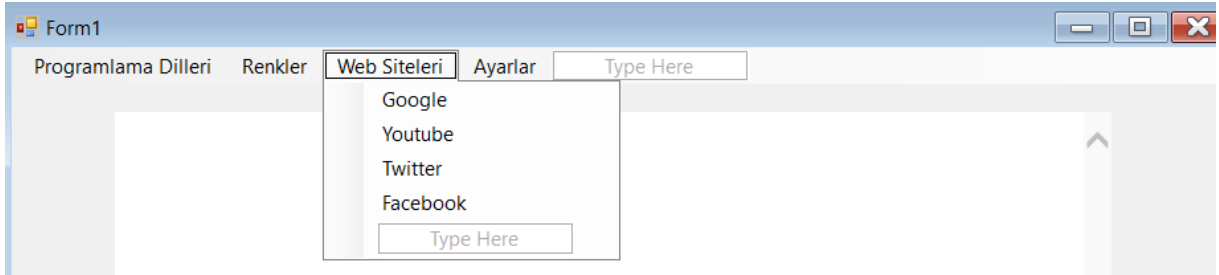


Web Browser aracı, form üzerinden web sayfalarına bir tarayıcı ile ulaşmak için kullanılır.

Varsayılan olarak internet Explorer alt yapısını kullanmaktadır.

 WebBrowser oluşturduk

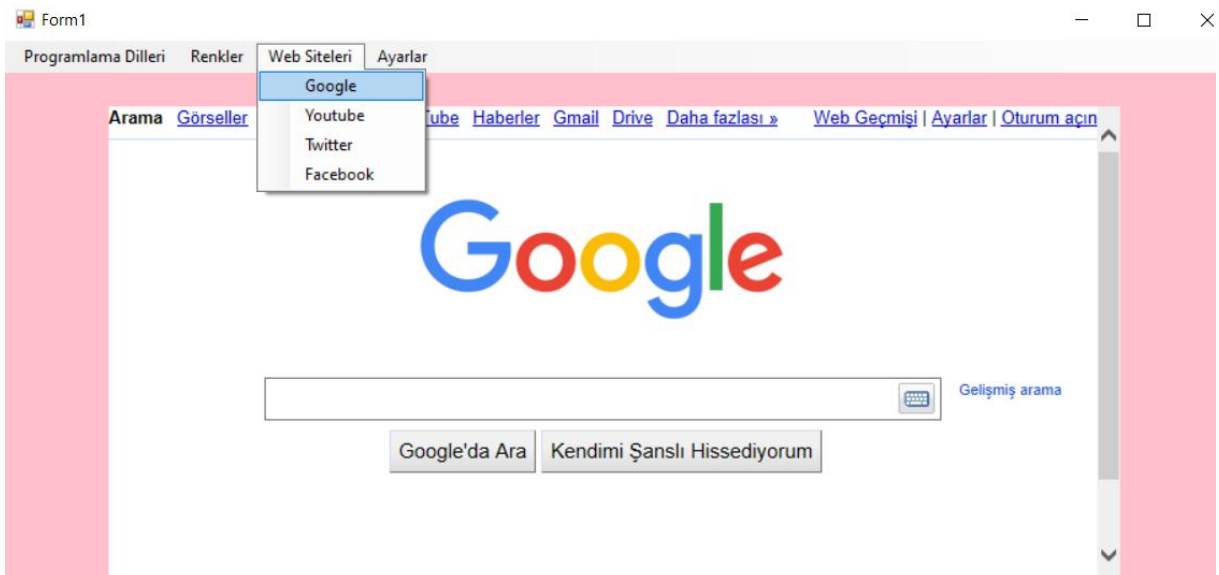
Amacımız Web Siteleri bölümünde herhangi birine tıkladığımıza o web sitesinin açılması.



Google'ın üzerine çift tıklıyoruz ve backend ekranına uygun kodu yazıyoruz

```
private void googleToolStripMenuItem_Click(object sender, EventArgs e)
{
    webBrowser1.Navigate("http://www.google.com");
}
```

Sonuç:

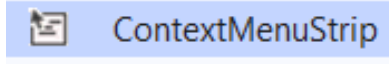


Context MenuStrip aracı, form üzerinde sağ tuş menüsü oluşturmak için kullanılır.

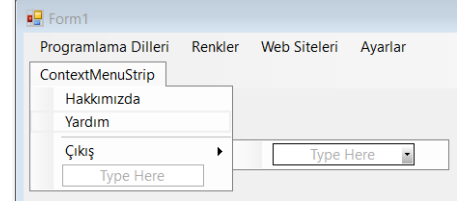
Yapı olarak MenuStrip aracına benzer.

Bir formda birden fazla Context MenuStrip aracı tanımlanabilir.

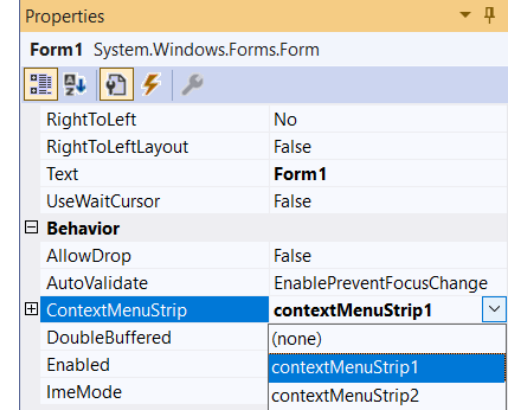
Kullanılacak olan sağ tuş menüsü formun özellikleri penceresinden ayarlanır.



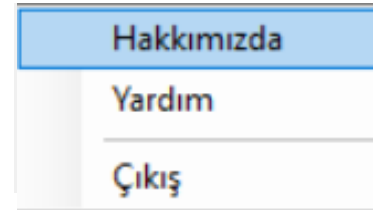
Oluşturuldu →



Forma sağ tıkladığımızda seçeneklerin ekrana gelmesi için ContextMenuStrip özelliğini tıklayıp (none) olan kısmı **contextMenuStrip1** ile değiştiriyoruz. →



Sonuç olarak Form üzerinde sağ tıkladığımızda yandaki seçenekler karşımıza gelir. →

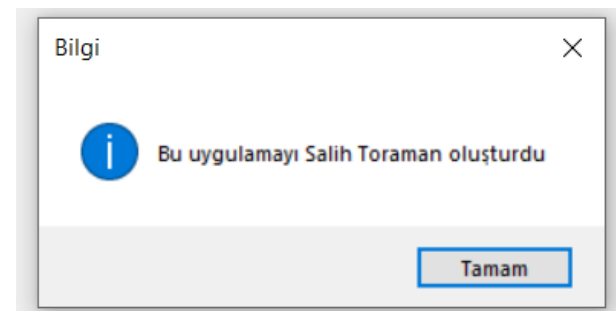


Hakkımızda seçeneğine tıkladığımızda bir ekrana bir yazı çıkmasını istediğimiz için backend tarafına gerekli kodlamaları yazdım. ↓

```
private void hakkımızdaToolStripMenuItem1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Bu uygulamayı Salih Toraman oluşturdu", "Bilgi", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
```

MessageBoxButtons.OK = Tamam butonu için

MessageBoxIcon.Information = Bilgi yazısı için



Timer aracı, zamanlayıcı olarak adlandırılan Timer aracında amaç istenen işlemlerin belirli periyotlarda otomatik olarak gerçekleştirilmesidir.

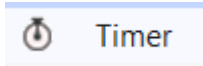
Timer aracı aşağıdaki uygulamalarda kullanılabilir;

Kronometre uygulaması

Trafik lambası simülasyonu

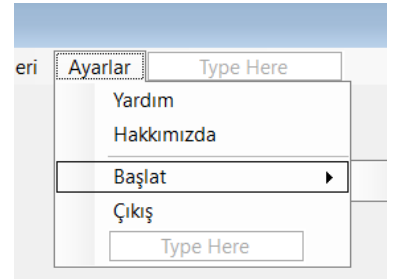
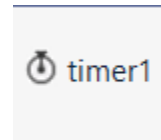
Yedek alma işlemleri

Amacımız Başlat'a tıkladığımızda **Timer** çalışmaya başlayacak ve label2'de sayılar 0,1,2,3,4,5 olarak artmaya başlayacak.



Form ekranı üzerine eklendi.

Timer'a ve Başlat'a çift tıklayarak backend ekranı karşımıza gelir. →

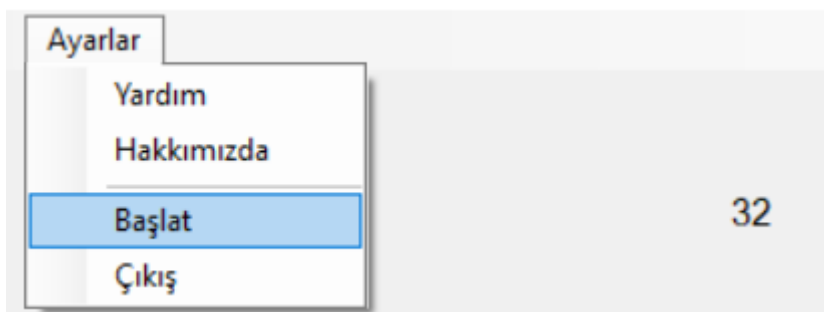


Backend tarafına uygun kod blokları yazıldı. ↓

```
int sayac=0;
1 reference
private void başlatToolStripMenuItem_Click(object sender, EventArgs e)
{
    timer1.Start();
}

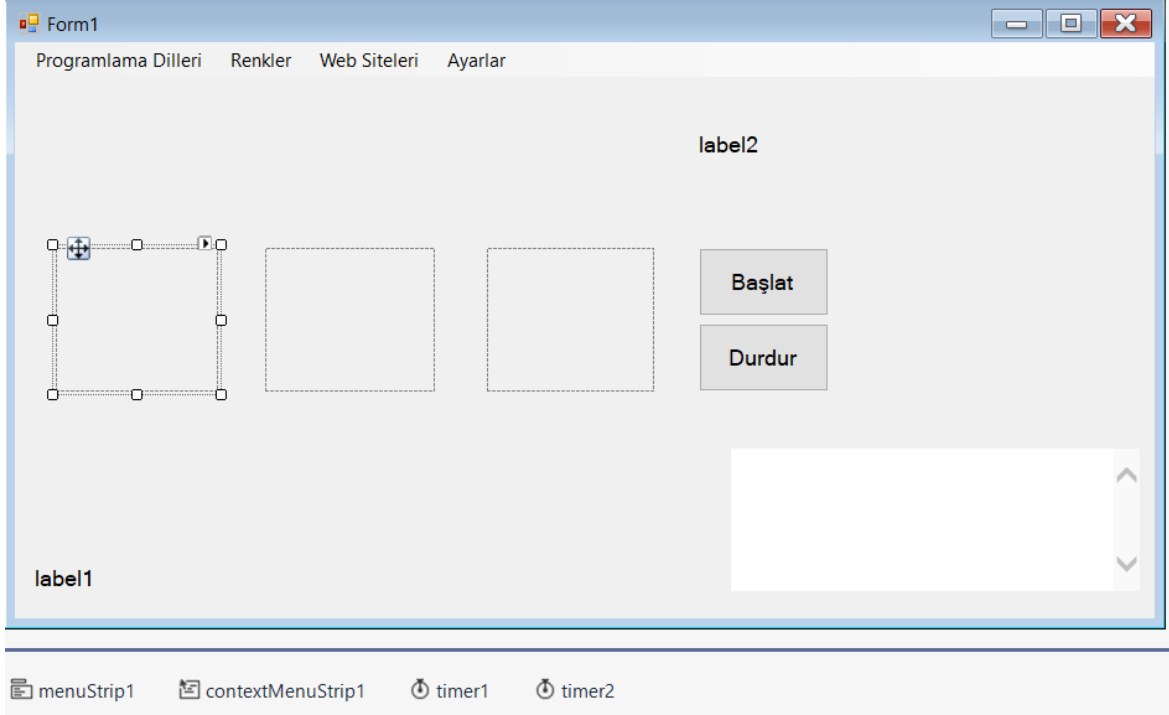
1 reference
private void timer1_Tick(object sender, EventArgs e)
{
    sayac++;
    label2.Text=sayac.ToString();
}
```

Başlata tıkladığımızda sayaç başlar. Her 10 değeri 1 saniyeye eşittir.



Trafik Işığı Simülasyonu

3 tane Panel, 2 tane Button ve 1 tane Timer2 ekledik. ↓



Başlat, Durdur ve Timer2'ye çift tıklayarak backend tarafına geçildi ve gerekli kodlar yazıldı. ↓

Not: [this](#) komutu Form1 ile ilgili yapılacak özelliklerde kullanılan komuttur.

```
private void BtnBaslat_Click(object sender, EventArgs e)
{
    timer2.Start();
}
```

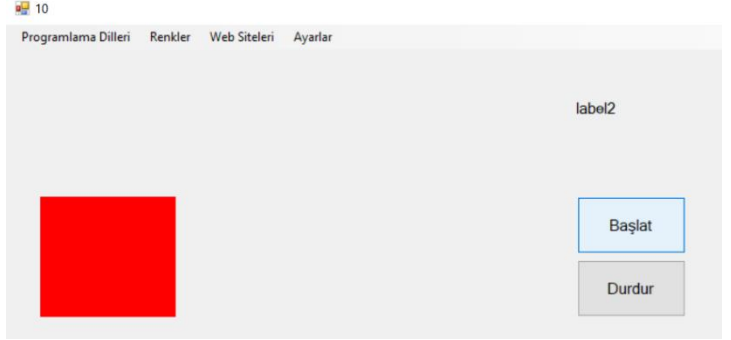
1 reference

```
private void BtnDurdur_Click(object sender, EventArgs e)
{
    timer2.Stop();
}
```

```
int sure = 0;
```

1 reference

```
private void timer2_Tick(object sender, EventArgs e)
{
    this.Text = sure.ToString();
    sure++;
    if (sure > 0 && sure <= 30)
    {
        panel1.BackColor = Color.Red;
        panel3.BackColor = Color.Transparent;
    }
    if (sure > 30 && sure <= 40)
    {
        panel2.BackColor = Color.Yellow;
    }
    if (sure > 40 && sure <= 70)
    {
        panel3.BackColor = Color.Green;
        panel1.BackColor = Color.Transparent;
        panel2.BackColor = Color.Transparent;
    }
    if (sure == 71)
    {
        sure = 0;
    }
}
```



70 den sonra sayaç sıfırlanır ve başa döner.

Not:

panel1.BackColor = Color.Red;

panel3.BackColor = Color.**Transparent**;

(kırmızı ışık yandığında yeşil ışık **söner**)

panel3.BackColor = Color.Green;

panel1.BackColor = Color.**Transparent**;

panel2.BackColor = Color.**Transparent**;

(yeşil ışık yandığında kırmızı ve sarı ışık **söner**)

Chart aracı, verilerin grafikler üzerinde gösteriminde kullanılan araçtır.

Özellikle veritabanı işlemlerinde çokça kullanılır.

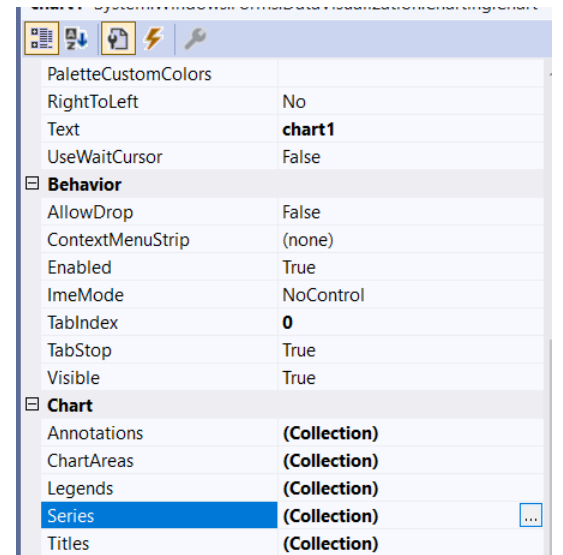
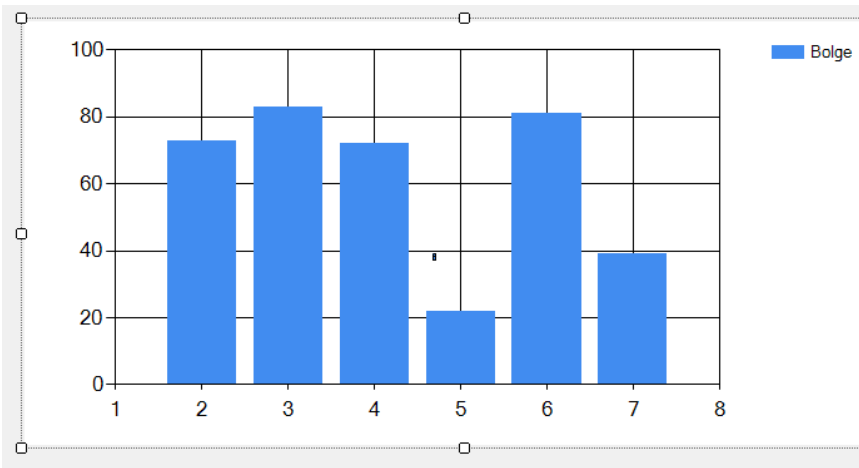
Par, pie, radar gibi türleri vardır.

Gösterimi yapılacak veriler <<series>> başlığı altında tutulur.

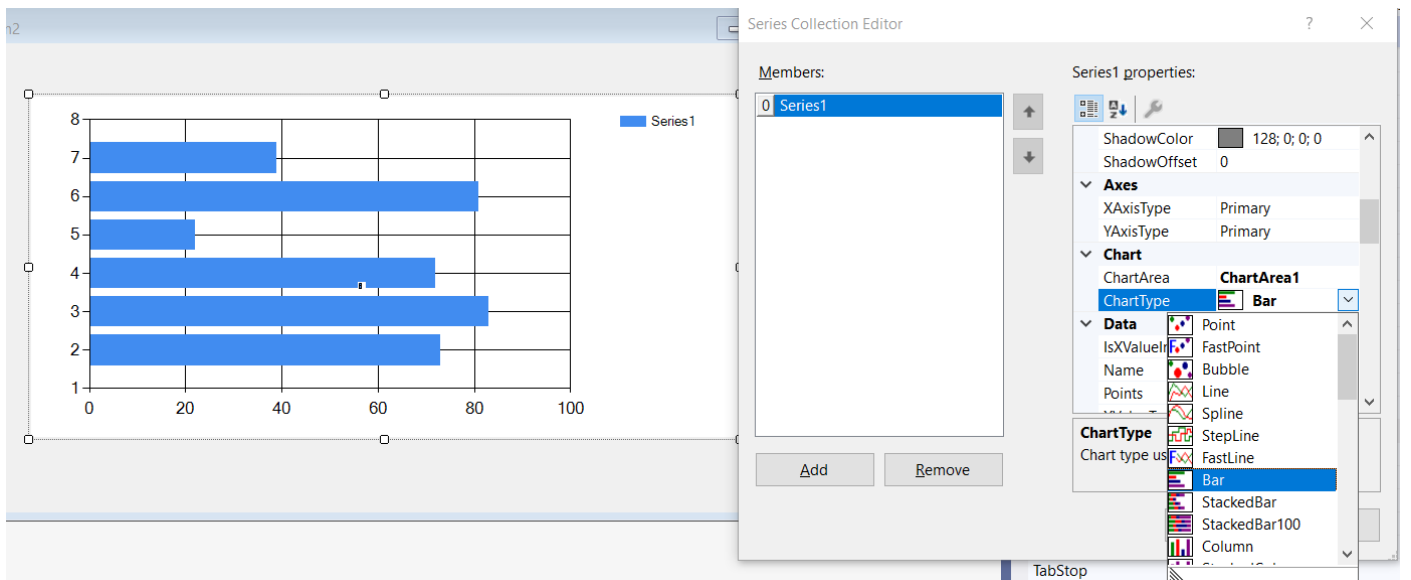
Verilere genellikle x ve y koordinatında değer ataması yapılır.



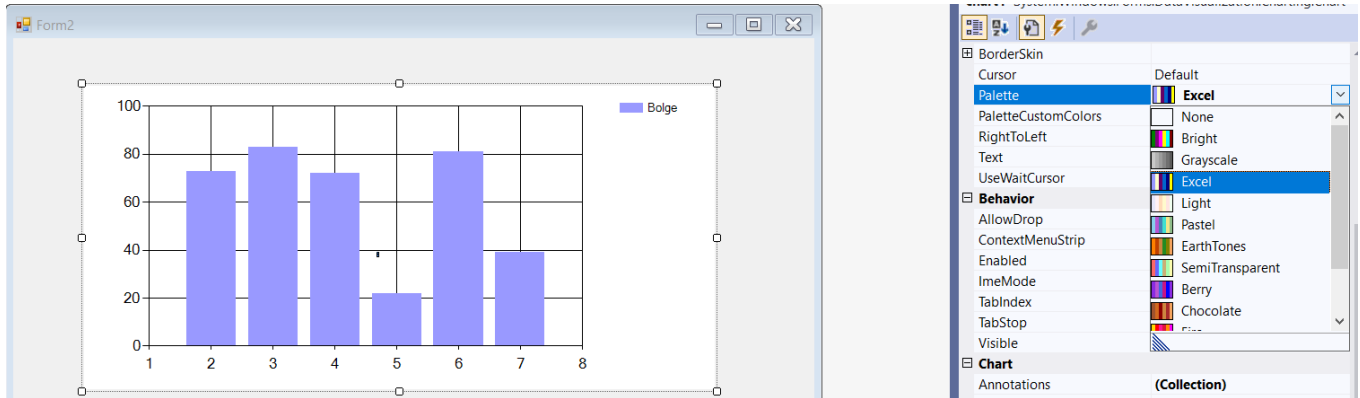
Grafiğin üzerine çift tıklayıp Properties kısmına gelerek Series kısmını aç ↓



ChartType kısmından grafiğin şeklini farklı türlerde seçebilirsin. ↓



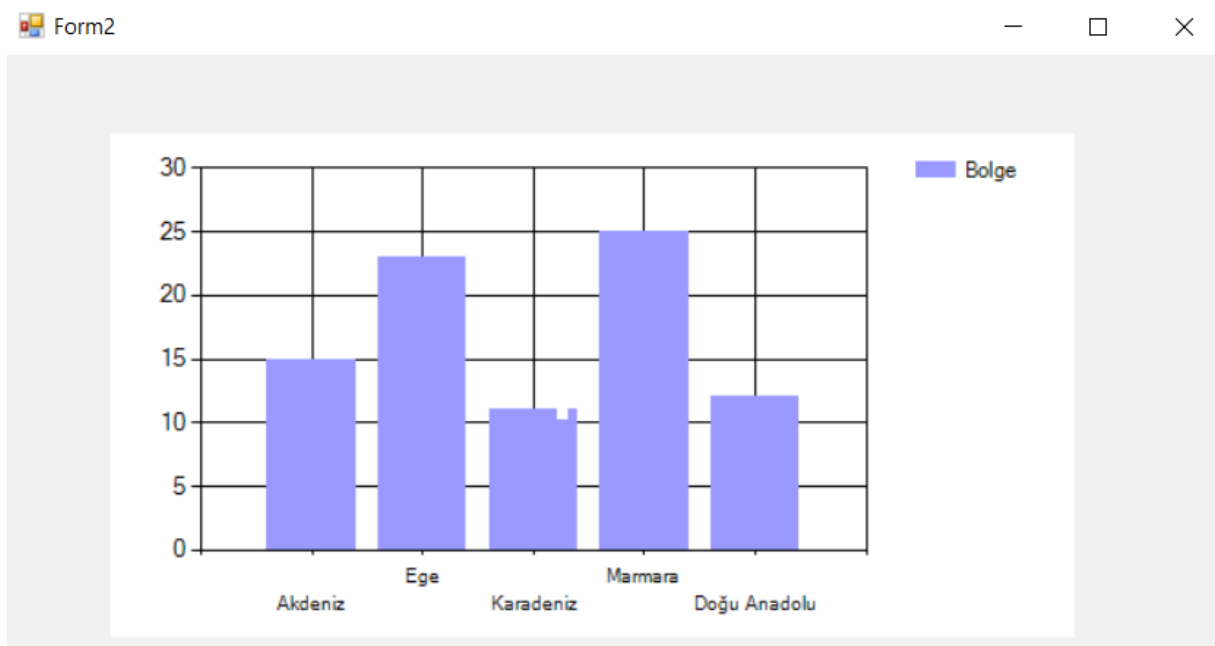
Palette kısmından grafik rengini seçebilirsin. ↓



Grafiği çift tıklayarak backend tarafını açarak kodlamaları yapıldı. ↓

```
private void chart1_Click(object sender, EventArgs e)
{
    chart1.Series["Bolge"].Points.AddXY("Akdeniz", 15);
    chart1.Series["Bolge"].Points.AddXY("Ege", 23);
    chart1.Series["Bolge"].Points.AddXY("Karadeniz", 11);
    chart1.Series["Bolge"].Points.AddXY("Marmara", 25);
    chart1.Series["Bolge"].Points.AddXY("Doğu Anadolu", 12);
}
```



Sonuç ↓



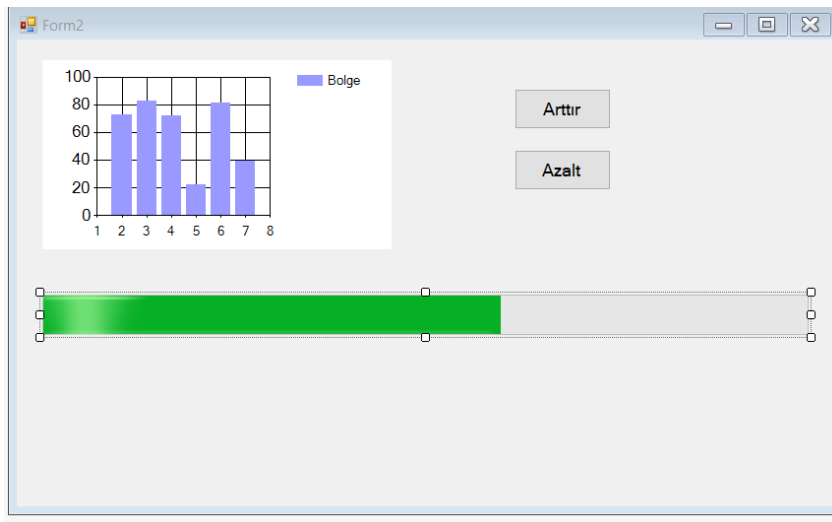
Progressbar aracı, sayısal değerin bir ilerleme çubuğu üzerinde gösterildiği araçtır.

Başlangıçta 0 – 100 arasında değer alır.

Minimum ve maximum değerleri değiştirilebilir.

 **ProgressBar**  **Button** 2 Button ve 1 ProgressBar ekledin

ProgressBar'ın değerini 0 değil de 60 dan başlattın ve daha sonra Arttır butonuna tıklayıp backend tarafına geçtin ve kodlamaları yaptın.



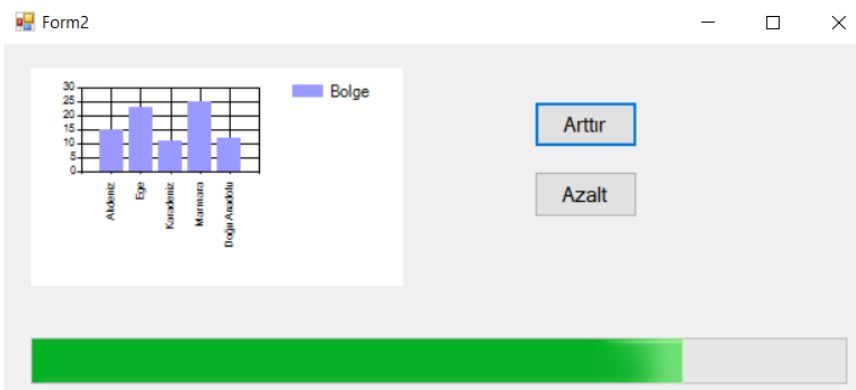
Accessibility	
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
Appearance	
BackColor	Control
Cursor	Default
ForeColor	Highlight
RightToLeft	No
RightToLeftLayout	False
UseWaitCursor	False
Behavior	
ContextMenuStrip	(none)
Enabled	True
MarqueeAnimationSpeed	100
Maximum	100
Minimum	0
Step	10
Style	Blocks
TabIndex	3
Value	60

```
private void BtnArttır_Click(object sender, EventArgs e)
{
    progressBar1.Value += 10;
}
```

1 reference

```
private void BtnAzalt_Click(object sender, EventArgs e)
{
    progressBar1.Value -= 10;
}
```

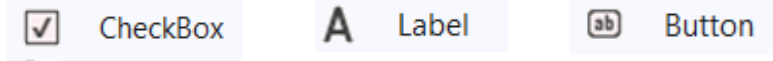
Sonuç olarak 10 arttırabilir veya azaltabilirsin.



Checkbox aracı, kontrol kutusu anlamına gelir.

Çoklu seçim durumlarında kullanılmaktadır.

5 tane CheckBox, 1 tane Label, 1 tane Button ekledin.

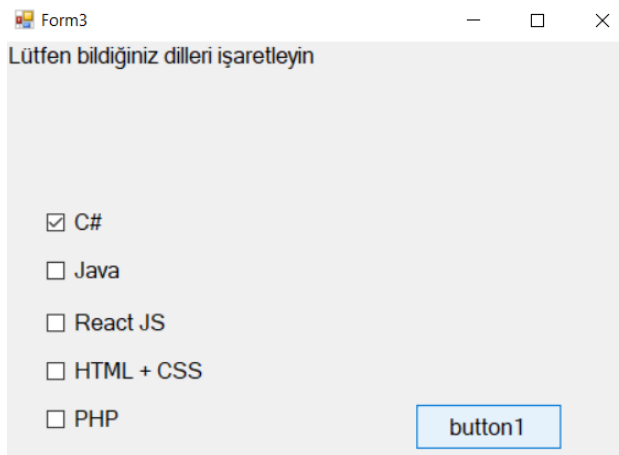


Button1'e çift tıklayarak backend tarafını açtın ve kodlamaları yaptın.

Button1'e tıkladığında C# (Checkbox1) işaretlenmiş olarak görülecek. ↓

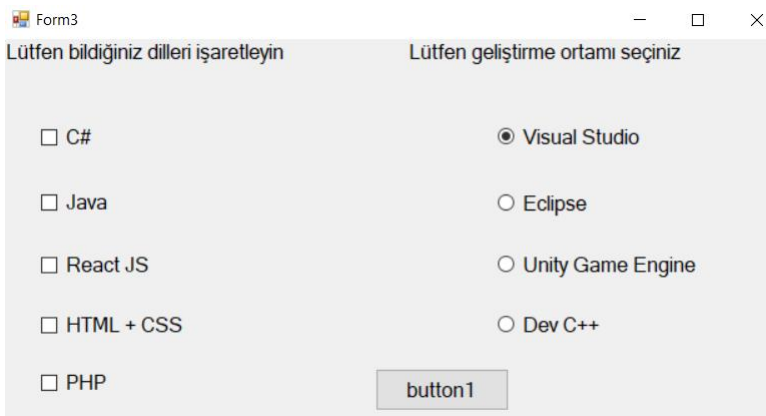
```
private void button1_Click(object sender, EventArgs e)
{
    checkBox1.Checked = true;
}
```

Sonuç ↓



Radiobutton aracı, çoklu seçim işlemlerinde sadece 1 tane değer seçimine izin veren araçtır.

Form üzerinde bulunan Checkbox araçlarının tamamı seçilebilirken, Radiobutton ise aynı anda sadece 1 tane seçilebilmektedir.



SQL ile C# arasındaki bağlantılarla alakalı notlar

En başta Form'da backend tarafına **using System.Data.SqlClient;** komutunu eklemelisin

DataGridView, veri tabanından veri çekmek yani verileri listelemek için kullanılır.

Database de tablolar arasında ilişkilerin kurulması için Database Diagrams kısmına sağ tuşla tıklayıp New Database Diagrams seçeneğine tıklaman gerekiyor. Böylece yeni bir veritabanı diagramı oluşturmuş olacaksın.

İlişki içerisine alınacak sütunların veri tipleri mutlaka **aynı** olmalı.

SqlConnection: Bağlantı sınıfı

SqlConnection baglanti = new SqlConnection(@"buraya baglanti adresini kopyala");
SqlConnection sınıfından baglanti adında bir nesne türettin.

SqlCommand: Komut sınıfı

SqlCommand komut = new SqlCommand ("Select * From TBLKATEGORI", baglanti);
SqlCommand sınıfından bir komut adında bir nesne türettin.

SqlDataAdapter: Köprü sınıfı

SqlDataAdapter da = new SqlDataAdapter (komut);

DataTable: Veri tablosu

DataTable dt = new DataTable();

da.Fill(dt);

da'dan gelen değer ile (yani köprüden gelen) dt'yi dolduracağım (veri tablomu).

dataGridView1.DataSource = dt;

dataGridView1'in veri kaynağı kısmına (DataSource kısmına) dt'den gelen değeri gönderiyoruz.

db.SaveChanges();

Değişiklikleri kaydetmek için

Sonuç olarak Form kısmında gerekli butona tıkladığında veritabanından veriler gelmiş olacak yani kategorileri listelemiş olacaksın.

Not: **baglanti, komut, da, dt** gibi nesne isimleri tamamen sana bağlı istediğin şekilde adlandırabilirsin.

ExecuteNonQuery, sorguyu çalıştırmak ve değişiklikleri veritabanına yansıtmak için kullanılan komut.
komut.ExecuteNonQuery();

baglanti.Close();
baglantiyi kapatmak için kullanılan komut.

MessageBox.Show, ekrana mesaj vermek için kullanılan komut.
MessageBox.Show("Kategoriniz başarılı bir şekilde eklendi");

comboBox1.DisplayMember = "Ad";
kullanıcımıza gözükecek olan kısım tablodaki "Ad" olacak.

comboBox1.ValueMember = "ID";
arka planda çalışacak olan kısım "ID" olacak.

comboBox1.DataSource = dt2;
dt2'den gelen değerleri comboBox1'e atamış olduk.

Dataset, SQL üzerinden temel crud işlemlerini uzun adonet sorgularına gerek kalmadan, bunları hazır olarak veren yapıdır.

Veri kümesi olarak çevrilebilir.

CRUD = Create – Read – Update – Delete

CRUD = Ekleme – Listeleme – Güncelleme – Silme

Prosedür, uzun SQL sorgularını tek kelimelik komutlara sığdıran yapılardır.

Programlama dillerindeki metotlara benzerler.

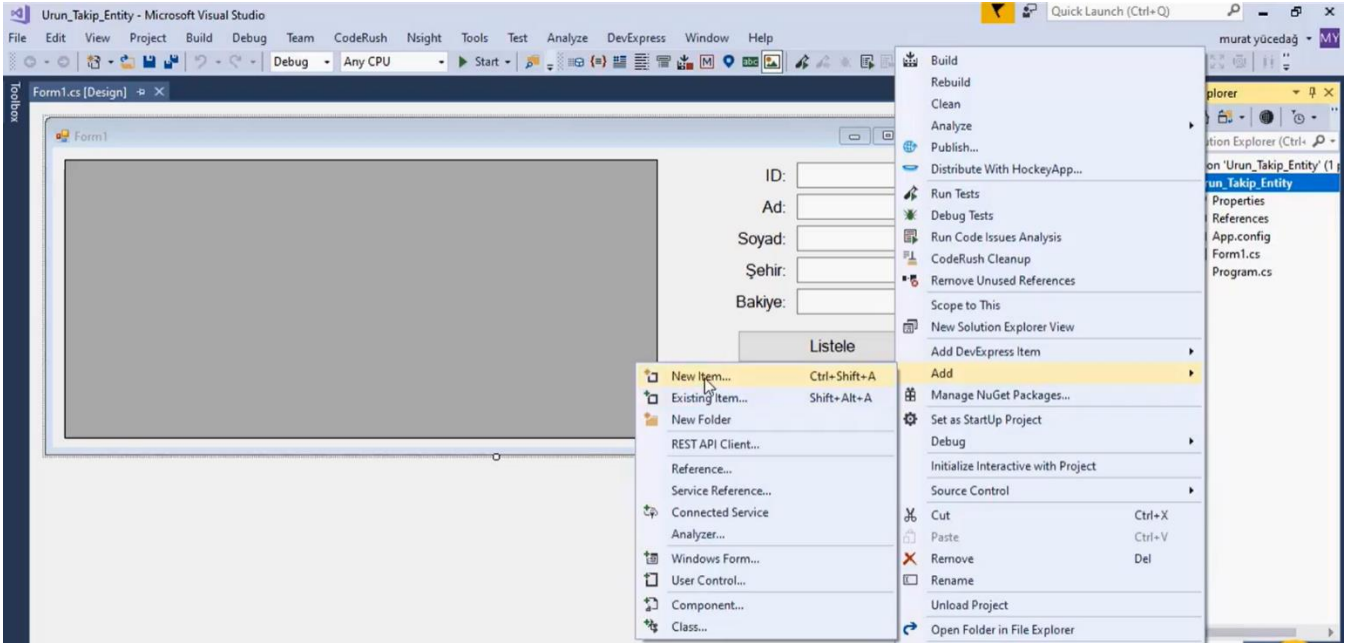
Creat komutu ile oluşturulurlar.

Execute komutu ile çağrılırlar.

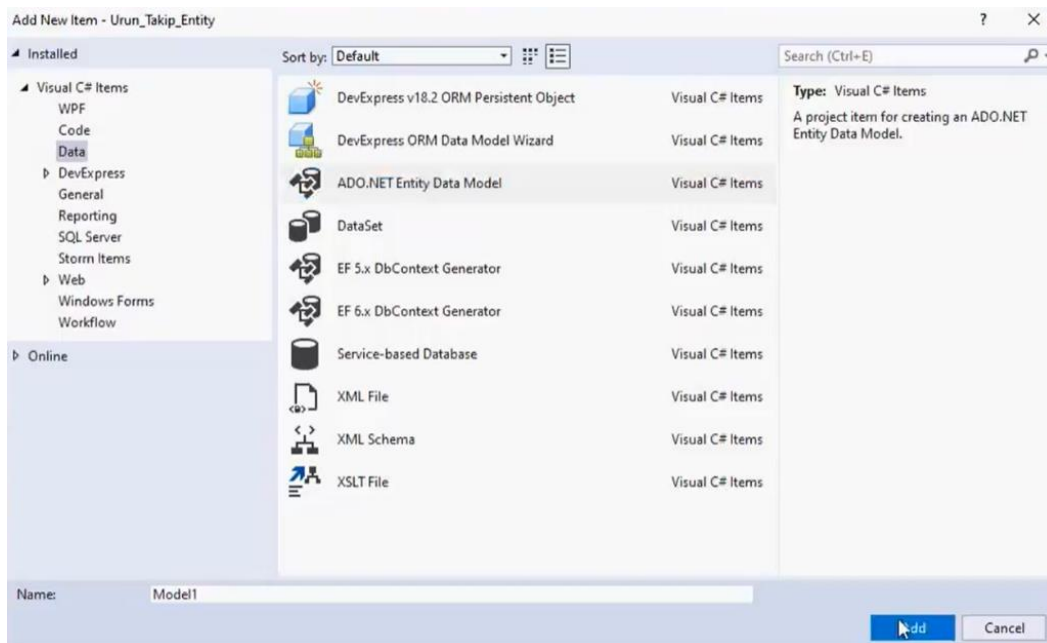
Entity Framework

Adım adım ilişkileri C# içerisine nasıl aktaracağımızı göstereceğim.

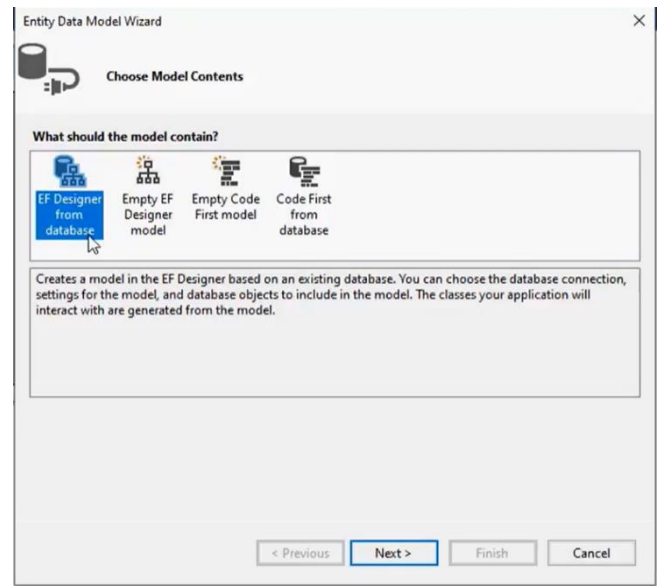
New Item diyoruz.



Data kısmından ADO.NET Entity Data Model adında yeni bir model oluşturuyoruz.



Seçeneği tıklayıp Next diyoruz ➡➡



New Connection seçeneğine tıklayıp yeni bir bağlantı oluşturup sunucu adımızı SQL den kopyalayıp Server name'in altına yapıştır.

1.adım

2.adım

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

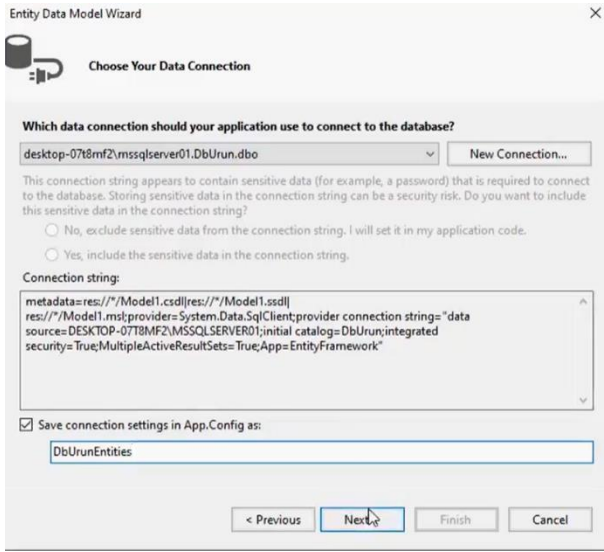
313

314

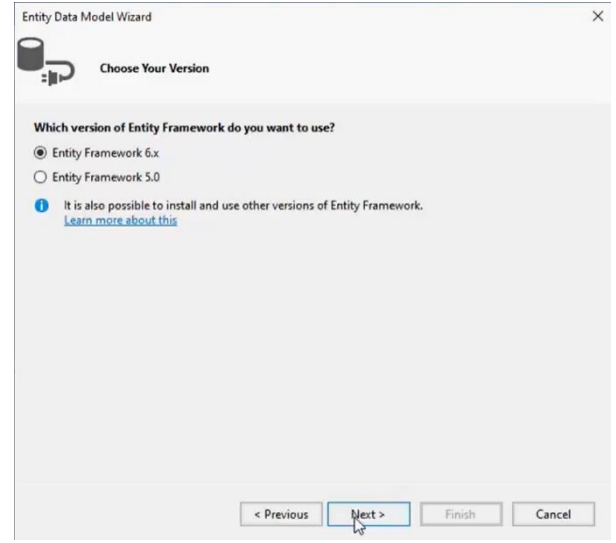
315

Next seçeneklerini tıkla.

1.adım

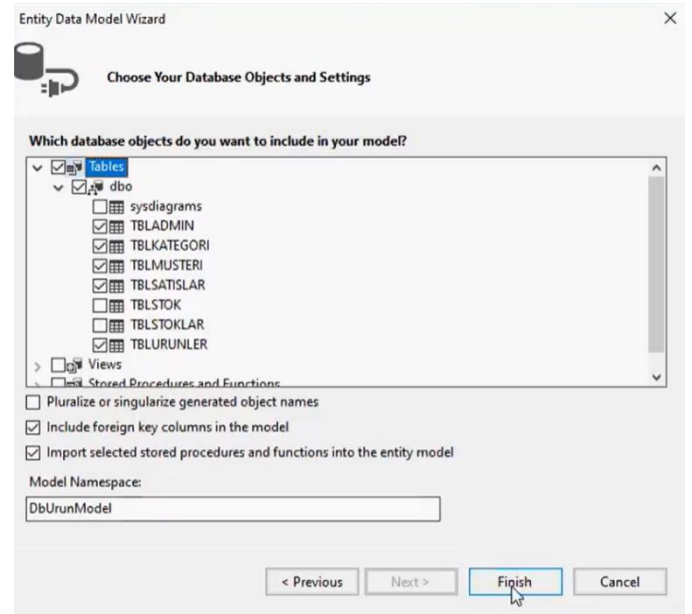


2.adım

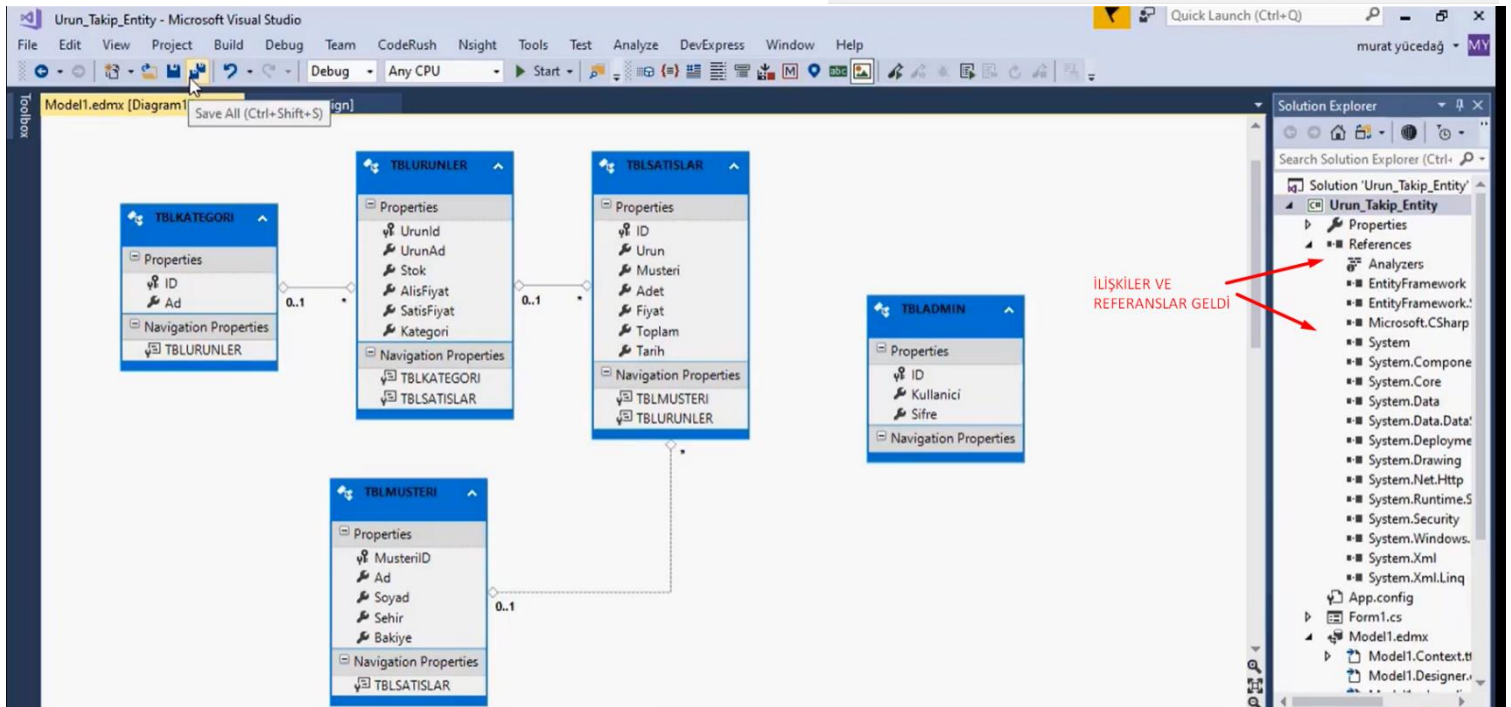


İhtiyacın olacak tabloları seç.

3.adım →→→→



SONUÇ ↓



Entity Framework, Microsoft tarafından geliştiren bir ORM'dir.

Object to Relational Mapping (ORM) = Nesne – İlişkisel Eşleme

Entity Framework, Nesne tabanlı programlama dilleri yapısına uygun olmayan katı veritabanı sorguları yerine veritabanı yapısının bir nesne gibi düşünülerek yazılım geliştirilmesine olanak sağlayan programlama teknolojisidir.

Entity Framework'un amacı daha az kodla daha fazla iş yapabilmektir.

Entity Framework'un 3 tane Temel geliştirme yaklaşımı vardır.

DbFirst

Veritabanı öncelikli yaklaşımdır ve çok sık tercih edilir.

Hazır veritabanı alınır sonra Visual Studio içerisindeki projenin içerisine entegre ediliyor ve bunun üzerinden geliştirme yapılıyor.

CodeFirst

Kod öncelikli yaklaşımdır ve hazır bir veritabanı yoktur

Veritabanı Visual Studio üzerinde sınıflar ve nesneler aracılığıyla oluşturulur aynı şekilde tablolar da sınıflar üzerinden oluşturuluyor.

Bütün işlemler Visual Studio üzerinde gerçekleştirilir daha sonrasında bu değişiklikler SQL üzerine aktarılır.

ModelFirst

Model öncelikli yaklaşımdır ve çok fazla tercih edilmez.

Yapı olarak DbFirst'e benzer.

```
namespace Urun_Takip_Entity
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        DbUrunEntities db = new DbUrunEntities();
        private void BtnListele_Click(object sender, EventArgs e)
        {
            dataGridView1.DataSource = db.TBLMUSTERI.ToList();
        }
    }
}
```

dataGridView1.DataSource = db.TBLMUSTERI.ToList();

dataGridView1'in veri kaynağı kısmına (DataSource kısmına)

db.TBLMUSTERI'den gelen değerleri gönderiyoruz ve Liste yapmasını istiyoruz.

Bunu da **db** nesnesi aracılığıyla yaptık.

Bizim t nesnesi aracılığıyla TBLMUSTERI tablomuz içerisinde yer alan SQL tarafındaki ismiyle sütunlara, C# tarafındaki ismiyle property'lere ulaşmamız ve değer ataması yapmamız gerekiyor. Ekleme işlemi yapıldı.

```
private void BtKaydet_Click(object sender, EventArgs e)
{
    TBLMUSTERI t = new TBLMUSTERI();
    t.Ad = TxtAd.Text;
    t.Bakiye = decimal.Parse(TxtBakiye.Text);
    t.Sehir = TxtSehir.Text;
    t.Soyad = TxtSoyad.Text;
    db.TBLMUSTERI.Add(t);
    db.SaveChanges();
    MessageBox.Show("Yeni Müşteri Kaydı Yapıldı!");
}
```

The screenshot shows a Windows Form titled 'Form1'. It contains a table with the following columns: MusteriID, Ad, Soyad, Sehir, Bakiye, and TBLSATISLA. The table has three rows of data: (1, Murat, Yücedağ, İstanbul, 24000,00), (2, Ayşe, Çınar, Bursa, 15000,00), and (4, Ali, Kaya, Ankara, 30000). A red arrow points to the row with MusteriID 4, and a red label 'YENİ' is next to it. To the right of the table is a form with input fields for ID, Ad, Soyad, Şehir, and Bakiye, and buttons for 'Listele' and 'Kaydet'. The Ad field contains 'Ali'.

Önce Sil butonuna tıklayarak backend tarafına geçmeliyiz.

int türünde **id** adında bir değişken oluşturduk ve TxtID.Text kısmına girmiş olduğumuz değeri **id** isimli değişkene atamış olduk.

var türünde **x** adında bir değişken daha oluşturduk ve db.TBLMUSTERI içerisinde dışarıdan göndermiş olduğumuz **id** değerini bulacak.

Son olarak db.TBLMUSTERI.Remove(x); diyerek **x** den gelen değer ile komple **x** satırını silmiş oluyoruz. Silme işlemi yapıldı.

```
private void BtnSil_Click(object sender, EventArgs e)
{
    int id = int.Parse(TxtID.Text);
    var x = db.TBLMUSTERI.Find(id);
    db.TBLMUSTERI.Remove(x);
    db.SaveChanges();
    MessageBox.Show("Müşteri sistemden silindi");
}
```

The screenshot shows the application interface after clicking the 'Sil' button. The table now has two rows: (1, Murat, Yücedağ, İstanbul, 24000,00) and (2, Ayşe, Çınar, Bursa, 15000,00). The row with MusteriID 4 has been removed. A small dialog box titled 'Müşteri sistemden silindi' is displayed in the center of the form. The ID field in the form on the right now contains '4'.

Güncelleme işlemi yapıldı.

```
private void BtnGuncelle_Click(object sender, EventArgs e)
{
    int id = int.Parse(TxtID.Text);
    var x = db.TBLMUSTERI.Find(id);
    x.Ad = TxtAd.Text;
    x.Soyad = TxtSoyad.Text;
    x.Sehir = TxtSehir.Text;
    x.Bakiye = decimal.Parse(TxtBakiye.Text);
    db.SaveChanges();
    MessageBox.Show("Müşteri bilgisi güncellendi");
}
```

The screenshot shows the application interface after clicking the 'Güncelle' button. The table now has three rows: (1, Murat, Yücedağ, İstanbul, 24000,00), (2, Ayşe, Çınar, Bursa, 15000,00), and (5, Mehmet, Yücedağ, Ankara, 12000). The row with MusteriID 4 has been replaced by a new row with MusteriID 5. The form on the right now has the ID field set to '5', Ad set to 'Mehmet', Soyad set to 'Yücedağ', Şehir set to 'Ankara', and Bakiye set to '12000'. The 'Güncelle' button is highlighted.

// Entity Framework'un DbFirst yaklaşımını kullanarak sadece istediğimiz sütunların geleceği bir liste oluşturacağız.

var türünde değerler adında bir değişken oluşturduk ve x adında bir değişken oluşturup değerlerini db.TBLMUSTERI'den almasını sağladık.

x ile atamaları yapıp değerler adındaki değişkene gerekli verileri gönderdik.

DataGridView1 in veri kaynağına (DataSource) değerler adındaki değişkene gerekli verileri göndererek Liste oluşturduk.

```
DbUrunEntities db = new DbUrunEntities();
private void BtnListele_Click(object sender, EventArgs e)
{
    //dataGridView1.DataSource = db.TBLMUSTERI.ToList();
    var degerler = from x in db.TBLMUSTERI
        select new
        {
            x.MusteriID,
            x.Ad,
            x.Soyad,
            x.Sehir,
            x.Bakiye
        };
    dataGridView1.DataSource = degerler.ToList();
}
```

dataGridView1 in özellikler kısmındaki AutoSizeColumnsMode seçeneğini Fill yaparak listenin sağındaki gri kısmı beyazlattık.

MusteriID	Ad	Soyad	Sehir	Bakiye
1	Murat	Yücedağ	İstanbul	24000,00
2	Ayşe	Çınar	Bursa	15000,00
5	Mehmet	Yücedağ	Ankara	12000,00

ID:
Ad:
Soyad:
Şehir:
Bakiye:
Listele Kaydet Sil Güncelle

MusteriID	Ad	Soyad	Sehir	Bakiye
1	Murat	Yücedağ	İstanbul	24000,00
2	Ayşe	Çınar	Bursa	15000,00
5	Mehmet	Yücedağ	Ankara	12000,00

ID:
Ad:
Soyad:
Şehir:
Bakiye:
Listele Kaydet Sil Güncelle

Properties

dataGridView1 System.Windows.Forms.DataGridView

(ApplicationSettings)

(DataBindings)

(Name) dataGridView1

AccessibleDescription

AccessibleName

AccessibleRole Default

AllowDrop False

AllowUserToAddRows True

AllowUserToDeleteRows True

AllowUserToOrderColumns False

AllowUserToResizeColumns True

AllowUserToResizeRows True

AlternatingRowsDefaultCellStyle DataGridViewCellStyle { }

Anchor Top, Left

AutoSizeColumnsMode None

AutoSizeRowsMode None

BackgroundColor ColumnHeader

BorderStyle AllCellsExceptHeader

CausesValidation AllCells

CellBorderStyle DisplayedCellsExceptHeader

ClipboardCopyMode DisplayedCells

ColumnHeadersBorderStyle Fill

Entity Framework Linq İstatistikler

Entity framework ile Count, Sum, Avg gibi komutlar kullanılarak istatistik hesaplamaları yapıldı.

```
namespace Urun_Takip_Entity
{
    public partial class FrmIstatistik : Form
    {
        public FrmIstatistik()
        {
            InitializeComponent();
        }
        DbUrunEntities db = new DbUrunEntities();
        private void FrmIstatistik_Load(object sender, EventArgs e)
        {
            DateTime bugun = DateTime.Today;
            LblMusteriSayisi.Text = db.TBLMUSTERI.Count().ToString();
            LblKategoriSayisi.Text = db.TBLKATEGORI.Count().ToString();
            LblUrunSayisi.Text = db.TBLURUNLER.Count().ToString();
            LblBeyazEsysa.Text = db.TBLURUNLER.Count(x => x.Kategori == 1).ToString();
            LblToplamStok.Text = db.TBLURUNLER.Sum(x => x.Stok).ToString();
            LblBugunSatisAdedi.Text = db.TBLSATISLAR.Count(x => x.Tarih == bugun).ToString();
            LblToplamKasa.Text = db.TBLSATISLAR.Sum(x => x.Toplam).ToString() + " ₺";
            LblBugunkuKasa.Text = db.TBLSATISLAR.Where(x => x.Tarih == bugun).Sum(y => y.Toplam).ToString() + " ₺";
            LblEnYuksekFiyatliUrun.Text = (from x in db.TBLURUNLER
                                         orderby x.SatisFiyat descending
                                         select x.UrunAd).FirstOrDefault();
            LblEnDusukFiyatliUrun.Text = (from x in db.TBLURUNLER
                                         orderby x.SatisFiyat ascending
                                         select x.UrunAd).FirstOrDefault();
        }
    }
}
```

Müşteri Sayısı	Kategori Sayısı	Ürün Sayısı	Beyaz Eşya Sayısı
3	6	15	5
Toplam Stok	Bugün Satış Adedi	Bugünkü Kasa Tutarı	Toplam Kasa Tutarı
447	1	325,00 ₺	2030,00 ₺
En Yüksek Fiyatlı Ürün	En Düşük Fiyatlı Ürün	En Fazla Stoklu Ürün	En Az Stoklu Ürün
Oyuncu Bilgisayarı	Su Isıtıcı	Su Isıtıcı	Kurutma Makinesi

N Katmanlı Mimari, büyük ölçekli projelerde kullanılan, projelerin böl-parçala-yönet prensibinde olmasını sağlayan geliştirme yaklaşımıdır. Kod okunaklığı artar.

Hata yönetimi kolaylaşır.

Proje daha terli toplu olur.

Katmanlar şu şekildedir;

Entity Layer katmanında Propertyler tanımlanır.

Data Access Layer, veri erişim katmanıdır. Temel veri tabanı işlemleri (ekleme-silme-güncelleme-listeleme) işlemleri bu katmanda gerçekleşir.

Business Layer katmanında Entity Layer ve Data Acces Layer'dan gelen verileri kullanıcıya sunmadan önce hata olup olmadığının kontrolü sağlanır

Presentation Layer katmanında kullanıcının göreceği Form tasarlanır.

Kriptoloji Algoritmaları, çeşitli iletilerin veya yazıların belli bir sisteme göre şifrlenmesi ve bu mesajların güvenli bir ortamda alıcıya iletilmesi ve iletilmiş mesajın deşifre edilmesidir.

Şifreleme türleri;

MD5

SHA1

Sezar

SHA256

DES

Triple DES

RC2

Sezar Şifreleme, tarihte ilk kez Romalı lider Jül Sezar tarafından kullanılmış olan şifreleme tekniğidir.

Yer değiştirme ve harf değiştirme şifrelemesidir.

Bir yazıdaki harflerin yerlerini değiştirerek kullanılan şifreleme türüdür.

Not: Tablo içinde yer alan sütunlar C#'da Property olarak tutulurlar. SQL Veri tabanında tablo olarak tutulan yapılar C#'da Sınıf olarak tutulurlar.

QR code nedir ? (Quick Response)

Çabuk tepki kelimelerinin baş harflerinden alır.

Mobil cihazların kameralarından okutulabilen özel matris barkod (veya iki boyutlu barkod) türüdür.

1994 yılında Japon bir firma tarafından geliştirilmiştir.

Bitmap, herhangi bir sıkıştırma yapmadan resmin özelliklerini tutan ve Microsoft firmasına ait bir resim dosyası biçimidir.

Oluşturma: Encoder / Çözme: Decoder

Solid Prensipleri

-Esnek ve yönetilebilir kod yazmayı sağlayan ve kodda okunaklığı arttıran, kod karmaşasını azaltmayı hedefleyen bir yazılım object oriented geliştirme modelidir.

-Solid prensibin 5 tane temel yaklaşımı vardır.

1) Single - Responsibility Principle (Tek Sorumluluk Prensibi)

Her bir sınıfın tek bir amacı olup sadece tek bir amaca hizmet etmesidir.

2) Open - Closed Principle (Açık Kapalı Prensibi)

Kod gelişime açık olmalı fakat değişime kapalı olmalı

Sınıfa yeni davranışlar eklenebilmeli

Sınıfın temel özelliğinin değişimi mümkün olmamalı

3) Liskov Substituion Principle (Liskov'un Yerine Geçme Prensibi)

Alt sınıflardan oluşan nesnelerin üst sınıfın nesneleri ile yer değiştirdikleri zaman aynı davranışı sergilemesini beklemektir.

4) Interface Segregation Principle (Arayüz Ayrım Prensibi)

Nesneler, ihtiyaç duymadıkları metotların bulunduğu Interfacelere bağlı olmaya zorlanmamalıdır.

5) Dependency Inversion Principle (Bağımlılıkların Ters Çevrilmesi Prensibi)

Bir sınıfın, metodun ya da özelliğin, onu kullanan diğer sınıflara karşı

bağımlılığı en aza indirgenmelidir. Alt sınıfta yapılan değişiklikler üst sınıfları etkilememelidir.

Örnek

```
string sentence = "My name is Salih Toraman";
```

```
var result = sentence.Length;
```

```
Console.WriteLine(result);
```

```
Console.ReadLine();
```

Cümlelerin kaç karakterden oluştuğunu gösterir

24

Örnek

```
string sentence = "My name is Salih Toraman";
```

```
bool result2 = sentence.EndsWith("n");
```

```
Console.WriteLine(result2);
```

```
Console.ReadLine();
```

Cümlelerin "n" ile bitip bitmediğini kontrol etmeni sağlar.

True

Örnek

```
string sentence = "My name is Salih Toraman";
```

```
bool result3 = sentence.StartsWith("My name");
```

```
Console.WriteLine(result3);
```

```
Console.ReadLine();
```

Cümlelerin "My name" ile başlayıp başlamadığını kontrol etmeni sağlar.

True

Örnek

```
string sentence = "My name is Salih Toraman";
```

```
var result4 = sentence.IndexOf("Salih");
```

```
Console.WriteLine(result4);
```

```
Console.ReadLine();
```

“Salih” in kelimesinin kaçınıcı karakterden sonra başladığını gösterir.

11

Örnek

```
string sentence = "My name is Salih Toraman";
```

```
var result5 = sentence.IndexOf(" ");
```

```
Console.WriteLine(result4);
```

```
Console.ReadLine();
```

Cümlede boşluğun kaçınıcı karakterde başladığını gösterir.

2

Örnek

```
string sentence = "My name is Salih Toraman";
```

```
var result6 = sentence.LastIndexOf(" ");
```

```
Console.WriteLine(result5);
```

```
Console.ReadLine();
```

Cümlede boşluğun sondan kaçınıcı karakterde başladığını gösterir.

16

Örnek

```
string sentence = "My name is Salih Toraman";  
var result7 = sentence.Insert(0, "Hello, ");  
Console.WriteLine(result7);  
Console.ReadLine();
```

Cümlelerin başına "Hello, " ekler.

Hello, My name is Salih Toraman

Örnek

```
string sentence = "My name is Salih Toraman";  
var result8 = sentence.Substring(11);  
Console.WriteLine(result8);  
Console.ReadLine();
```

Cümleyi 11.karakterden itibaren alır yani cümleyi parçalar.

Salih Toraman

Örnek

```
string sentence = "My name is Salih Toraman";  
var result9 = sentence.ToLower();  
Console.WriteLine(result9);  
Console.ReadLine();
```

Cümleyi komple küçük harfli yapar.

my name is salih toraman

Örnek

```
string sentence = "My name is Salih Toraman";  
var result10 = sentence.ToUpper();  
Console.WriteLine(result10);  
Console.ReadLine();
```

Cümleyi komple büyük harfli yapar.

MY NAME İS SALİH TORAMAN

Örnek

```
string sentence = "My name is Salih Toraman";  
var result11 = sentence.Replace("T", "D");  
Console.WriteLine(result11);  
Console.ReadLine();
```

Cümlede belli karakterleri değiştirmek için kullanırız.

Salih Doraman

Örnek

```
string sentence = "My name is Salih Toraman";  
var result12 = sentence.Remove(2);  
Console.WriteLine(result12);  
Console.ReadLine();
```

Cümlede 2.karakterden sonrakileri cümleden atar.

My