ArrayList

Example

ArrayList list = new ArrayList();

* ArrayList sınıfı IList, IEnumerable ve ICollection arayüzlerini uygulamaktadır.
* IList arayüzünü uyguladığı için ArrayList’in elemanlarına indeksle erişilebilmektedir.
* ArrayList sınıfı System.Collections’da tanımlıdır.
* ArrayList’in kapasitesi belirtilmediği zaman bunun 0 olduğu varsayılmaktadır. İlk eleman eklendiğinde kapasite 4 yapılır.
* ArrayList tanımlanırken kapasitesi yapıcı metoda parametre olarak gönderilebilir.
* Koleksiyonun kapasitesi Capacity özelliğinde tutulmaktadır.

Example

ArrayList list = new ArrayList(10);

Console.WriteLine(list.Capacity); //10

* ArrayList’e eleman eklenirken IList kaynaklı Add() metodu kullanılmaktadır. Bu metod object türünde bir parametre almakta ve geriye koleksiyona eklediği elemanın index numarasını döndürmektedir.
* Add() metodu parametre olarak aldığı bilgiyi koleksiyonun sonuna eklemektedir.

Example

ArrayList list = new ArrayList();

list.Add(1);

list.Add(true);

list.Add("ArrayList");

Console.WriteLine(list[0]); //1

Console.WriteLine(list[1]); //True

Console.WriteLine(list[2]); //ArrayList

* ArrayList’e eklenen elemanlar gerçekte object türünde dizi değişkene aktarılmaktadır. Eklenen eleman object türünde değilse geri planda dönüştürme işlemi yapılmaktadır. (boxing)
* ArrayList’den okunan veriler kullanılmadan önce tip dönüşümü yapılmalıdır. (unboxing)
* Okuma ve yazma sırasında veriler boxing-unboxing işlemlerine maruz kalmakta bu da performans kaybına sebebiyet vermektedir.
* Koleksiyonlara tanımlama sırasında eleman ekleyebilirsiniz.

Example

ArrayList numbers = new ArrayList() { 1, 2, 3 };

* Bir elemanı koleksiyonun sonu yerine herhangi bir konuma eklemek için Insert() metodu kullanılmaktadır. Bu metod iki parametreye sahiptir. İlk parametre sıra numarası(index) ikinci parametre ise elemanın içeriğidir.

Example

ArrayList fruits = new ArrayList();

fruits.Add("Apple");

fruits.Add("Plum");

fruits.Add("Cherry");

//3. eleman Melon olur. 3. sırada bulunan eski eleman ise artık 4. eleman olacaktır.

fruits.Insert(2, "Melon");

Insert() metodu ile koleksiyona eleman eklenirken dikkatli olunmalıdır. Örneğin aşağıdaki kod hataya neden olacaktır.

Example

ArrayList teams = new ArrayList(10);

teams.Add("Fenerbahçe");

teams.Add("Galatasaray");

teams.Add("Beşiktaş");

teams.Insert(4, "Trabzonspor");

* ICollection arayüzünü uygulamış sınıflardan yola çıkılarak hazırlanan nesneler ArrayList’in yapıcı metoduna parametre olarak verilebilmektedir.

Example

string[] cities = new string[3];

cities[0] = "Edirne";

cities[1] = "İstanbul";

cities[2] = "Ankara";

ArrayList cityList = new ArrayList(cities);

* ICollection arayüzünü uygulamış sınıflardan yola çıkılarak hazırlanan nesneleri ArrayList’in herhangi bir konumuna eklerken InsertRange() metodu kullanılmaktadır.

Example

string[] cities = new string[3];

cities[0] = "Edirne";

cities[1] = "İstanbul";

cities[2] = "Ankara";

ArrayList cityList = new ArrayList();

cityList.InsertRange(0, cities);

* ArrayList içerisinde arama yapılırken IndexOf() ve LastIndexOf() metodları kullanılmaktadır. Bu metodlar üç parametreye sahiptir. İlk parametre aranacak olan değerdir, ikinci parametre başlangıç indeksidir, üçüncü parametre ise aramaya dahil edilecek eleman sayısıdır.
* ArrayList’in eleman sayısı Count özelliğinde tutulmaktadır.

Example

ArrayList numbers = new ArrayList(10);

numbers.Add(1);

numbers.Add(2);

numbers.Add(3);

Console.WriteLine(numbers.Count); //3

Console.WriteLine(numbers.Capacity); //10

* ArrayList’in kapasitesi eleman sayısına eşitlenebilir. Bunun için TrimToSize() metodu kullanılmaktadır.

Example

ArrayList numbers = new ArrayList(10);

numbers.Add(1);

numbers.Add(2);

numbers.Add(3);

numbers.TrimToSize();

Console.WriteLine(numbers.Count); //3

Console.WriteLine(numbers.Capacity); //3

* ArrayList içerisinden eleman silinirken RemoveAt() metodu kullanılmaktadır. Bu metod parametre olarak silinecek elemanın sıra numarasını almaktadır.
* ArrayList içerisinden eleman silinirken Remove() metodu kullanılabilir. Bu metod parametre olarak silinecek elemanın içeriğini almaktadır.
* ArrayList’in tüm elemanlarını silmek istiyorsanız Clear() metodunu kullanabilirsiniz.
* ArrayList içerisinden belirli konumdan belirli sayıda elemanı silebilirsiniz. Bunun için RemoveRange() metodu kullanılır.
* ArrayList sınıfı IEnumerator arayüzünü uyguladığı için bu arayüzde bulunan GetEnumerator() metodu ile ArrayList’in elemanlarını numaralı bir listeye alabiliriz.

Example

ArrayList numbers = new ArrayList(10);

numbers.Add(1);

numbers.Add(2);

numbers.Add(3);

IEnumerator enumerator = numbers.GetEnumerator();

Bu liste üzerinde MoveNext() metodu ile dolaşabiliriz. Bu metod geriye bool türünde değer döndürmektedir.

Example

ArrayList numbers = new ArrayList(10);

numbers.Add(1);

numbers.Add(2);

numbers.Add(3);

IEnumerator enumerator = numbers.GetEnumerator();

while (enumerator.MoveNext())

{

Console.WriteLine(enumerator.Current);

}

MoveNext() metodu son elemanın üzerindeyken kullanıldığında geriye false değer döner.

Numaralı listenin ilk elemanının üzerine gidilmek istendiğinde Reset() metodu kullanılmaktadır.

Stack

Example

Stack stack = new Stack();

* Stack türündeki koleksiyonlar son giren ilk çıkar mantığı ile çalışmaktadır.
* Stack nesneleri object türünde bilgi sakladıkları için her türde veri aktarılabilmektedir.
* Stack sınıfı ICloneable, IEnumerable ve ICollection arayüzlerini uygulamaktadır.
* Stack nesnesi oluştururken kapasiteyi yapıcı metoda parametre olarak verebilirsiniz.
* Stack nesnesine eleman eklenirken Push() metodu kullanılmaktadır.
* Stack nesnesinde tutulan elemanlar okunurken Pop() metodu kullanılmaktadır.

Example

Stack numbers = new Stack();

numbers.Push(1);

numbers.Push(2);

numbers.Push(3);

Console.WriteLine(numbers.Count); //3

Console.WriteLine(numbers.Pop()); //3

Console.WriteLine(numbers.Pop()); //2

Console.WriteLine(numbers.Pop()); //1

Console.WriteLine(numbers.Count); //0

* Push() metodu her kullanıldığında Stack nesnesinin eleman sayısı 1 artar. Pop() metodu her kullanıldığında Stack nesnesinin eleman sayısı 1 azalır.
* Eleman okunurken silme işleminin yapılmasını istemiyorsanız Pop() yerine Peek() metodunu kullanabilirsiniz.
* Stack nesnesi içerisindeki tüm elemanları silmek istiyorsanız Clear() metodunu kullanabilirsiniz.
* Mevcut bir Stack nesnesinden yararlanarak aynı içeriğe sahip ikinci bir Stack nesnesi oluşturmak istiyorsanız Clone() metodunu kullanabilirsiniz.

Example

Stack stack1 = new Stack();

stack1.Push(1);

stack1.Push(2);

stack1.Push(3);

Stack stack2 = (Stack) stack1.Clone();

Queue

Example

Queue queue = new Queue();

* Queue türündeki koleksiyonlar ilk giren ilk çıkar mantığı ile çalışmaktadır.
* Queue sınıfı ICloneable, IEnumerable ve ICollection arayüzlerini uygulamaktadır.
* Queue nesnesi oluştururken kapasiteyi yapıcı metoda parametre olarak verebilirsiniz.
* Queue nesnesine eleman eklenirken Enqueue() metodu kullanılmaktadır.
* Queue nesnesinden eleman çıkarma işlemi Dequeue() metodu ile yapılmaktadır.

Example

Queue numbers = new Queue();

numbers.Enqueue(1);

numbers.Enqueue(2);

numbers.Enqueue(3);

Console.WriteLine(numbers.Dequeue()); //1

Console.WriteLine(numbers.Dequeue()); //2

Console.WriteLine(numbers.Dequeue()); //3

* Eleman okunurken silme işleminin yapılmasını istemiyorsanız Peek() metodunu kullanabilirsiniz.
* Queue nesnesinin elemanlarını GetEnumerator() metodu ile numaralı bir listeye aktarabiliriz. Bu sayede Queue nesnesinin eleman sayısı değişmeden veriler üzerinde çalışabiliriz.

Example

Queue numbers = new Queue();

numbers.Enqueue(1);

numbers.Enqueue(2);

numbers.Enqueue(3);

IEnumerator enumerator = numbers.GetEnumerator();

while (enumerator.MoveNext())

{

Console.WriteLine(enumerator.Current);

}

Console.Write(numbers.Count); //3

IList

Example

public interface IList : ICollection, IEnumerable

{

object this[int index] { get; set; }

bool IsReadOnly { get; }

bool IsFixedSize { get; }

int Add(object value);

bool Contains(object value);

void Clear();

int IndexOf(object value);

void Insert(int index, object value);

void Remove(object value);

void RemoveAt(int index);

}

IEnumerable

Example

public interface IEnumerable

{

public IEnumerator GetEnumerator();

}

IEnumerator

Example

public interface IEnumerator

{

public bool MoveNext();

public void Reset();

public object Current { get; }

}

* foreach döngüsü IEnumerator arayüzünden yararlanarak veri okuma işlemi yapmaktadır.