**DİZİLER**

|  |  |  |
| --- | --- | --- |
|  | **DİZİLER** | **KOLLEKSİYONLAR** |
| ***Görevi*** | **Aynı Tipte Verileri saklamak** | **Aynı ve Farklı Tipte Verileri saklamak** |
| ***Eleman Sayıları*** | **Belirlenmeli değiştirmek için Array.Resize** | **Belirtmeden sonsuz(eklendiği kadar yer kaplar)** |
| ***İsim Uzayları*** | **System.Array** | **System.Collections** |
| ***Hızları*** | **Daha hızlıdır** | **Daha Yavaştır.** |
| ***Boxing & Unboxing*** | **Dönüştürme yok** | **Dönüştürme var** |

**İlk Uygulama**

string[] adsoyad = new string[3];

adsoyad[0] = "Talha Tektaş";

adsoyad[1] = "Günay Tektaş";

adsoyad[2] = "Azra Uncu";

//foreach (string k in adsoyad) Console.WriteLine(k);

**Dizi Tanımlarken Atama**

string[] adsoyad ={ "Volkan","Talha","Sami" }; //Tanımlarken Atama

char[] karakter = { '!', '?' }; //Karakter dizisi Tanımlama

//foreach (char k in karakter) Console.WriteLine(k);

**Forsuz deneme**

int[] dizi = new int[51];

dizi[0] = 0;

dizi[1] = 2;

dizi[2] = 4;

dizi[3] = 6;

//...

dizi[50] = 100;

**Döngüyle Dizi Ataması**

int i,j=0;

int[] dizi = new int[51];

for (i = 0; i <= 100; i += 2)

{

dizi[j] = i;

j++;

}

for (i = 0; i <= 50; i ++){Console.WriteLine(i+" "+dizi[i]);}

//foreach (int deger in dizi){Console.WriteLine("Dizi Değeri= "+deger);}

**Çok Boyutlu Dizi**

string[,] adsoyad = new string[3, 3];

string[,] adsoyad = {{"Volkan","Karakaş"},{"cemil","ipekçi"},{"Dominos","Acikti"}};//2\*3=6

Console.WriteLine(adsoyad[1,0]); //cemil

**Dikdörtgensel Diziler**

int[,] dizi = new int[3, 4] {

{ 1, 2, 3, 4 },

{ 11, 22, 33, 44 },

{ 111, 222, 333, 444 }

};//3\*4 dikdörtgensel dizi

for (int i = 0; i < **dizi.GetLength(0)**; i++)

{

for (int j = 0; j < **dizi.GetLength(1)**; j++) Console.Write(dizi[i, j] + "\t");

Console.WriteLine();

}

**rectangular\_array**

int[,] dizi = new int[3, 4]

{ { 1, 2, 3, 4 }, { 1, 2, 3, 4 }, { 1, 2, 3, 4 } };

for (int i = 0; i < dizi.GetLength(0); i++)

{

for (int j = 0; j < dizi.GetLength(1); j++)

{

Console.Write(dizi[i,j]);

if (j == dizi.GetLength(1)-1)

Console.WriteLine();

}

}

Console.ReadLine();

**Düzensiz Diziler(Jagged Array)**

int[][] duzensiz\_dizi = { new int[] { 1},

new int[] { 1, 2, 3 },

new int[] { 1, 2 } };

foreach (var satir in duzensiz\_dizi)

{

foreach (var eleman in satir)

Console.Write("{0} ", eleman);

Console.WriteLine();

}

Console.ReadLine();

**jagged\_array\_listeleme**

int[][] duzensiz\_dizi = new int[3][]; //Tek Boyutlu 3 satırdan oluşan dizi

//Her satırda kaç eleman olacağı belirlendi.

duzensiz\_dizi[0] = new int[1];

duzensiz\_dizi[1] = new int[3];

duzensiz\_dizi[2] = new int[2];

duzensiz\_dizi[0][0] = 1;

duzensiz\_dizi[1][0] = 1;

duzensiz\_dizi[1][1] = 2;

duzensiz\_dizi[1][2] = 3;

duzensiz\_dizi[2][0] = 1;

duzensiz\_dizi[2][1] = 2;

foreach (var satir in duzensiz\_dizi)

{

foreach (var eleman in satir)

Console.Write("{0} ", eleman);

Console.WriteLine();

}

Console.ReadLine();

**Array\_Resize & new string**

string[] adsoyad = new string[2];

adsoyad[0] = "Talha";

adsoyad[1] = "Sümeyye";

Console.WriteLine("Dizinin İlk Elemanının Değeri = " + adsoyad[0]);

adsoyad = new string[3]; //İçindeki bilgiler temizlenerek boyutu büyüyor

//Array.Resize(ref adsoyad, 3); //İçindeki bilgiler kalarak boyutu büyüyor

//Array.Resize(ref dizi değişkeni adı, yeni sizi boyutu);

adsoyad[2] = "Ömer";

Console.WriteLine("Dizinin İlk Elemanının Değeri = " + adsoyad[0]);

Console.WriteLine("Dizinin Son Elemanının Değeri = " + adsoyad[2]);

Console.ReadKey();

**DİZİ DEĞİŞKENLERİNDE KULLANILAN METODLAR**

**Array\_CreateInstance**

Array okulturu = Array.CreateInstance(typeof(string), 3);

string[] okulturleri = (string[])okulturu;

okulturleri[0] = ("Endüstri Meslek");

okulturleri[1] = ("Teknik");

okulturleri[2] = ("Anadolu Teknik");

Console.Write(okulturleri[1]);

Console.ReadKey();

**Array\_IsArray**

string[] okulturu = { "Anadolu", "Meslek", "Teknik" };

string ad = "Volkan";

Type tip;

tip = okulturu.GetType();

Console.WriteLine(tip.IsArray);

tip = ad.GetType();

Console.WriteLine(tip.IsArray);

Console.ReadKey();

**Array\_SetValue**

Array okulturleri = Array.CreateInstance(typeof(string), 3);

okulturleri.SetValue("Endüstri Meslek", 0);

okulturleri.SetValue("Teknik", 1);

okulturleri.SetValue("Anadolu Teknik", 2);

Console.Write(okulturleri.GetValue(2));

Console.ReadKey();

**Array\_GetValue**

Array diller = Array.CreateInstance(typeof(string), 3, 3);

diller.SetValue("VB.Net", 0, 0);

diller.SetValue("C#", 0, 1);

diller.SetValue("C++", 0, 2);

diller.SetValue("Python", 1, 0);

diller.SetValue("Perl", 1, 1);

diller.SetValue("Ruby", 1, 2);

diller.SetValue("Asp.Net", 2, 0);

diller.SetValue("JavaScript", 2, 1);

diller.SetValue("Silverlight", 2, 2);

//Console.WriteLine(diller.GetValue(0, 1));

for (int i = 0; i < diller.GetLength(0); i++)

{

for (int j = 0; j < diller.GetLength(1); j++)

{

Console.Write(diller.GetValue(i, j) +" ");

if (j == diller.GetLength(1) - 1)

Console.WriteLine();

}

}

Console.ReadKey();

**Array\_Join**

string[] diller;

diller = new string[listBox1.Items.Count];

byte i;

for (i = 0; i <= listBox1.Items.Count - 1; i++)

{

diller[i] = listBox1.Items[i].ToString();

}

textBox1.Text = string.Join(",", diller);

**Array\_Copy**

string[] diller = new string[3];

diller[0] = "VB.Net";

diller[1] = "C#";

diller[2] = "ASP.Net";

string[] digerdiller = new string[3];

digerdiller[0] = "Python";

digerdiller[1] = "F#";

digerdiller[2] = "C++";

Array.Copy(digerdiller, diller, 3);

Console.WriteLine(diller[0]);

Console.WriteLine(diller[1]);

Console.WriteLine(diller[2]);

Console.ReadKey();

**Array\_Copy\_2**

string[] diller = new string[5];

diller[0] = "VB.Net";

diller[1] = "C#";

diller[2] = "ASP.Net";

string[] digerdiller = new string[3];

digerdiller[0] = "Python";

digerdiller[1] = "F#";

digerdiller[2] = "C++";

Array.Copy(digerdiller, 1, diller, 3, 2);

Console.WriteLine(diller[0]);

Console.WriteLine(diller[1]);

Console.WriteLine(diller[2]);

Console.WriteLine(diller[3]);

Console.WriteLine(diller[4]);

Console.ReadLine();

**Array\_Contains**

string aranan;

string[] isimler = new string[4];

isimler[0] = "Hasan";

isimler[1] = "Murat";

isimler[2] = "Talha";

isimler[2] = "Onur";

Console.Write("Aranan değeri giriniz:");

aranan = Console.ReadLine();

if (isimler.Contains(aranan) == false)

{

Console.Write("Aranan değer bulunamadı");

}

else

{

Console.Write("Aranan eleman bulundu");

}

Console.ReadKey();

**INDEXOF METODU İLE ARAMA YAPMA**

**Array\_IndexOf**

int indexno;

string aranan;

Array okulturu = Array.CreateInstance(typeof(string), 3);

okulturu.SetValue("Endüstri Meslek", 0);

okulturu.SetValue("Teknik", 1);

okulturu.SetValue("Anadolu Teknik", 2);

Console.Write("Aranan değeri giriniz:");

aranan = Console.ReadLine();

indexno = Array.IndexOf(okulturu, aranan.ToString());

if (indexno == -1)

{

Console.Write("Aranan değer bulunamadı");

}

else

{

Console.Write("Aranan eleman bulundu.İndex değeri = " +

indexno);

}

Console.ReadKey();

**Array\_LastIndexOf**

int indexno;

string aranan;

Array okulturu = Array.CreateInstance(typeof(string), 5);

okulturu.SetValue("Endüstri Meslek", 0);

okulturu.SetValue("Teknik", 1);

okulturu.SetValue("Teknik", 2);

okulturu.SetValue("Anadolu Teknik", 3);

okulturu.SetValue("Anadolu Teknik", 4);

Console.Write("Aranan değeri giriniz:");

aranan = Console.ReadLine();

indexno = Array.LastIndexOf(okulturu, aranan.ToString());

if (indexno == -1)

{

Console.Write("Aranan değer bulunamadı");

}

else

{

Console.Write("Aranan eleman bulundu.İndex değeri = " +

indexno);

}

Console.ReadKey();

**Array\_BinarySearch**

int indexno;

string aranan;

Array okulturu = Array.CreateInstance(typeof(string), 3);

//okulturu.SetValue("Endüstri Meslek", 0);

//okulturu.SetValue("Teknik", 1);

//okulturu.SetValue("Anadolu Teknik", 2);

okulturu.SetValue("Anadolu Teknik", 0);

okulturu.SetValue("Endüstri Meslek", 1);

okulturu.SetValue("Teknik", 2);

Console.Write("Aranan değeri giriniz:");

aranan = Console.ReadLine();

indexno = Array.BinarySearch(okulturu, aranan.ToString());

if (indexno < 0)

{

Console.Write("Aranan değer bulunamadı");

}

else

{

Console.Write("Aranan eleman bulundu.İndex değeri = " +

indexno);

}

Console.ReadKey();

**KOLLEKSİYONLAR**

**ArrayList\_Add**

ArrayList isimler = new ArrayList();

isimler.Add("Hasan");

isimler.Add("Murat");

isimler.Add("Talha");

isimler.Add("Onur");

Console.Write(isimler[2]);

Console.ReadKey();

**ArrayList\_Add\_FarkliDegiskenTurleri**

ArrayList bilgi = new ArrayList();

bilgi.Add("Talha");

bilgi.Add(123);

bilgi.Add("a");

Console.Write(bilgi[1]);

Console.ReadKey();

**ArrayList\_Insert**

ArrayList isimler = new ArrayList();

isimler.Add("Hasan");

isimler.Add("Murat");

isimler.Add("Talha");

isimler.Add("Onur");

isimler.Insert(0, "Sümeyye");

Console.WriteLine(isimler[0]);

Console.WriteLine(isimler[1]);

Console.ReadKey();

**ArrayList\_FixedSize**

ArrayList isimler = new ArrayList();

isimler.Add("Hasan");

isimler.Add("Murat");

isimler.Add("Talha");

isimler.Add("Onur");

ArrayList yeniisimler = ArrayList.FixedSize(isimler);

//yeniisimler.Insert(0, "Sümeyye");

yeniisimler[0] = "Sümeyye";

Console.WriteLine(yeniisimler[0]);

Console.WriteLine(yeniisimler[1]);

Console.ReadKey();

**ArrayList\_Sort**

ArrayList isimler = new ArrayList();

isimler.Add("Murat");

isimler.Add("Hasan");

isimler.Add("Talha");

isimler.Add("Onur");

isimler.Sort();

Console.WriteLine(isimler[0]);

Console.WriteLine(isimler[1]);

Console.WriteLine(isimler[2]);

Console.WriteLine(isimler[3]);

Console.ReadKey();

**ArrayList\_Sort\_Rakam**

ArrayList rakamlar = new ArrayList();

rakamlar.Add(8);

rakamlar.Add(5);

rakamlar.Add(3);

rakamlar.Add(9);

rakamlar.Add(7);

rakamlar.Sort();

Console.WriteLine(rakamlar[0]);

Console.WriteLine(rakamlar[1]);

Console.WriteLine(rakamlar[2]);

Console.WriteLine(rakamlar[3]);

Console.WriteLine(rakamlar[4]);

Console.ReadKey();

**ArrayList\_IComparer**

using System;

using System.Collections;

using System.Text;

namespace ArrayList\_IComparer

{

public class Compare : IComparer

{

int IComparer.Compare(Object x, Object y)

{

return ((new CaseInsensitiveComparer()).Compare(y, x));

}

}

class Program

{

static void Main(string[] args)

{

Compare CompareOptions = new Compare();

ArrayList rakamlar = new ArrayList();

rakamlar.Add(8);

rakamlar.Add(5);

rakamlar.Add(3);

rakamlar.Add(9);

rakamlar.Add(7);

rakamlar.Sort(CompareOptions);

Console.WriteLine(rakamlar[0]);

Console.WriteLine(rakamlar[1]);

Console.WriteLine(rakamlar[2]);

Console.WriteLine(rakamlar[3]);

Console.WriteLine(rakamlar[4]);

Console.ReadKey();

}

}

}

**ArrayList\_Count**

ArrayList rakamlar = new ArrayList();

rakamlar.Add(8);

rakamlar.Add(5);

rakamlar.Add(3);

rakamlar.Add(9);

rakamlar.Add(7);

rakamlar.Sort();

for (int i = 0; i <= rakamlar.Count - 1; i++)

{

Console.WriteLine(rakamlar[i]);

}

Console.ReadKey();

**ArrayList\_Remove**

ArrayList nickname = new ArrayList();

nickname.Add("Talha");

nickname.Add(123);

nickname.Add("a");

nickname.Remove(123);

Console.Write(nickname[1]);

Console.ReadKey();

**ArrayList\_RemoveAt**

ArrayList bilgi = new ArrayList();

bilgi.Add("Talha");

bilgi.Add(123);

bilgi.Add("a");

Console.WriteLine(bilgi[bilgi.Count - 1]);

bilgi.RemoveAt(bilgi.Count - 1);

Console.WriteLine(bilgi[bilgi.Count - 1]);

Console.ReadKey();

**Clear Metodu**

Diziadi.Clear();

**ArrayList\_Reverse**

ArrayList isimler = new ArrayList();

isimler.Add("Hasan");

isimler.Add("Murat");

isimler.Add("Talha");

isimler.Add("Onur");

isimler.Reverse();

Console.WriteLine(isimler[0]);

Console.WriteLine(isimler[1]);

Console.WriteLine(isimler[2]);

Console.WriteLine(isimler[3]);

Console.ReadKey();

**ArrayList\_Capacity**

ArrayList rakamlar = new ArrayList();

rakamlar.Add(1);

rakamlar.Add(2);

rakamlar.Add(3);

rakamlar.Add(4);

rakamlar.Add(5);

Console.WriteLine("Eleman Sayısı= " + rakamlar.Count);

Console.WriteLine("Kapasite= " + rakamlar.Capacity);

rakamlar.Remove(5);

rakamlar.Remove(4);

Console.WriteLine("Yeni Eleman Sayısı= " + rakamlar.Count);

Console.WriteLine("Yeni Kapasite= " + rakamlar.Capacity);

Console.ReadKey();

**ArrayList\_GetRange**

ArrayList diller = new ArrayList();

diller.Add("VB.Net");

diller.Add("C#");

diller.Add("Visual C++");

diller.Add("ASP.Net");

ArrayList digerdiller = diller.GetRange(0, 2);

foreach (string eleman in digerdiller)

Console.WriteLine(eleman);

Console.ReadKey();

**ArrayList\_AddRange**

ArrayList diller = new ArrayList();

diller.Add("VB.Net");

diller.Add("C#");

diller.Add("ASP.Net");

ArrayList digerdiller = new ArrayList();

digerdiller.Add("Python");

digerdiller.Add("F#");

digerdiller.Add("C++");

diller.AddRange(digerdiller);

foreach (string eleman in diller)

Console.WriteLine(eleman);

Console.ReadKey();

**ArrayList\_InsertRange**

ArrayList diller = new ArrayList();

diller.Add("VB.Net");

diller.Add("C#");

diller.Add("ASP.Net");

ArrayList digerdiller = new ArrayList();

digerdiller.Add("Python");

digerdiller.Add("F#");

digerdiller.Add("C++");

diller.InsertRange(2, digerdiller);

foreach (string eleman in diller)

Console.WriteLine(eleman);

Console.ReadKey();

**ArrayList\_RemoveRange**

ArrayList diller = new ArrayList();

diller.Add("VB.Net");

diller.Add("C#");

diller.Add("ASP.Net");

diller.RemoveRange(0, 2);

Console.Write(diller[0]);

Console.ReadKey();

**ArrayList\_CopyTo**

Array diller = Array.CreateInstance(typeof(string), 3);

diller.SetValue("VB.Net", 0);

diller.SetValue("C#", 1);

diller.SetValue("ASP.Net", 2);

ArrayList digerdiller = new ArrayList();

digerdiller.Add("Python");

digerdiller.Add("F#");

digerdiller.Add("C++");

digerdiller.CopyTo(diller);

Console.WriteLine(diller.GetValue(0));

Console.WriteLine(diller.GetValue(1));

Console.WriteLine(diller.GetValue(2));

Console.ReadKey();

**ArrayList\_CopyTo\_2**

Array diller = Array.CreateInstance(typeof(string), 4);

diller.SetValue("VB.Net", 0);

diller.SetValue("C#", 1);

diller.SetValue("ASP.Net", 2);

ArrayList digerdiller = new ArrayList();

digerdiller.Add("Python");

digerdiller.Add("F#");

digerdiller.Add("C++");

//digerdiller.CopyTo(diller, 1);

//digerdiller.CopyTo(0, diller, 0, 0);

//digerdiller.CopyTo(0, diller, 0, 2);

//digerdiller.CopyTo(0, diller, 1, 2);

digerdiller.CopyTo(1, diller, 0, 2);

foreach (string eleman in diller)

Console.WriteLine(eleman);

Console.ReadKey();

**ArrayList\_Clone**

ArrayList diller = new ArrayList();

diller.Add("VB.Net");

diller.Add("C#");

diller.Add("ASP.Net");

// ArrayList digerdiller = diller;

ArrayList digerdiller = new ArrayList();

digerdiller = (ArrayList)diller.Clone();

diller.Remove("ASP.Net");

Console.WriteLine(digerdiller[0]);

Console.WriteLine(digerdiller[1]);

Console.WriteLine(digerdiller[2]);

Console.ReadKey();

**ArrayList\_ToArray**

ArrayList diller = new ArrayList();

diller.Add("VB.Net");

diller.Add("C#");

diller.Add("ASP.Net");

string[] yenidizi = (string[])diller.ToArray(typeof(string));

foreach (string eleman in yenidizi)

Console.WriteLine(eleman);

Console.ReadKey();

**HashTable\_Add**

Hashtable sınıflar = new Hashtable();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

//Anahtarlar

ICollection anahtarlar = sınıflar.Keys;

foreach (string eleman in anahtarlar)

Console.WriteLine(eleman);

//Değerler

//ICollection degerler = sınıflar.Values;

//foreach (string eleman in degerler)

//Console.WriteLine(eleman);

Console.ReadLine();

**Hashtable\_Remove**

Hashtable sınıflar = new Hashtable();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

foreach (DictionaryEntry eleman in sınıflar)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

sınıflar.Remove("A10B");

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

foreach (DictionaryEntry eleman in sınıflar)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

Console.ReadLine();

Hashtable\_varsayilan\_anahtar

Hashtable isimler = new Hashtable();

isimler[0] = ("Volkan AKTAŞ");

isimler[1] = ("İrfan MERGAN");

isimler[2] = ("Bünyamin KARAHAN");

isimler[3] = ("Hale KATMER");

isimler[4] = ("Tamer MATBAN");

foreach (DictionaryEntry eleman in isimler)

listBox1.Items.Add(eleman.Key + " = " + eleman.Value);

**HashTable\_Contain**

string aranan;

Hashtable sınıflar = new Hashtable();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

Console.Write("Aranan anahtarı giriniz:");

aranan = Console.ReadLine();

//if (sınıflar.ContainsValue(aranan) == false)

if (sınıflar.ContainsKey(aranan) == false)

{

Console.Write("Aranan anahtar bulunamadı");

}

else

{

Console.Write("Aranan anahtar bulundu");

}

Console.ReadKey();

**SortedList\_IndexOf**

SortedList sınıflar = new SortedList();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

//Console.WriteLine(sınıflar.IndexOfKey("T10A"));

//Console.WriteLine(sınıflar.IndexOfValue("Teknik Lise"));

//Console.WriteLine(sınıflar.GetKey(4));

Console.WriteLine(sınıflar.GetByIndex(4));

Console.ReadLine();

**Sorted\_List**

int i;

SortedList sınıflar = new SortedList();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

//sınıflar["E10A"] = "Teknik Lise";

//foreach (DictionaryEntry eleman in sınıflar)

//listBox1.Items.Add(eleman.Key + " = " + eleman.Value);

for (i=0; i < sınıflar.Count; i++)

listBox1.Items.Add(sınıflar.GetByIndex(i));

**SortedList\_IComparer**

SortedList sınıflar = new SortedList(new Compare());

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

foreach (DictionaryEntry eleman in sınıflar)

listBox1.Items.Add(eleman.Key + " = " + eleman.Value);

**SortedList\_GetEnumerator**

SortedList sınıflar = new SortedList();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

IDictionaryEnumerator eleman = sınıflar.GetEnumerator();

while (eleman.MoveNext())

{

listBox1.Items.Add(eleman.Key.ToString() + " = " + eleman.Value.ToString());

}

**SortedList\_TryGetValue**

SortedList<string, string> sınıflar = new SortedList<string, string>();

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

string deger;

if (sınıflar.TryGetValue("T10A", out deger))

{

Console.WriteLine("T10A anahtarının değeri = " + deger);

}

Console.ReadLine();

**SortedList\_IComparer**

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SortedList\_IComparer

{

public partial class Form1 : Form

{

public class Compare : IComparer

{

int IComparer.Compare(Object x, Object y)

{

return ((new CaseInsensitiveComparer()).Compare(y, x));

}

}

public Form1()

{

InitializeComponent();

}

private void btnEkle\_Click(object sender, EventArgs e)

{

SortedList sınıflar = new SortedList(new Compare());

sınıflar.Add("E10A", "Endüstri Meslek");

sınıflar.Add("E10B", "Endüstri Meslek");

sınıflar.Add("T10A", "Teknik Lise");

sınıflar.Add("A10A", "Anadolu Teknik");

sınıflar.Add("A10B", "Anadolu Teknik");

foreach (DictionaryEntry eleman in sınıflar)

listBox1.Items.Add(eleman.Key + " = " + eleman.Value);

}

}

}

**GENERİC COLLECTİONS**

**LİST SINIFI**

**List\_Nesnesi**

List<string> listemiz = new List<string>();

listemiz.Add("VB.NET");

listemiz.Add("C#");

listemiz.Add("ASP.NET");

Console.WriteLine(listemiz[listemiz.Count - 1]);

listemiz.RemoveAt(listemiz.Count - 1);

Console.WriteLine(listemiz[listemiz.Count - 1]);

Console.ReadLine();

**list\_To\_ArrayList**

List<string> listemiz = new List<string>();

listemiz.Add("VB.NET");

listemiz.Add("C#");

listemiz.Add("ASP.NET");

ArrayList yenidizi = new ArrayList(listemiz);

foreach (string eleman in yenidizi)

Console.WriteLine(eleman);

Console.ReadKey();

**STACK (YIĞIN) SINIFI**

**stack\_push**

Stack<string> isimler = new Stack<string>();

isimler.Push("Talha");

isimler.Push("Sümeyye");

isimler.Push("Azra");

isimler.Push("Murat");

foreach (string i in isimler)

{

Console.WriteLine(i);

}

Console.ReadLine();

**stack\_ElementAt**

int i;

Stack<string> isimler = new Stack<string>();

isimler.Push("Talha");

isimler.Push("Sümeyye");

isimler.Push("Azra");

isimler.Push("Murat");

for (i = 0; i<isimler.Count ; i++)

Console.WriteLine(isimler.ElementAt(i));

Console.ReadLine();

**stack\_pop**

Stack<string> isimler = new Stack<string>();

isimler.Push("Talha");

isimler.Push("Sümeyye");

isimler.Push("Azra");

isimler.Push("Murat");

Console.WriteLine("En üstte yer alan eleman= " +isimler.Peek());

isimler.Pop();

Console.WriteLine("En üstte yer alan eleman= " + isimler.Peek());

Console.ReadLine();

**QUEUE (KUYRUK) SINIFI**

**Queue\_Sınıfı**

Queue<string> isimler = new Queue<string>();

isimler.Enqueue("Talha");

isimler.Enqueue("Sümeyye");

isimler.Enqueue("Azra");

isimler.Enqueue("Murat");

foreach(string i in isimler)

Console.WriteLine(i);

Console.ReadLine();

**Queue\_Enqueue**

Queue<string> isimler = new Queue<string>();

isimler.Enqueue("Talha");

isimler.Enqueue("Sümeyye");

isimler.Enqueue("Azra");

isimler.Enqueue("Murat");

Console.WriteLine("En üstte yer alan eleman= " +isimler.Peek());

isimler.Dequeue();

Console.WriteLine("En üstte yer alan eleman= " +isimler.Peek());

Console.ReadLine();

**LİNKEDLİST (BAĞLI LİSTE) SINIFI**

**LinkedList\_Add**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

foreach (var eleman in liste)

{

Console.WriteLine(eleman);

}

Console.ReadLine();

**LinkedList\_First\_Last**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

Console.WriteLine("Listenin ilk elemanı= " + liste.First.Value);

Console.WriteLine("Listenin son elemanı= " + liste.Last.Value);

Console.ReadLine();

**LinkedList\_Remove**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

liste.Remove("Değer 3");

foreach (var eleman in liste)

{

Console.WriteLine(eleman);

}

Console.ReadLine();

**LinkedList\_Remove\_First\_Last**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

Console.WriteLine("Listenin ilk elemanı= " +liste.First.Value);

Console.WriteLine("Listenin son elemanı= " + liste.Last.Value);

liste.RemoveFirst();

liste.RemoveLast();

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

Console.WriteLine("Listenin ilk elemanı= " + liste.First.Value);

Console.WriteLine("Listenin son elemanı= " + liste.Last.Value);

Console.ReadLine();

**LinkedList\_Add\_Find**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

LinkedListNode<string> bulunan = liste.Find("Değer 3");

Console.WriteLine(" Bulunan değer= " +bulunan.Value.ToString());

Console.WriteLine(" Bulunan değerin sonrası= " + bulunan.Next.Value.ToString());

Console.WriteLine(" Bulunan değerin öncesi= " + bulunan.Previous.Value.ToString());

Console.ReadLine();

**LinkedList\_Add\_Before\_After**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

//liste.AddBefore(liste.Find("Değer 3"), "Eklenen Değer");

liste.AddAfter(liste.Find("Değer 3"), "Eklenen Değer");

foreach (var eleman in liste)

{

Console.WriteLine(eleman);

}

Console.ReadLine();

**LinkedList\_Previous\_Next**

LinkedList<string> liste = new LinkedList<string>();

liste.AddLast("Değer 2");

liste.AddLast("Değer 3");

liste.AddLast("Değer 4");

liste.AddFirst("Değer 1");

//Baştan sonra doğru listeleme

LinkedListNode<string> eleman = liste.First;

while (eleman != null)

{

Console.WriteLine(eleman.Value);

eleman = eleman.Next;

}

//sondan başa doğru listeleme

//LinkedListNode<string> eleman = liste.Last;

//while (eleman != null)

//{

//Console.WriteLine(eleman.Value);

//eleman = eleman.Previous;

//}

Console.ReadLine();

**DİCTİONARY (SÖZLÜK) SINIFI**

**Dictionary\_Add**

Dictionary<string, string> dallar = new Dictionary<string, string>();

dallar.Add("Volkan AKTAŞ", "Programlama");

dallar.Add("Bünyamin KARAHAN", "Programlama");

dallar.Add("İrfan MERGAN", "Donanım");

dallar.Add("Hale KATMER", "Web Tasarım");

//Console.WriteLine(dallar["Bünyamin KARAHAN"]);

foreach (var eleman in dallar)Console.WriteLine(eleman);

Console.ReadLine();

**Dictionary\_key\_value**

Dictionary<string, string> dallar = new Dictionary<string, string>();

dallar.Add("Volkan AKTAŞ", "Programlama");

dallar.Add("Bünyamin KARAHAN", "Programlama");

dallar.Add("İrfan MERGAN", "Donanım");

dallar.Add("Hale KATMER", "Web Tasarım");

foreach (var eleman in dallar)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

Console.ReadLine();

**Dictionary\_KeyValuePair**

Dictionary<string, string> dallar = new Dictionary<string, string>();

dallar.Add("Volkan AKTAŞ", "Programlama");

dallar.Add("Bünyamin KARAHAN", "Programlama");

dallar.Add("İrfan MERGAN", "Donanım");

dallar.Add("Hale KATMER", "Web Tasarım");

foreach (KeyValuePair<string, string> eleman in dallar)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

Console.ReadLine();

**Dictionary\_Remove**

Dictionary<string, string> dallar = new Dictionary<string, string>()

{

{"Volkan AKTAŞ", "Programlama"},

{"Bünyamin KARAHAN", "Programlama"},

{"İrfan MERGAN", "Donanım"},

{"Hale KATMER", "Web Tasarım"}

};

dallar.Remove("İrfan MERGAN");

foreach (var eleman in dallar)

Console.WriteLine(eleman);

Console.ReadLine();

**Dictionary\_ToList**

Dictionary<string, string> dallar = new Dictionary<string, string>()

{

{"Volkan AKTAŞ", "Programlama"},

{"Bünyamin KARAHAN", "Programlama"},

{"İrfan MERGAN", "Donanım"},

{"Hale KATMER", "Web Tasarım"}

};

foreach (var eleman in dallar.Keys.ToList())

dallar[eleman] = "Programlama";

foreach (KeyValuePair<string, string> eleman in dallar)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

Console.ReadLine();

**Dictionary\_Sort**

Dictionary<string, int> bilgi = new Dictionary<string, int>()

{

{"Talha AKTAŞ", 10},

{"Sümeyye AKTAŞ", 7},

{"Murat AKTAŞ" , 17},

{"Azra UNCU", 4}

};

List<string> lst\_bilgi = new List<string>(bilgi.Keys);

lst\_bilgi.Sort();

foreach (string anahtar in lst\_bilgi)

Console.WriteLine(anahtar);

Console.ReadLine();

**Dictionary\_Linq**

Dictionary<string, int> bilgi = new Dictionary<string, int>()

{

{"Talha AKTAŞ", 10},

{"Sümeyye AKTAŞ", 7},

{"Murat AKTAŞ" , 17},

{"Mehmet GÖKMEN", 19}

};

//var sırala = from eleman in bilgi orderby eleman.Key descending select eleman.Key;

var arama = bilgi.Where(eleman => eleman.Key.StartsWith("M"))

.Select(eleman => eleman.Key);

foreach (string anahtar in arama)

Console.WriteLine(anahtar);

Console.ReadLine();

**SORTEDDICTIONARY SINIFI**

**SortedDictionary**

SortedDictionary<string, int> bilgi = new SortedDictionary<string, int>()

{

{"Talha AKTAŞ", 10},

{"Sümeyye AKTAŞ", 7},

{"Murat AKTAŞ" , 17},

{"Azra UNCU", 4}

};

foreach (var eleman in bilgi)

Console.WriteLine(eleman);

Console.ReadLine();

**SORTEDSET SINIFI**

**SortedSet\_AddRemove**

SortedSet<string> okul = new SortedSet<string>();

okul.Add("Teknik");

okul.Add("Anadolu Teknik");

okul.Add("Endüstri Meslek");

okul.Add("Anadolu Meslek");

foreach (string degerler in okul)

Console.WriteLine(degerler);

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

okul.Remove("Anadolu Meslek");

foreach (string degerler in okul)

Console.WriteLine(degerler);

Console.ReadLine();

**SortedSet\_TekrarDeger**

SortedSet<string> okul = new SortedSet<string>();

okul.Add("Teknik");

okul.Add("Anadolu Teknik");

okul.Add("Endüstri Meslek");

//okul.Add("Anadolu Meslek");

//bool donen\_deger=okul.Add("Teknik");

foreach (string degerler in okul)

Console.WriteLine(degerler);

//Console.WriteLine(donen\_deger);

Console.ReadLine();

**SortedSet\_RemoveWhere**

SortedSet<string> okul = new SortedSet<string>();

okul.Add("Teknik");

okul.Add("Anadolu Teknik");

okul.Add("Endüstri Meslek");

okul.Add("Anadolu Meslek");

foreach (string degerler in okul)

Console.WriteLine(degerler);

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

okul.RemoveWhere(deger=>deger.StartsWith("A"));

foreach (string degerler in okul)

Console.WriteLine(degerler);

Console.ReadLine();

**List\_To\_SortedSet**

List<string> list = new List<string>();

list.Add("Teknik");

list.Add("Anadolu Teknik");

list.Add("Endüstri Meslek");

list.Add("Anadolu Meslek");

list.Add("Teknik");

SortedSet<string> s\_set = new SortedSet<string>(list);

foreach (string degerler in s\_set)

Console.WriteLine(degerler);

Console.ReadLine();

**SortedSet\_RemoveWhere**

SortedSet<string> okul = new SortedSet<string>();

okul.Add("Teknik");

okul.Add("Anadolu Teknik");

okul.Add("Endüstri Meslek");

okul.Add("Anadolu Meslek");

foreach (string degerler in okul)

Console.WriteLine(degerler);

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

okul.RemoveWhere(deger => deger.StartsWith("A"));

foreach (string degerler in okul)

Console.WriteLine(degerler);

Console.ReadLine();

**SortedList Türetme**

List<string> list = new List<string>();

list.Add("Teknik");

list.Add("Anadolu Teknik");

list.Add("Endüstri Meslek");

list.Add("Anadolu Meslek");

list.Add("Teknik");

SortedSet<string> s\_set = new SortedSet<string>(list);

foeach(string degerler in s\_set)

Console.WriteLine(degerler);

Console.ReadLine();

**HASHSET SINIFI**

**HashSet\_AddRemove**

HashSet<string> okul = new HashSet<string>();

okul.Add("Teknik");

okul.Add("Anadolu Teknik");

okul.Add("Endüstri Meslek");

okul.Add("Anadolu Meslek");

foreach (string degerler in okul)

Console.WriteLine(degerler);

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

okul.Remove("Anadolu Meslek");

foreach (string degerler in okul)

Console.WriteLine(degerler);

Console.ReadLine();

**HashSet\_SetEquals**

List<string> list = new List<string>();

list.Add("C#");

list.Add("VB.Net");

list.Add("ASP.Net");

list.Add("MVC");

list.Add("C#");

HashSet<string> h\_set = new HashSet<string>();

h\_set.Add("VB.Net");

h\_set.Add("C#");

h\_set.Add("ASP.Net");

h\_set.Add("MVC");

if (h\_set.SetEquals(list))

Console.WriteLine("Hashset, List'e eşit");

else

Console.WriteLine("Hashset, List'e eşit değil");

Console.ReadLine();

**HashSet\_InterSectWith**

List<string> list = new List<string>();

list.Add("C#");

list.Add("VB.Net");

list.Add("ASP.Net");

list.Add("MVC");

list.Add("Silverlight");

HashSet<string> h\_set = new HashSet<string>();

h\_set.Add("VB.Net");

h\_set.Add("C#");

h\_set.Add("WPF");

h\_set.IntersectWith(list);

foreach (string degerler in h\_set)

Console.WriteLine(degerler);

Console.ReadLine();

**SPECİALİZED (ÖZELLEŞTİRİLMİŞ) KOLEKSİYONLAR**

**Listed Dictionary**

using System;

using System.Collections;

using System.Collections.Specialized;

using System.Text;

namespace ListDictionary\_AddRemove

{

class Program

{

static void Main(string[] args)

{

ListDictionary isimler = new ListDictionary();

isimler.Add("Volkan AKTAŞ", 1);

isimler.Add("Bünyamin KARAHAN", 2);

isimler.Add("İrfan MERGAN", 3);

isimler.Add("İsmail Hakkı GÜVEN", 4);

foreach (DictionaryEntry eleman in isimler)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

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

isimler.Remove("Volkan AKTAŞ");

foreach (DictionaryEntry eleman in isimler)

Console.WriteLine(eleman.Key + " = " + eleman.Value);

Console.ReadLine();

}

}

}

**ListDictionary & Hash Table**

using System;

using System.Collections;

using System.Collections.Specialized;

using System.Text;

namespace ListDictionary\_Performans

{

class Program

{

static void Main(string[] args)

{

DateTime baslangiczamani,bitiszamani;

TimeSpan fark;

ListDictionary isimler\_list = new ListDictionary();

isimler\_list.Add("Volkan AKTAŞ", 1);

isimler\_list.Add("Bünyamin KARAHAN", 2);

isimler\_list.Add("İrfan MERGAN", 3);

isimler\_list.Add("İsmail Hakkı GÜVEN", 4);

isimler\_list.Add("Murat UNCU", 5);

Hashtable isimler\_hash = new Hashtable();

isimler\_hash.Add("Volkan AKTAŞ", 1);

isimler\_hash.Add("Bünyamin KARAHAN", 2);

isimler\_hash.Add("İrfan MERGAN", 3);

isimler\_hash.Add("İsmail Hakkı GÜVEN", 4);

isimler\_hash.Add("Murat UNCU", 5);

baslangiczamani = DateTime.Now;

for (int i = 0; i < 1000000; i++)

{

object a = isimler\_list["Volkan AKTAŞ"];

}

bitiszamani = DateTime.Now;

fark = bitiszamani-baslangiczamani;

Console.WriteLine("ListDictionary işlem zamanı (ms) = " + fark.Milliseconds);

baslangiczamani = DateTime.Now;

for (int i = 0; i < 1000000; i++)

{

object a = isimler\_hash["Volkan AKTAŞ"];

}

bitiszamani = DateTime.Now;

fark = bitiszamani - baslangiczamani;

Console.WriteLine("Hashtable işlem zamanı (ms) = " + fark.Milliseconds);

Console.ReadLine();

}

}

}

**HYBRIDDICTIONARY SINIFI**

**HybridDictionary\_Performans**

using System;

using System.Collections;

using System.Collections.Specialized;

using System.Text;

namespace HybridDictionary\_Performans

{

class Program

{

static void Main(string[] args)

{

DateTime baslangiczamani, bitiszamani;

TimeSpan fark;

int[] dizi = new int[1000000];

Random rs = new Random();

for (int i = 0; i < 999999; i++)

dizi[i]=rs.Next(0,100000);

HybridDictionary hl = new HybridDictionary();

baslangiczamani = DateTime.Now;

for (int i = 0; i <= 999999; i++)

hl.Add(i, dizi[i]);

bitiszamani = DateTime.Now;

fark = bitiszamani - baslangiczamani;

Console.WriteLine("HybridDictionary işlem zamanı (ms) = " + fark.Milliseconds);

Hashtable ht = new Hashtable();

baslangiczamani = DateTime.Now;

for (int i = 0; i < 999999; i++)

ht.Add(i, dizi[i]);

bitiszamani = DateTime.Now;

fark = bitiszamani - baslangiczamani;

Console.WriteLine("Hashtable işlem zamanı (ms) = " + fark.Milliseconds);

Console.ReadLine();

}

}

}

**HybridDictionary\_StringPerformans**

using System;

using System.Collections;

using System.Collections.Generic;

using System.Collections.Specialized;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace HybridDictionary\_StringPerformans

{

class Program

{

static void Main(string[] args)

{

string[] strChars = {"A","B","C","D","E","F","G","H","I","İ","J","K","L","M","N","O","Ö","P","R","S","Ş","T","U","Ü","V","Y","Z"};

DateTime baslangiczamani, bitiszamani;

TimeSpan fark;

string[] dizi = new string[1000000];

Random rnd = new Random();

int mb = strChars.GetUpperBound(0);

for (int i = 0; i <999999; i++)

dizi[i]=strChars[rnd.Next(mb)];

HybridDictionary hl = new HybridDictionary();

baslangiczamani = DateTime.Now;

for (int i = 0; i <= 999999; i++)

hl.Add(i, dizi[i]);

bitiszamani = DateTime.Now;

fark = bitiszamani - baslangiczamani;

Console.WriteLine("HybridDictionary işlem zamanı (ms) = " + fark.Milliseconds);

Hashtable ht = new Hashtable();

baslangiczamani = DateTime.Now;

for (int i = 0; i < 999999; i++)

ht.Add(i, dizi[i]);

bitiszamani = DateTime.Now;

fark = bitiszamani - baslangiczamani;

Console.WriteLine("Hashtable işlem zamanı (ms) = " + fark.Milliseconds);

Console.ReadLine();

}

}

}

**ORDERED DİCTİONARY SINIFI**

**Ordered\_Dictionary**

using System;

using System.Collections.Specialized;

using System.Text;

namespace Ordered\_Dictionary

{

class Program

{

static void Main(string[] args)

{

OrderedDictionary od = new OrderedDictionary();

od.Add("T10A","Teknik");

od.Add("A10A","Anadolu Teknik");

od.Add("E10A","Endüstri Meslek");

od.Insert(0,"M10A","Anadolu Meslek");

string[] anahtarlar = new string[od.Keys.Count];

od.Keys.CopyTo(anahtarlar, 0);

for (int i = 0; i < od.Keys.Count; i++)

{

Console.WriteLine(

"İndex = {0}, Anahtar = {1}, Değer = {2}",

i,anahtarlar[i],od[i]);

}

Console.ReadLine();

}

}

}

**OrderedDictionary\_Remove\_RemoveAt**

using System;

using System.Collections;

using System.Collections.Specialized;

using System.Text;

namespace OrderedDictionary\_Remove\_RemoveAt

{

class Program

{

static void Main(string[] args)

{

OrderedDictionary od = new OrderedDictionary();

od.Add("T10A", "Teknik");

od.Add("A10A", "Anadolu Teknik");

od.Add("E10A", "Endüstri Meslek");

od.Insert(0, "M10A", "Anadolu Meslek");

od.Remove("E10A");

od.RemoveAt(0);

IDictionaryEnumerator eleman = od.GetEnumerator();

while (eleman.MoveNext())

{

Console.WriteLine("Anahtar = {0}, Değer = {1}",

eleman.Key,eleman.Value);

}

Console.ReadLine();

}

}

}

**STRING COLLECTİON SINIFI**

**StringCollection\_Add**

using System;

using System.Collections.Specialized;

using System.Text;

namespace StringCollection\_Add

{

class Program

{

static void Main(string[] args)

{

StringCollection diller = new StringCollection();

diller.Add("F#");

diller.Add("VB.Net");

diller.Insert(0,"C#");

string[] scriptdiller = new string[]{"Perl", "Ruby", "Python"};

diller.AddRange(scriptdiller);

foreach (string isim in diller)

Console.WriteLine(isim);

Console.ReadLine();

}

}

}

**StringCollection\_textBox**

using System;

using System.Collections.Specialized;

using System.Windows.Forms;

namespace StringCollection\_textBox

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void btnAktar\_Click(object sender, EventArgs e)

{

StringCollection sc = new StringCollection();

foreach (string satırlar in textBox1.Lines)

{

sc.Add(satırlar);

}

MessageBox.Show(sc[0]);

}

}

}