T.C.
EGE ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

ALGORİTMA ve PROGRAMLAMA – II
(C# Sürümü)
DERS NOTLARI

Y. Doç. Dr. Aybars UĞUR

Şubat, 2004
İZMİR

C# PROGRAMLAMA

ÖRNEKLER

Değişken Tanımlama, Aritmetik İşlemler, String'ler, I/O İşlemleri, Metotlar, Diziler (Array), Denetim Yapıları (if, for, while, ...), GUI ...

ÖRNEK 1 : Ekrana Yazdırma Komutu

```
using System;

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

Ekran Çıktısı : Merhaba

ÖRNEK 2 : Klavyeden Okuma Komutu ve string

Klavyeden bir metin girilmesini bekler. "Enter" tuşuna basılınca sonlanır.

```
using System;

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

ÖRNEK 3 : Veri Tipleri, Değişkenler ve İşlemler

```
using System;

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

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

Ekran Çıktısı : 111,930002784729

ÖRNEK 4 : Tip Dönüşümleri

```
using System;

class TipDonusum
{
  public static void Main(string[] args)
  {

    double sayi = Double.Parse(Console.ReadLine());
    Console.WriteLine("Double : "+Math.Sqrt(sayi)+
      "   "+"Int : "+(int)Math.Sqrt(sayi));
  }
}
```

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

ÖRNEK 5 : İki sayıyı toplayan metot ve kullanımını içeren C# programı

```
using System;

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

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

ÖRNEK 6 : Tamsayı, Döngü, Dizi, Metot ve Ekrana Yazdırma

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

```
using System;

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

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

ÖRNEK 7 : (string'ler) Verilen bir string dizisini, ters sırada (sondan başa doğru) listeleyen C# programını yazınız.

```
Ekran Çıktısı :
Kemal
Cemil
Zekiye
Ali
```

```
using System;

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

ÖRNEK 8 : if, if else

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

```
using System;

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

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

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

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

    return false;
  }

}
```

ÖRNEK 9 : Boş bir diziye arka arkaya eleman ekleyen metodu içeren C# programını yazınız.

```
using System;

class DiziElemanEkle
{

  static string[] strDizi;
  static int elemanSayac = 0;

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

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

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

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

}
```

ÖRNEK 10 : Matrisler

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

```
using System;

class MatrisListele
{
  public static void Main(string[] args)
  { int[,] matris = { { 5,6,7,8 }, { 9, 10, 11, 12} };
    listele(matris); }

  public static void listele(int[,] matris)
  {
    for(int i=0; i<2; ++i)
    {
      for(int j=0; j<4; ++j)
        Console.Write(matris[i,j]+" ");
      Console.WriteLine();
    }
  }
}
```

```
Ekran Çıktısı :
4
30
fghijklmno
abcdefghijklmnopqrstuvwxyzab
cdeABCDEFG
Merhaba
Merhaba
```

ÖRNEK 11 : String'ler

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

BASİT ALIŞTIRMALAR

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

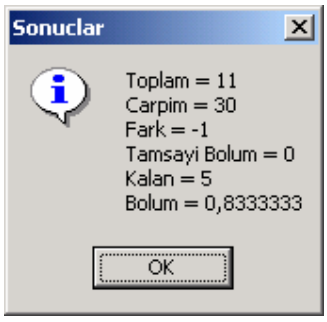
ÖRNEK 12 : Mesaj Kutusu Kullanımı

Kullanıcıdan iki tamsayı isteyerek bunların toplamını, çarpımını, farkını, bölümünü ve bölümünden kalanını bulup sonuçları yazdıran C# programı.

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

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

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

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

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

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

Ekran Çıktısı :

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

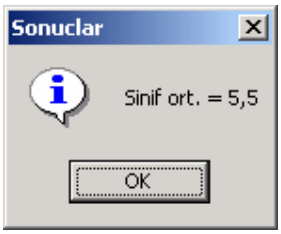


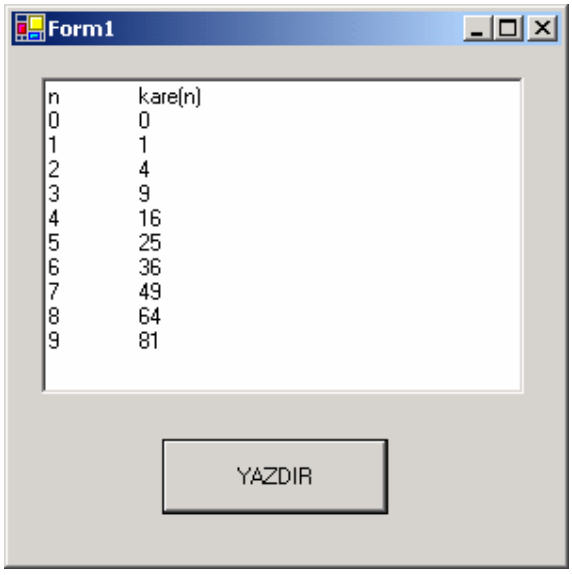
ÖRNEK 13 : While Döngüsü Kullanımı
Not ortalamasını bulan C# programı (-1 değeri girilene kadar notları okur).

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

class NotOrt
{
  public static void Main(string[] args)
  {
    float ortalama;
    int sayac=0, notu, toplam=0;

    Console.WriteLine("Notu giriniz (Exit : -1)");
    string str = Console.ReadLine();
    notu = Int32.Parse(str);

    while(notu!=-1) {
      toplam += notu; ++sayac;
      Console.WriteLine("Notu giriniz (Exit : -1)");
      str = Console.ReadLine();
      notu = Int32.Parse(str);
    };

    string s;
    if (sayac==0) s = "Not girilmedi!";
             else s = "Sinif ort. = "+(float)toplam/sayac;

    MessageBox.Show(s,"Sonuclar",
      MessageBoxButtons.OK,MessageBoxIcon.Information);

  }
}
```

Ekran Çıktısı

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





ÖRNEK 14 : GUI Bileşeni

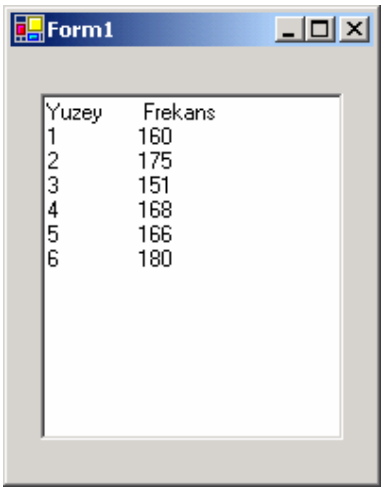
RichTextBox (Metin Kutusu)

Windows Application

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;

namespace WindowsApplication11
{
   /// <summary>
   /// Summary description for
   /// </summary>
   public class Form1 : System.Windows.Forms.Form
   {
      private System.Windows.Forms.RichTextBox richTextBox1;
      private System.Windows.Forms.Button button1;
      private System.ComponentModel.Container components =
        null;

      public Form1()
      { InitializeComponent(); }


      /// <summary>
      /// The main entry point for the application.
      /// </summary>
      [STAThread]
      static void Main()
      {
         Application.Run(new Form1());
      }

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

      private void button1_Click(object sender,
        System.EventArgs e)
      {
         string str = "n"+"\t"+"kare(n)\n";

         for(int i=0; i<10; ++i)
            str+=""+i+"\t"+kare(i)+"\n";

         richTextBox1.Text = str;
      }
   }
}
```

ÖRNEK 15 : Random sayı üretme ve Kullanma

Altı yüzlü bir zarın 1000 kere atılması sonucu her bir yüzün kaçar kere geldiğini bularak listeleyen C# Programı.



Windows Application

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

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

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

  metAlan.ReadOnly = true;

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


}
```

ÖRNEK 16 : Cep Telefonu


```
private void button1_Click(object sender,
System.EventArgs e)
{
   textBox1.Text += 1;
}

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

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

NESNE YÖNELİMLİ PROGRAMLAMA
OBJECT ORIENTED PROGRAMMING

Sınıf Örneği : Bir Rasyonel Sayı Sınıfı, r1 Nesnesi oluşturulması ve kullanılması.

```
using System;

class Rasyonel_sayi
{
  long pay;
  long payda;

  public Rasyonel_sayi()
  { pay = 2; payda = 3; }

  public Rasyonel_sayi(long pay, long payda)
  { this.pay = pay; this.payda = payda; }

  public void yazdir()
  { Console.WriteLine("{0}/{1}",pay,payda); }
}

class main
{
  public static void Main()
  {
    Rasyonel_sayi r1 = new Rasyonel_sayi();
    r1.yazdir();
  }
}
```

Sonuç : 2/3

TEMEL BİLGİ ve TERMİNOLOJİ

Sınıf (Class) : Soyut bir veri tipinin hem verilen tiplerdeki veriler kümesini, hem de bu değerler üzerinde yapılabilecek işlemler kümesini bir araya getirir. Örnek : "Rasyonel_sayi" sınıfı.

Nesne (Object) : Sınıf tipindeki değişkenlere nesne adı verilir.
Örnek : "r1" nesnesi.

Metot (Method) : Bir eylemi veya işlemi gerçekleştiren sınıf üyesidir. "Rasyonel_sayi()" yapıcı metotları ve "yazdir()" metodu Rasyonel_sayi sınıfının metotlarıdır.

Sınıf Üyeleri (Class Members) : Sınıfın elemanlarına üye adı verilir. Değişkenler, metotlar ...
Örnekler : "pay", "payda" değişkenleri; " Rasyonel_sayi()" yapıcı metotları ve "yazdir()" metodu "Rasyonel_sayi" sınıfının üyeleridir.

Yapıcı metot (Constructor) : Sınıftan yeni bir nesne yaratıldığı anda çağrılan metoda yapıcı metot adı verilir. Yapıcı metot ismi, sınıf ismi ile aynıdır.

```
Rasyonel_sayi r1 = new Rasyonel_sayi(3,5);
```

"r1" nesnesi "new" deyimi ile oluşturulurken " Rasyonel_sayi" sınıfının iki tane "long" parametre alan yapıcı metodu devreye girer.

```
// Yapıcı metot
public Rasyonel_sayi(long pay, long payda)
{ this.pay = pay; this.payda = payda; }
```

ARAMA (SEARCH)

ARDIŞIK (DOĞRUSAL) ARAMA (LINEAR SEARCH)

```
using System;

public class LinearSearcher
{

  public static void Main(string[] args)
  {
    int[] dizi = { 1, 3, 5, 7, 9, 11 };

    int aranan = Int32.Parse(Console.ReadLine());
    int indis  = LinearSearch(dizi, aranan);

    if (indis!=-1)
      Console.WriteLine(indis+" . konumda bulundu");
    else
      Console.WriteLine("bulunamadı");
  }

  public static int LinearSearch(int[] dizi, int anahtar)
  {
    for(int i=0; i<dizi.Length; i++)
      if(dizi[i]==anahtar) return i;

    return -1;
  }

}
```

Ekran Çıktısı :

C:\ALG>LinearSearch
4
bulunamadı

C:\ALG>LinearSearch
5
2 . konumda bulundu

İKİLİ ARAMA
BINARY SEARCH

```
using System;

public class BinarySearchTest
{
 public static void Main(string[] args)
 {
  int[] dizi = { 1, 3, 5, 7, 9, 11 };
  string mesaj;

  int aranan =
    Int32.Parse(Console.ReadLine());
  int indis  = BinarySearch(dizi, aranan);

  if (indis!=-1)
    mesaj = indis+" . konumda bulundu";
  else
    mesaj ="bulunamadı";

  Console.WriteLine(mesaj);
 }

 public static int BinarySearch(int[] dizi, int anahtar)
 {
  int bas=0; int son=dizi.Length-1;  int orta;

  while(bas<=son)
  {
    orta = (bas+son)/2;

    if (anahtar==dizi[orta]) return orta;
    else if(anahtar<dizi[orta])
      son = orta-1;
    else
      bas = orta+1;
  }
  return -1;
 }
}
```

Ekran Çıktısı :

```
C:\ALG>BinarySearch
5
2 . konumda bulundu

C:\ALG>BinarySearch
4
bulunamadı
```

SIRALAMA (SORTING)

BUBBLE SORT (KABARCIK SIRALAMASI)

```
using System;

public class BinarySearchTest
{
 public static void Main(string[] args)
 {
  int[] a = { 2,6,4,8,10,12,89,68,45,37 };

  string str = "Veriler (Sıralanmadan Önce)    :";
  for(int i=0; i<a.Length; ++i) str+=" "+a[i];
  Console.WriteLine(str);

  BubbleSort(a);

  str = "Veriler küçükten büyüğe sıralı :";
  for(int i=0; i<a.Length; ++i) str+=" "+a[i];
  Console.WriteLine(str);
 }

 public static void BubbleSort(int[] b)
 {
  for(int tur=1; tur<b.Length; ++tur)
    for(int i=0; i<b.Length-1; ++i)
      if(b[i]>b[i+1]) Swap(b,i);
 }

 public static void Swap(int[] c, int ilk)
 {
  int temp=c[ilk];
  c[ilk]=c[ilk+1];
  c[ilk+1]=temp;
 }
}
```

```
Veriler (Sıralanmadan Önce)    : 2 6 4 8 10 12 89 68 45 37
Veriler küçükten büyüğe sıralı : 2 4 6 8 10 12 37 45 68 89
```

SELECTION SORT (SEÇMELİ SIRALAMA)

```
using System;

class SelectionSort
{
  public static void Main(string[] args)
  {
    int i, j, tur, yer;
    string[] dizi =
    { "Ali", "Cemil", "Veli", "Abdullah", "Kemal" };
    string enkVeri, temp;

    Console.WriteLine();
    for(i=0; i<5; ++i)
      Console.Write(dizi[i]+" ");

    for(tur=0; tur<4; ++tur)
    {
      enkVeri = dizi[tur];
      yer = tur;
      for(j=tur+1; j<5; ++j)
      {
        if(dizi[j].CompareTo(enkVeri)<0)
        { yer = j;
          enkVeri=dizi[yer]; };
      }

      temp = dizi[tur];
      dizi[tur] = enkVeri;
      dizi[yer] = temp;
    };

    Console.WriteLine();
    for(i=0; i<5; ++i)
      Console.Write(dizi[i]+" ");
  }
}
```

Ali Cemil Veli Abdullah Kemal
Abdullah Ali Cemil Kemal Veli

BAĞLAÇLI LİSTELER
LINKED LISTS

LİSTELER

Günlük yaşamda listeler pek çok yerde kullanılmaktadır. Alışveriş listeleri, adres listeleri, davetli listeleri gibi. Bilgisayar programlarında da listeler yararlı ve yaygın olarak kullanılan veri yapılarındandırlar. Programlama açısından liste, aralarında doğrusal ilişki olan veriler topluluğu olarak görülebilir.

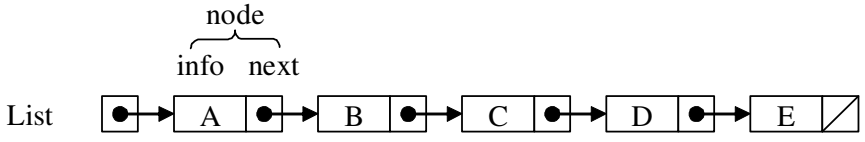
BAĞLAÇLI LİSTELER

Kendi tipindeki bir yapıyı gösteren bir işaretçi üyesine sahip yapılara self-referential structures adı verilir. Örnek olarak :

```
class Dugum //node
{
  private string veri; // info
  private Dugum sonraki; // next
.....
}
```

yapısı, veri adlı string tipli bilgi elemanının yanında, bir düğüm yapısında bir bellek bölgesine işaret eden sonraki (next) işaretçisine sahiptir. Bu tür yapıların arka arkaya birbirine bağlanması mantığı listelerde oldukça yararlıdır.

Listedeki her düğümde bir sonraki düğümün adresinin tutulduğu veri yapısı (doğrusal) bağlı liste olarak adlandırılır (Şekil 1). Listenin her bir elemanına düğüm (node) adı verilir. Düğümler, bilgi ve bağ (adres) sahalarından oluşmaktadırlar. Bağ sahalarında referanslar kullanılmaktadır. Listenin ilk elemanına dışarıdan bir referans (list) ile erişilmektedir. Diğer düğümlere de bağlar yardımı ile ulaşılabilmektedir. Son düğümün sonraki adres (sonraki) sahası NULL değerini içerir. NULL bağı, liste sonunu belirtir. Elemanı olmayan liste boş liste olarak adlandırılır.



Şekil 1 : Doğrusal Bağlaçlı Liste

Sıralı bellek kullanımının (dizi) en büyük dezavantajı, hiç kullanılmasa veya az kullanılsa bile sabit miktardaki belleğin bu yapılara ayrılmış olarak tutulmasıdır. Ayrıca sabit bellek miktarı aşıldığında da taşma oluşması ve eleman ekleme işleminin yapılamamasıdır. Bağlaçlı listeler üzerinde gerçekleştirildiklerinde ise bu problemler ortadan kalkmaktadır. Bellekten sadece gerektiği kadar yer ayrılmakta ve bellek boyutu bitene kadar bu yapılara ekleme işlemi yapılabilmektedir.

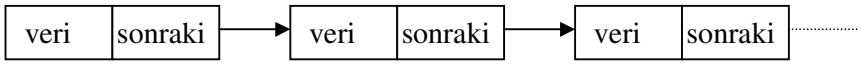
Bağlaçlı listeler, başka veri yapılarının gerçekleştiriminde kullanılabildikleri gibi kendileri de veri yapısıdırlar. Bağlaçlı listelerde elemanların eklenme ve çıkarılmasında bir sınırlama yoktur. Başa ve sona olduğu gibi araya da eleman eklenebilir; baştan ve sondan olduğu gibi ortadan da eleman çıkarılabilir. Bağlaçlı liste dolaşılarak herhangi bir elemanına erişilebilir. Bir bağlı listenin n. elemanına erişmek için n tane işlem yapmak yani kendinden önceki (n-1) eleman üzerinden geçmek gerekmektedir. Elemanların bellekteki yerleri dizilerdeki gibi sıralı olmadığından elemanlar ve sıraları ile yerleştikleri bellek bölgeleri arasında bir ilişki yoktur.

Bağlaçlı listelerin diziler üzerine avantajı, bir grup eleman arasına eleman eklemede ve bir grup eleman arasından eleman çıkarmada ortaya çıkar.

Dizilerde bir eleman silerken arada boşluk kalmasını engellemek için ilerisindeki (sağındaki) tüm elemanları bir geriye (sola) kaydırmak gerekir. Eleman eklemede de yer açmak için konulacağı yerdeki ve ilerisindeki elemanları bir ileriye (sağa) kaydırmak gerekecektir. Kaç tane elemanın yer değiştireceği (birer kaydırılacağı) dizi boyutuna bağlı olarak ve eklenecek elemanın yerine bağlı olarak değişecektir. Bağlaçlı listelerde ise eleman ekleme ve çıkarma için yapılan iş liste boyutundan bağımsızdır.

Bağlaçlı Listede Kullanılan Sınıflar :

Dugum : Tek düğüme ilişkin veriler ve işlemler
Liste : Listeye ilişkin veri ve işlemler
ListeTest : Listenin test edilmesi




```
using System;

// Dugum Sınıfı
class Dugum
{
  private string veri;
  private Dugum sonraki;

  public Dugum(string str):this(str,null){ }

  public Dugum(string str, Dugum sonrakiDugum)
  {
    veri = str;
    sonraki = sonrakiDugum;
  }

  public Dugum Sonraki
  {
    get { return sonraki; }
    set { sonraki=value; }
  }

  public string Veri
  {
    get { return veri; }
  }

}
```

```
// Liste Sınıfı
class Liste
{
 private Dugum bas;
 private Dugum son;

 private string etiket;

 public Liste(string listeAd)
 {
  etiket = listeAd;
  bas=son=null;
 }

 public Liste():this("Liste")
 { }


 public bool IsEmpty()
 { return bas==null; }

 public void basaEkle(string eklenecekEleman)
 {
  if(IsEmpty())
    bas=son=new Dugum(eklenecekEleman);
  else
    bas=new Dugum(eklenecekEleman, bas);
 }

 public void sonaEkle(string eklenecekEleman)
 {
  if(IsEmpty())
    bas=son=new Dugum(eklenecekEleman);
  else
    son=son.Sonraki=new Dugum(eklenecekEleman);
 }
```

```
public string bastanSil()
{
  string deger = null;
  if(!IsEmpty())
  {
    deger = bas.Veri;

    if(bas==son)
      bas=son=null;
    else
      bas=bas.Sonraki;
  }
  return deger;
}

public string sondanSil()
{
  string deger = null;
  if(!IsEmpty())
  {
    deger = son.Veri;

    if(bas==son)
      bas=son=null;
    else
    {
      Dugum etkin = bas;
      while(etkin.Sonraki!=son)
        etkin=etkin.Sonraki;
      son=etkin;
      etkin.Sonraki=null;
    }
  }
  return deger;
}
```

```
  public void Yazdir()
  {
    if(IsEmpty())
    { Console.WriteLine("Boş Liste"); return; };

    Dugum etkin = bas;

    while(etkin!=null)
    {
      Console.Write(etkin.Veri+" ");
      etkin=etkin.Sonraki;
    }

    Console.WriteLine("\n");
  }
}
```

// ListeTest Sınıfı

```
class ListeTest
{
  public static void Main(string[] args)
  {
    Liste liste = new Liste();

    for(int i=1; i<5; i++)
      liste.basaEkle(""+i);
    liste.Yazdir();
    for(int i=5; i<10; ++i)
      liste.sonaEkle(""+i);
    liste.Yazdir();
  }
}
```

Ekran Çıktısı :

```
C:\ALG>LinkedList
4 3 2 1

4 3 2 1 5 6 7 8 9
```

KOLEKSİYON SINIFLARI
COLLECTION CLASSES

ArrayList
Stack
Queue
.....

Hazır Yapılar ve Önemi

Hazır Sıralama :

```
using System;

class BuiltInSort
{
  public static void Main(string[] args)
  {
    string[] dizi =
    { "Ali", "Cemil", "Veli", "Abdullah", "Kemal" };

    Array.Sort(dizi);

    for(int i=0; i<dizi.Length; ++i)
      Console.WriteLine(dizi[i]);

  }
}
```

DOSYALAR
1. SIRADAN ERİŞİMLİ DOSYALAR
2. DOĞRUDAN ERİŞİMLİ DOSYALAR (DERSTE)

SIRADAN ERİŞİMLİ DOSYALAR BASİT ÖRNEK

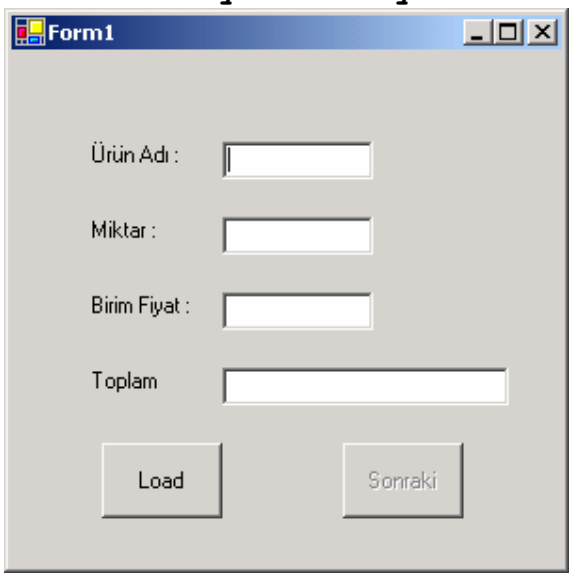
KLAVYEDEN DİSKE YAZDIRAN PROGRAM :
STREAMWRITER ile DOSYAYA YAZMA
(using System.IO;)

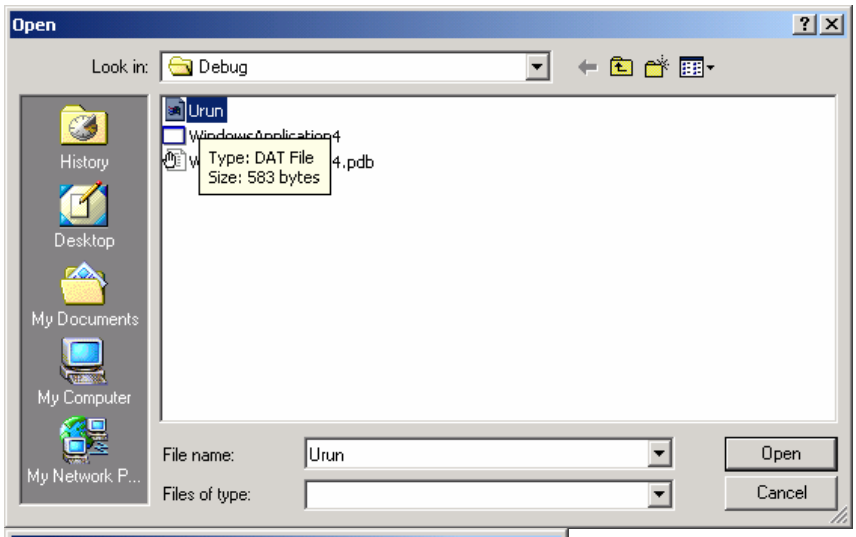
```
string str;
FileStream fout;

try
{
   fout=new FileStream ("test.txt", FileMode.Create);
}
catch (IOException exc)
{
   Console.WriteLine(exc.Message+"Dosya Acilamadi");
   return;
}

StreamWriter fstr_out = new StreamWriter(fout);

Console.WriteLine("Metni girin.");
do
{
   Console.Write(": ");
   str=Console.ReadLine();
   if(str!="q")
   {
      str+="\r\n"; // Yeni satir ekle
      try
      {
         fstr_out.Write(str);
      }
      catch(IOException exc)
      {
         Console.WriteLine(exc.Message+"Dosyada Hata");
         return;
      }
   };
} while(str.ToLower()!="q");

fstr_out.Close();
}
```

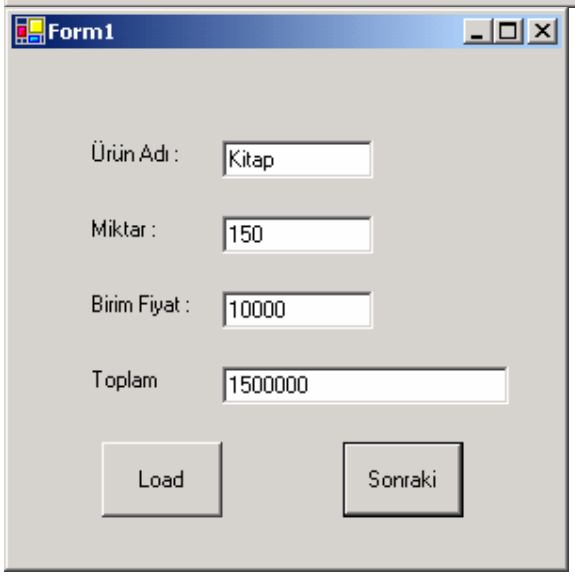
DİSKTEN EKRANA YAZDIRAN PROGRAM : STREAMREADER ile DOSYADAN OKUMA

```
using System;
using System.IO;

class DosyaOku
{
  public static void Main(string[] args)
  {

   string s;
   FileStream fin;

   try
   {
      fin=new FileStream ("test.txt", FileMode.Open);
   }
   catch (FileNotFoundException exc)
   {
      Console.WriteLine(exc.Message+"Dosya Acilamadi");
      return;
   }

   StreamReader fstr_in = new StreamReader(fin);

   while((s=fstr_in.ReadLine())!=null) {
     Console.WriteLine(s);
   }

   fstr_in.Close();
  }
}
```

ÖRNEK : Dosyadaki Kayıtları Tek Tek Listeleyen Program







```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;

using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Serialization;
using System.IO;

namespace WindowsApplication4
{
    /// <summary>
    /// Summary description for Form1.
    /// </summary>
    public class Form1 : System.Windows.Forms.Form
    {
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.TextBox textBox2;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.TextBox textBox4;
        private System.Windows.Forms.Button bLoad;
        private System.Windows.Forms.Button bSonraki;
        private System.ComponentModel.Container components = null;

        public Form1()
        { InitializeComponent(); }

        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.Run(new Form1());
        }

        public void FormuTemizle()
        {
            for(int i=0; i<Controls.Count; i++)
            {
                Control etkin = Controls[i];
                if(etkin is TextBox) etkin.Text="";
            }
        }
        public void MetinleriYaz(string[] degerler)
        {
            textBox1.Text = degerler[0];
            textBox2.Text = degerler[1];
            textBox3.Text = degerler[2];
            textBox4.Text =
                ""+ ((Double.Parse(degerler[1]))*
                    (Int32.Parse(degerler[2])));
        }
```

```
public string[] MetinleriAl()
{
  string[] degerler = new string[4];

  degerler[0] = textBox1.Text;
  degerler[1] = textBox2.Text;
  degerler[2] = textBox3.Text;
  degerler[3] = textBox4.Text;

  return degerler;

}

private FileStream input;
private BinaryFormatter reader = new BinaryFormatter();

private void bLoad_Click(object sender, System.EventArgs e)
{
    OpenFileDialog fileChooser = new OpenFileDialog();
    DialogResult  result = fileChooser.ShowDialog();
    string fileName;
    if(result==DialogResult.Cancel) return;
    fileName = fileChooser.FileName;
    FormuTemizle();
    if (fileName=="" || fileName== null)
      MessageBox.Show("Hatalı Dosya Adı","Hata",
        MessageBoxButtons.OK,MessageBoxIcon.Error);
    else
    { input = new FileStream(fileName, FileMode.Open,
                             FileAccess.Read);
      bSonraki.Enabled = true;
    }
}

private void bSonraki_Click(object sender, System.EventArgs e)
{
    try
    {
      Kayit kayit =
        (Kayit)reader.Deserialize(input);

      string[] degerler = new string[]
      { kayit.UrunAd.ToString(),
        kayit.Miktar.ToString(),
        kayit.BrFiyat.ToString() };
      MetinleriYaz(degerler);
    }
    catch (SerializationException)
    {
      input.Close();
      bLoad.Enabled = true;
      bSonraki.Enabled = false;
      FormuTemizle();
      MessageBox.Show("Dosya Sonu","",
        MessageBoxButtons.OK,MessageBoxIcon.Information);
  }
}

private FileStream output;
private BinaryFormatter formatter = new BinaryFormatter();
```

```
        private void Form1_Load(object sender, System.EventArgs e)
        {
          Kayit kayit = new Kayit();

          output = new FileStream ("Urun.dat",
            FileMode.OpenOrCreate, FileAccess.Write);

          string[,] degerler =
            new string[3,3] { { "Kitap","15.0","10000"},
                              {"Defter","50.0","3000"},
                              {"Kalem","200.0","120"}};
              for (int i=0; i<3; ++i)
              { kayit.UrunAd = degerler[i,0];
                kayit.Miktar = Double.Parse(degerler[i,1]);
                kayit.BrFiyat = Int32.Parse(degerler[i,2]);
                formatter.Serialize(output,kayit);
              }

              output.Close ();
        }
    }

    [Serializable]
    public class Kayit
    {

        private string urunAd;
        private double miktar;
        private int brFiyat;

        public Kayit() : this("",0.0,0)
        {}

        public Kayit(string adDegeri,
            double miktarDegeri, int brFiyatDegeri)
        {
            urunAd = adDegeri;
            miktar = miktarDegeri;
            brFiyat = brFiyatDegeri;
        }

        public string UrunAd
        {   get { return urunAd; }
            set { urunAd = value; }
        }

        public double Miktar
        {   get { return miktar; }
            set { miktar = value; }
        }

        public int BrFiyat
        {   get { return brFiyat; }
            set { brFiyat = value; }
        }
    }
}
```