namespace BirKelimeBirIslem.UI.Win.Classes.Islem.Enums

{

public class Enum

{

public enum Islem\_Isaret

{

//Dörtişlem bilgisini saklar

Ekle,

Cikar,

Carp,

Bol

}

}

}

using System;

using System.Collections.Generic;

using System.Text;

namespace BirKelimeBirIslem.UI.Win.Classes

{

public class OyunAyarlari

{

#region Properties

Random rnd { get; set; }

public int IkiBasamakli { get; set; }

public List<int> TekBasamakli { get; private set; }

public int HedefSayi { get; set; }

Cozum Cozum { get; set; }

#endregion Properties

#region Constructor

public OyunAyarlari()

{

//İşlem oyunu için gerekli veriler constructor metodunda dolduruluyor

rnd = new Random();

if (TekBasamakli != null) TekBasamakli.Clear();

IkiBasamakli = IkiBasamakliOlustur();

TekBasamakli = TekBasamakliOlustur();

HedefSayi = HedefSayiOlustur();

}

#endregion

#region Functions

public Cozum Basla()

{

//Çözüm class ı initialize edilerek arama işlemi tetikleniyor

this.Cozum = new Cozum(HedefSayi, TekBasamakli, IkiBasamakli);

return Cozum;

}

#endregion

#region SayılarıOlustur

// Random olarak sayılar oluşturuluyor

private int IkiBasamakliOlustur()

{

List<int> sayilar = new List<int>() { 10, 20, 30, 40, 50, 60, 70, 80, 90 };

return sayilar[rnd.Next(0, sayilar.Count)];

}

private List<int> TekBasamakliOlustur()

{

List<int> sayilar = new List<int>();

for (int i = 0; i < 5; i++)

{

sayilar.Add(rnd.Next(1, 9));

}

return sayilar;

}

private int HedefSayiOlustur()

{

return rnd.Next(100, 1000);

}

#endregion PlayGame

#region ToString

//sonuç verisi string ezilerek geriye return ediliyor

public override string ToString()

{

StringBuilder sb = new StringBuilder();

sb.Append(Cozum);

return sb.ToString();

}

#endregion ToString

}

}

using System;

using System.Collections.Generic;

using System.Text;

using static BirKelimeBirIslem.UI.Win.Classes.Islem.Enums.Enum;

namespace BirKelimeBirIslem.UI.Win.Classes

{

public class Cozum

{

#region Properties

public List<int> TumSayilar { get; private set; }

public int Hedef2 { get; private set; }

private Hesapla Hesapla { get; set; }

private int enYakin = int.MaxValue;

public int puan;

#endregion Properties

#region Constructor

public Cozum(int \_hedef, List<int> \_tekbasamakli, int \_ikibasamakli)

{

TumSayilar = new List<int>();

this.Hedef2 = \_hedef;

this.TumSayilar.Add(\_ikibasamakli);

this.TumSayilar.AddRange(\_tekbasamakli);

Coz();

}

#endregion Constructor

#region CozunArama

public void Coz()

{

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

{

Hesapla \_denklem = new Hesapla(TumSayilar[i]);

List<int> arttikliste = KisaListeOlustur(TumSayilar, i);

if (CozumAra(\_denklem, arttikliste))

break;

}

}

private List<int> KisaListeOlustur(List<int> \_eskiliste, int \_sirano)

{

List<int> yeniliste = new List<int>();

for (int i = 0; i < \_eskiliste.Count; i++)

if (i != \_sirano)

yeniliste.Add(\_eskiliste[i]);

return yeniliste;

}

private bool CozumAra(Hesapla \_denklembaslangici, List<int> \_artikliste)

{

for (int i = 0; i < \_artikliste.Count; i++)

{

foreach (Islem\_Isaret islem in Enum.GetValues(typeof(Islem\_Isaret)))

{

List<int> yeniartikliste = KisaListeOlustur(\_artikliste, i);

int siradakisayi = \_artikliste[i];

if (CozumeKartiEkle(siradakisayi, islem, \_denklembaslangici, yeniartikliste))

return true;

}

}

return false;

}

private bool CozumeKartiEkle(int \_siradakisayi, Islem\_Isaret \_islem, Hesapla \_denklembaslangici, List<int> \_artikliste)

{

Hesapla denklem = new Hesapla(\_denklembaslangici, \_islem, \_siradakisayi);

if (Math.Abs(Hedef2 - denklem.DegerHesapla) <= enYakin)

{

enYakin = Math.Abs(Hedef2 - denklem.DegerHesapla);

this.Hesapla = denklem;

if (denklem.DegerHesapla == Hedef2) return true;

}

if (\_artikliste.Count == 0)

{

return false;

}

return CozumAra(denklem, \_artikliste);

}

#endregion Solve

#region ToString

public override string ToString()

{

StringBuilder sb = new StringBuilder();

sb.Append(this.Hesapla.ToString());

int fark = Math.Abs(this.Hesapla.DegerHesapla - Hedef2);

string ifade;

switch (fark)

{

case 0:

puan = 10;

ifade = "Alınan Puan: 10" + Environment.NewLine + "(Tam Sonuç)";

break;

case 1:

puan = 9;

ifade = "Alınan Puan: 9" + Environment.NewLine + "(Bir Yaklaşık Sonuç)";

break;

case 2:

puan = 8;

ifade = "Alınan Puan: 8" + Environment.NewLine + "(İki Yaklaşık Sonuç)";

break;

case 3:

puan = 7;

ifade = "Alınan Puan: 7" + Environment.NewLine + "(Üç Yaklaşık Sonuç)";

break;

case 4:

puan = 6;

ifade = "Alınan Puan: 6" + Environment.NewLine + "(Dört Yaklaşık Sonuç)";

break;

case 5:

puan = 5;

ifade = "Alınan Puan: 5" + Environment.NewLine + "(Beş Yaklaşık Sonuç)";

break;

case 6:

puan = 4;

ifade = "Alınan Puan: 4" + Environment.NewLine + "(Altı Yaklaşık Sonuç)";

break;

case 7:

puan = 3;

ifade = "Alınan Puan: 3" + Environment.NewLine + "(Yedi Yaklaşık Sonuç)";

break;

case 8:

puan = 2;

ifade = "Alınan Puan: 2" + Environment.NewLine + "(Sekiz Yaklaşık Sonuç)";

break;

case 9:

puan = 1;

ifade = "Alınan Puan: 1" + Environment.NewLine + "(Dokuz Yaklaşık Sonuç)";

break;

default:

puan = 0;

ifade = "Puan Alamadınız!" + Environment.NewLine + "Çözüm Bulunamadı";

break;

}

sb.Append(Environment.NewLine + ifade);

return sb.ToString();

}

#endregion ToString

}

}

using System;

using System.Collections.Generic;

using System.Text;

using static BirKelimeBirIslem.UI.Win.Classes.Islem.Enums.Enum;

namespace BirKelimeBirIslem.UI.Win.Classes

{

public class Hesapla:OyunAyarlari

{

#region Properties

private List<int> SiradakiSayi { get; set; }

private List<Islem\_Isaret> Operatorler { get; set; }

public int DegerHesapla

{

//elde eilen işlem ve çözüm verileri 4 işleme(enum) tabi tutulup yaklaşıklık hesaplanıyor

get

{

int deger = SiradakiSayi[0];

for (int i = 1; i < SiradakiSayi.Count; i++)

{

switch (Operatorler[i - 1])

{

case Islem\_Isaret.Ekle:

deger += SiradakiSayi[i];

break;

case Islem\_Isaret.Cikar:

deger -= SiradakiSayi[i];

break;

case Islem\_Isaret.Carp:

deger \*= SiradakiSayi[i];

break;

case Islem\_Isaret.Bol:

deger /= SiradakiSayi[i];

break;

}

}

return deger;

}

}

#endregion Properties

#region Constructors

public Hesapla(int \_siradaki\_sayi)

{

//listede gelen veriler ve işaretler değişkenlere aktarılıyor.

this.SiradakiSayi = new List<int>() { \_siradaki\_sayi };

this.Operatorler = new List<Islem\_Isaret>();

}

public Hesapla(Hesapla \_denklembaslangici, Islem\_Isaret \_islem, int \_siradakisayi)

{

//listedeki veriler ayrıştırılıyor

this.SiradakiSayi = new List<int>((int[])\_denklembaslangici.SiradakiSayi.ToArray().Clone());

this.SiradakiSayi.Add(\_siradakisayi);

this.Operatorler = new List<Islem\_Isaret>((Islem\_Isaret[])\_denklembaslangici.Operatorler.ToArray().Clone());

this.Operatorler.Add(\_islem);

}

#endregion Constructors

#region ToString

public override string ToString()

{

//en yakın çözüm string olarak return ediliyor

StringBuilder sb = new StringBuilder();

int siradaki = SiradakiSayi[0];

int \_siradakideger = -1;

for (int i = 1; i < SiradakiSayi.Count; i++)

{

string satir = YeniIslemSatiriOlustur(siradaki, i, ref \_siradakideger);

sb.Append(satir);

sb.Append(Environment.NewLine);

siradaki = \_siradakideger;

}

return sb.ToString();

}

private string YeniIslemSatiriOlustur(int \_simdikisayi, int \_kartsirasi, ref int \_siradakideger)

{

//veri ve işaret hesaplamaya gönderiliyor

int siradaki\_sayi = SiradakiSayi[\_kartsirasi];

Islem\_Isaret islem = (Islem\_Isaret)Operatorler[\_kartsirasi - 1];

string islemsembol = IslemSembolEkle(islem);

\_siradakideger = YeniDegerHesapla(\_simdikisayi, siradaki\_sayi, islem);

return string.Format("{0} {1} {2} = {3}", \_simdikisayi, islemsembol, siradaki\_sayi, \_siradakideger);

}

private string IslemSembolEkle(Islem\_Isaret Operator)

{

//enum lar işlem işaretlerine dönüşüyor

switch (Operator)

{

case Islem\_Isaret.Ekle:

return "+";

case Islem\_Isaret.Cikar:

return "-";

case Islem\_Isaret.Carp:

return "\*";

case Islem\_Isaret.Bol:

return "/";

default:

return null;

}

}

private int YeniDegerHesapla(int \_sayi1, int \_sayi2, Islem\_Isaret \_islem)

{

//sıradaki 2 sayı arasındaki işlem sonucu dönüyor

switch (\_islem)

{

case Islem\_Isaret.Ekle:

return \_sayi1 + \_sayi2;

case Islem\_Isaret.Cikar:

return \_sayi1 - \_sayi2;

case Islem\_Isaret.Carp:

return \_sayi1 \* \_sayi2;

case Islem\_Isaret.Bol:

return \_sayi1 / \_sayi2;

default:

return -1;

}

}

#endregion ToString

}

}

using BirKelimeBirIslem.UI.Win.Classes;

using BirKelimeBirIslem.UI.Win.Forms.Base;

using System;

namespace BirKelimeBirIslem.UI.Win.Forms.IslemForm

{

public partial class IslemKarti : BaseForm

{

public IslemKarti()

{

InitializeComponent();

btnOlustur.ItemClick += BtnOlustur\_ItemClick;

btnBasla.ItemClick += BtnBasla\_ItemClick;

tglGiris.ItemClick += TglGiris\_ItemClick;

}

OyunAyarlari oyunayarlari =new OyunAyarlari();

private DateTime \_startTime = DateTime.Now;

private TimeSpan \_timeSpan = new TimeSpan(0, 1, 0); // 1 minute 10 seconds as total time

System.Diagnostics.Stopwatch st = new System.Diagnostics.Stopwatch();

private void VeriKontrol()

{

//Testcase uygulanıyor

if (string.IsNullOrEmpty(txtSayi1.Text)) txtSayi1.Text = "1";

if (string.IsNullOrEmpty(txtSayi2.Text)) txtSayi2.Text = "1";

if (string.IsNullOrEmpty(txtSayi3.Text)) txtSayi3.Text = "1";

if (string.IsNullOrEmpty(txtSayi4.Text)) txtSayi4.Text = "1";

if (string.IsNullOrEmpty(txtSayi5.Text)) txtSayi5.Text = "1";

if (string.IsNullOrEmpty(txtSayi6.Text)) txtSayi6.Text = "10";

if (string.IsNullOrEmpty(txtHedef.Text)) txtSayi6.Text = "100";

}

private void IslemKarti\_Load(object sender, System.EventArgs e)

{

lblSure.Caption = "00:00";

}

private void TglGiris\_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)

{

try

{

//random-user oyun ayarı yapılıyor

if (tglGiris.Checked)

{

txtSayi1.ReadOnly = false;

txtSayi2.ReadOnly = false;

txtSayi3.ReadOnly = false;

txtSayi4.ReadOnly = false;

txtSayi5.ReadOnly = false;

txtSayi6.ReadOnly = false;

txtHedef.ReadOnly = false;

}

else

{

txtSayi1.ReadOnly = true;

txtSayi2.ReadOnly = true;

txtSayi3.ReadOnly = true;

txtSayi4.ReadOnly = true;

txtSayi5.ReadOnly = true;

txtSayi6.ReadOnly = true;

txtHedef.ReadOnly = true;

}

}

catch (Exception)

{

throw;

}

}

private void BtnBasla\_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)

{

try

{

VeriKontrol();

lstIslemler.Items.Clear();

oyunayarlari.TekBasamakli[0] = Convert.ToInt32(txtSayi1.Text);

oyunayarlari.TekBasamakli[1] = Convert.ToInt32(txtSayi2.Text);

oyunayarlari.TekBasamakli[2] = Convert.ToInt32(txtSayi3.Text);

oyunayarlari.TekBasamakli[3] = Convert.ToInt32(txtSayi4.Text);

oyunayarlari.TekBasamakli[4] = Convert.ToInt32(txtSayi5.Text);

oyunayarlari.IkiBasamakli = Convert.ToInt32(txtSayi6.Text);

oyunayarlari.HedefSayi = Convert.ToInt32(txtHedef.Text);

st.Start();

var ifd = oyunayarlari.Basla();

st.Stop();

lblSure.Caption = st.Elapsed.Seconds + ":" + st.Elapsed.Milliseconds;

lstIslemler.Items.Add(ifd.ToString());

lblPuan.Caption = ifd.puan.ToString();

st.Reset();

}

catch (Exception)

{

throw;

}

}

private void BtnOlustur\_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)

{

try

{

//random sayılar üretiliyor

oyunayarlari = new OyunAyarlari();

lstIslemler.Items.Clear();

txtSayi1.Text = oyunayarlari.TekBasamakli[0].ToString();

txtSayi2.Text = oyunayarlari.TekBasamakli[1].ToString();

txtSayi3.Text = oyunayarlari.TekBasamakli[2].ToString();

txtSayi4.Text = oyunayarlari.TekBasamakli[3].ToString();

txtSayi5.Text = oyunayarlari.TekBasamakli[4].ToString();

txtSayi6.Text = oyunayarlari.IkiBasamakli.ToString();

txtHedef.Text = oyunayarlari.HedefSayi.ToString();

}

catch (Exception)

{

throw;

}

}

}

}