No:190160104002

Adı: Berkant

Soyadı: EVREN

Görsel Programlama 2   
1.Vize Ödevi

2.Hafta: Sınıf, nesne, kapsülleme, kalıtım kavramlarının nasıl işlediğini gördük.

Kodlar:

Form: using System;

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 WindowsFormsApplication6

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

araba Fiat = new araba();//yeni bir nesne üretiyoruz.

Fiat.yakit = "dizel";//nesnenin değişkenlerini atıyoruz.

Fiat.renk = "kırmızı";

Fiat.fiyat = 45000;

Fiat.MARKASI = "LineA";

Fiat.model = "turbo";

label1.Text = Fiat.yakit;//değişkenleri labela yazdırıyoruz.

label2.Text = Fiat.renk;

label3.Text = Fiat.fiyat.ToString();

label4.Text = Fiat.MARKASI;

label5.Text = Fiat.model;

}

}

}

Class1: using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WindowsFormsApplication6

{

class araba:araba1 //araba isminde bir sınıf oluşturuyor. Yanındaki araba1 ise başka bir sınıfı buraya atıyor.

{

public string yakit; //başka yerlerden görünmesi için public bir değişken oluşturuyor.

public string renk;

public int fiyat;

private string marka; //başka yerlerde görünmeyen sadece bu sınıfa özel bir değişken oluşturuyor.

public string MARKASI

{

get { return marka; } //marka değişkeni başka yerde görünmediğini için burada döndürüyor.

set { marka = value.ToLower(); } //bütün harlerin küçük yazılmasını sağlıyor.

}

}

}

Class2: using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WindowsFormsApplication6

{

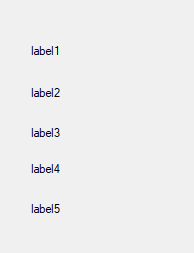
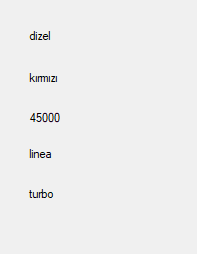
class araba1

{

public string model;

}

}

Form Design:Çalışınca:

Açıklama: Her zaman ki gibi bir form sayfası açıyoruz. Formun içinde değil de bu sefer bir class açıp içine değişkenleri atıyoruz böylelikle o değişkenlerden istediğimiz kadar nesne üretme şansımız oluyor. Atadığımız değişkenleri form sayfası üzerinde çağırıp düzenliyoruz. Kapsülleme işleminde ise private olarak bir değişken alıp public bir değişken içinde onu döndürüyoruz. Böylelikle içinde değişiklikler yapabiliyoruz. Kullanıcı bunu görmüyor. Kalıtımda ise başka bir class açıp onu eski classımızın üstüne gömüyoruz bu sayede yeni clasımızı eski classımızın içinde gibi gözüküyor istediğimiz zaman çağırabiliyoruz. Hazır classımız yanda duruyor kullanmak istediğimizde çağırıyoruz.

3.Hafta: Console write, writeLine , read komutlarını ve sınıf içi metodları nasıl olduğunu gördük.

Kodlar:

program.cs: using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

haber hbr = new haber();

hbr.habermetni();

hbr.kacdefa();

hbr.karesinial(10); //geriye değer döndüren metoddur.

Console.Read();

}

}

}

haber.cs: using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication1

{

class haber

{

public haber()

{

Console.WriteLine("Hoşgeldiniz"); //yazıyı yazar imleci aşağıya indirir.

}

public void habermetni() //geriye değer döndürmeyen metoddur.

{

Console.WriteLine("Merhabalar");

}

public void kacdefa()

{

for(int i=1; i<=3; i++)

{

Console.WriteLine("Nasılsınız" + i);

}

}

public int karesinial (int x) //geriye değer döndüren metoddur.

{

int sayininkaresi = x \* x;

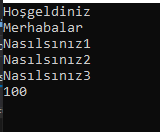
Console.WriteLine(sayininkaresi);

return sayininkaresi;

}

}

}

Çalışınca: 

Açıklama: Bu hafta console üzerinden kodlarla ilerleyeceğimiz için console application açıyoruz. Direk program.cs dosyası önümüze geliyor. Burada console.read’ın program çalışınca gelen programın kalmasını sağladığını, console.write’ın içine yazılanların programa yazıldığını ve console.writeline’ın da aynı işlevi görüp sadece imleci bir alt satıra inmesini yarayan kod olduğunu öğrendik. Sonrasında daha önce aynı form sayfasında içinde çağırdığımız metodu bu sefer bir class dosyası üzerinden çağırma işleme yaptık. Geriye döndürmeyen metodta void’in kullanıldığını, geriye değer döndüren metoda da ise int ve string gibi değişkenlerin kullanıldığını, bu geriye değer döndüren metodların içindekileri program içinde yapıldığını ve son olarak da yapıcı metodu öğrendim. Bu yapıcı metodta ise sınıf ile aynı isimde bir metod kullandığımızda sınıfı çağırdığımızda direk olarak programa geldiğini öğrendim.

4.Hafta: MassageBox’ın parametrelerini, açılır menüyü ve farede sağ tıkladığımızda açılan menüyü gördük.

Kodlar: using System;

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 WindowsFormsApplication8

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

MessageBox.Show("Merhaba", "Mesajınız",MessageBoxButtons.YesNoCancel,MessageBoxIcon.Error);

button1.Text = "değiştirildi"; //messagebox’ı özelleştiriyoruz.

}

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

{

this.BackColor = Color.Red; //açılır menüde tuşa basınca arka fon rengi değiştirmek.

}

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

{

this.BackColor = Color.Yellow;

}

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

{

Application.Exit();//programı kapatıyor.

}

private void gT250RToolStripMenuItem\_Click(object sender, EventArgs e)

{

label1.Text = "En Güzel Motor Ucuz Olarak :D";

webBrowser1.Navigate("https://www.youtube.com/watch?v=-GTUULb\_VKQ");//web browser da istediğimiz linki açıyor.

}

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

{

Application.Restart();//programı yeniden başlatıyor.

}

private void hesapMakinesiToolStripMenuItem1\_Click(object sender, EventArgs e)

{

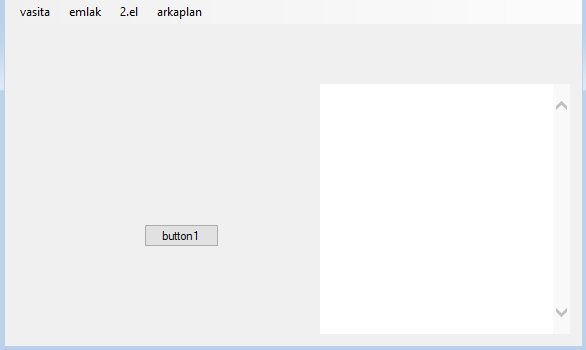
System.Diagnostics.Process.Start("cals.exe");//sistemde bulunan bir uygulamayı açıyor.

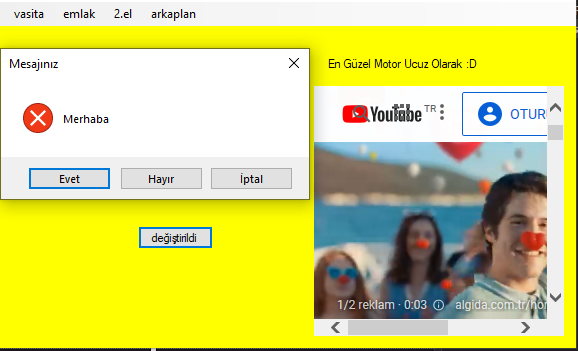
}

}

}

Açıklama: MessageBox.Show kodunda en son sadece içine yazı yazmayı görmüştük. Buna ek olarak massagebox’ın içinde ilk tırnakların içine yazdığımız içeriği yazdıktan sonra virgül koyup yeniden tırnak açtığımızda bu sefer massagebox’ın başlığını yazıldığını, bir virgül daha attığımızda bu sefer de messagebox’ın butonunu değiştirmeye ve bir virgül daha attıktan sonra da messagebox’ın içindeki butonlarını değiştirebileceğimizi öğrendik. Toolbox kısmından MenuStrip eklediğimizde yukarda açılır menü yapabiliyoruz. İstediğimiz alt menü yapabiliyor ve içlerini doldurabiliyoruz. Doldurduğumuzda oluşan yerlere çift tıklayıp oraya bastığımızda ne olmasını istediğimizi belirtebiliyoruz. Aynı şekilde ContextMenuStrip ekleyip program çalışırken sağ tıka bastığımızda neler olması isteğimize göre değiştirmeyi öğrendim.

Form Design: 

Çalışınca: 

5.Hafta: 2 form arası işlem yapmayı gördük.

Kodlar:

Form1: using System;

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 WindowsFormsApplication10

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Form2 f2yegit = new Form2(); //form2’ye gitmek için bir nesne türetiyor.

f2yegit.isim = textBox1.Text; //textboxdaki veriyi isim değişkenine atıyor

f2yegit.Show(); //form2’yi açmasını söylüyor.

this.Hide(); //olduğumuz formu gizleme işlemi yapıyor.

}

}

}

Form2: using System;

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 WindowsFormsApplication10

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

public string isim;

private void button1\_Click(object sender, EventArgs e)

{

Form1 f1egit = new Form1();

f1egit.Show();

this.Close(); //olduğumuz formu kapatıyor.

}

private void Form2\_Load(object sender, EventArgs e)

{

label1.Text = isim;

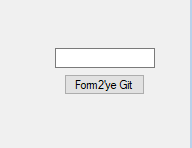
}

}

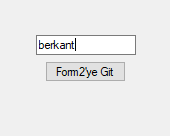
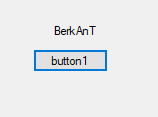
}

Açıklama: İlk başta 2 form arası geçişi ve diğer formun görünmez olduğunu daha sonra ise bir değişken yardımı ile bir şeyi diğer form da başka bir yere yazdırmayı öğrendim.

Form Design:

Form1:  Form2: 

Çalışınca:

Form1:  Form2: 

6.Hafta: Lokasyon değiştirmeyi gördük.

Kodlar:

using System;

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 WindowsFormsApplication10

{

public partial class Form3 : Form

{

public Form3()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Point konum = new Point(); //yeni bir nesne oluşturuyor.

konum.X = 150; //point kullandığımız için x ve y ler çıkıyor. Bunlara değer veriyoruz.

konum.Y = 150;

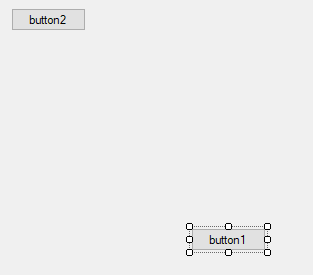
button2.Location = konum; //butonumuzun lokasyonunu bu nesneye göre değiştirmesini söylüyoruz.

}

}

}

Açıklama: Butona bastığımız zaman diğer butonun yerini yazdığımız lokasyona nasıl geçtiğini öğrendim.

Form Design: 

Çalışınca: 