MEDIC

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tutoring\_PAW\_15.\_06.\_2022

{

public class Medic:ICalculVenitMedic

{

private string nume;

private string specializare;

private List<Pacient> listaPacienti;

public Medic(string nume, string specializare)

{

this.nume = nume;

this.specializare = specializare;

this.listaPacienti = new List<Pacient>();

}

public string Nume { get => nume; set => nume = value; }

public string Specializare { get => specializare; set => specializare = value; }

public List<Pacient> ListaPacienti { get => listaPacienti; set => listaPacienti = value; }

public override string ToString()

{

StringBuilder sb = new StringBuilder();

sb.Append("Medicul " + nume + " are specializarea " + specializare + "si pacientii: ");

foreach(Pacient p in listaPacienti)

{

sb.Append(Environment.NewLine + p.ToString());

}

return sb.ToString();

}

public float calculeazaVenit()

{

float suma = 0;

foreach(Pacient p in ListaPacienti)

{

suma += p.Tarif\_consultatie;

}

return suma;

}

//Medic m;

//m+=p

public static Medic operator+(Medic m, Pacient p)

{

m.listaPacienti.Add(p);

return m;

}

//Medic m;

//m[0]

public Pacient this[int index]

{

get

{

if(index >= 0 && index < listaPacienti.Count)

{

return listaPacienti[index];

}

else

{

throw new IndexOutOfRangeException();

}

}

set

{

if (index >= 0 && index < listaPacienti.Count)

{

listaPacienti[index] = value;

}

else

{

throw new IndexOutOfRangeException();

}

}

}

}

}

PACIENT

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tutoring\_PAW\_15.\_06.\_2022

{

public class Pacient:ICloneable, IComparable

{

string nume;

int varsta;

float tarif\_consultatie;

public Pacient(string nume, int varsta, float tarif\_consultatie)

{

this.nume = nume;

this.varsta = varsta;

this.tarif\_consultatie = tarif\_consultatie;

}

public string Nume { get => nume; set => nume = value; }

public int Varsta { get => varsta; set => varsta = value; }

public float Tarif\_consultatie { get => tarif\_consultatie; set => tarif\_consultatie = value; }

public object Clone()

{

Pacient p = (Pacient)this.MemberwiseClone();

return p;

}

public int CompareTo(object obj)

{

Pacient p = (Pacient)obj;

if(this.tarif\_consultatie > p.Tarif\_consultatie)

{

return 1;

}

else

{

return -1;

}

}

public override string ToString()

{

return "Pacientul "+nume+" in varsta de: "+varsta+" plateste "+tarif\_consultatie;

}

}

}

PROGRAM

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Tutoring\_PAW\_15.\_06.\_2022

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

ICALCULVENIT

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tutoring\_PAW\_15.\_06.\_2022

{

interface ICalculVenitMedic

{

float calculeazaVenit();

}

}

FORM 1

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;

using System.Data.OleDb;

namespace Tutoring\_PAW\_15.\_06.\_2022

{

public partial class Form1 : Form

{

string conexiune\_string;

Medic m;

public Form1()

{

InitializeComponent();

m = new Medic("Medic1", "Specializare1");

textBox1.Text = m.Nume + " - " + m.Specializare;

conexiune\_string = "Provider = Microsoft.ACE.OLEDB.12.0; Data Source = BD.accdb";

}

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

{

Form2 form2 = new Form2();

form2.ShowDialog();

if(form2.DialogResult == DialogResult.OK)

{

try

{

string nume = form2.tbNume.Text;

int varsta = Convert.ToInt32(form2.tbVarsta.Text);

float tarif = float.Parse(form2.tbTarif.Text);

Pacient p = new Pacient(nume, varsta, tarif);

m += p;

ListViewItem itm = new ListViewItem(nume);

itm.SubItems.Add(varsta.ToString());

itm.SubItems.Add(tarif.ToString());

lvPacienti.Items.Add(itm);

lvPacienti.Refresh();

}

catch(Exception ex)

{

MessageBox.Show("S-a generat o exceptie: " + ex.Message);

}

}

}

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

{

Form2 form2 = new Form2();

int poz;

if(lvPacienti.SelectedItems.Count > 0)

{

ListViewItem itm = lvPacienti.SelectedItems[0];

poz = itm.Index;

form2.tbNume.Text = itm.SubItems[0].Text;

form2.tbVarsta.Text = itm.SubItems[1].Text;

form2.tbTarif.Text = itm.SubItems[2].Text;

form2.ShowDialog();

if(form2.DialogResult == DialogResult.OK)

{

itm.SubItems[0].Text = form2.tbNume.Text;

itm.SubItems[1].Text = form2.tbVarsta.Text;

itm.SubItems[2].Text = form2.tbTarif.Text;

Pacient p = m.ListaPacienti[poz];

p.Nume = form2.tbNume.Text;

p.Varsta = Convert.ToInt32(form2.tbVarsta.Text);

p.Tarif\_consultatie = float.Parse(form2.tbTarif.Text);

m[poz] = p;

}

}

else

{

MessageBox.Show("Nu s-a selectat un element!");

}

}

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

{

int poz;

if (lvPacienti.SelectedItems.Count > 0)

{

ListViewItem itm = lvPacienti.SelectedItems[0];

poz = itm.Index;

itm.Remove();

m.ListaPacienti.RemoveAt(poz);

}

else

{

MessageBox.Show("Nu s-a selectat element!");

}

}

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

{

tbVenit.Text = "Venitul medicului este: " + m.calculeazaVenit();

}

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

{

Form3 form3 = new Form3(m);

form3.Show();

}

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

{

OleDbConnection conexiune = new OleDbConnection(conexiune\_string);

try

{

conexiune.Open();

OleDbCommand comanda = new OleDbCommand("Select \* From Pacienti");

comanda.Connection = conexiune;

OleDbDataReader reader = comanda.ExecuteReader();

while(reader.Read())

{

ListViewItem lvi = new ListViewItem(reader["Nume"].ToString());

lvi.SubItems.Add(reader["Varsta"].ToString());

lvi.SubItems.Add(reader["Tarif"].ToString());

lvPacienti.Items.Add(lvi);

m.ListaPacienti.Add(new Pacient(reader["Nume"].ToString(),

Convert.ToInt32(reader["Varsta"].ToString()),

float.Parse(reader["Tarif"].ToString())));

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

conexiune.Close();

}

}

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

{

OleDbConnection conexiune = new OleDbConnection(conexiune\_string);

try

{

if(m.ListaPacienti.Count > 0)

{

conexiune.Open();

foreach (Pacient p in m.ListaPacienti)

{

OleDbCommand command = new OleDbCommand();

command.Connection = conexiune;

command.CommandText = "SELECT MAX(id) FROM Pacienti";

int id = Convert.ToInt32(command.ExecuteScalar());

command.CommandText = "INSERT INTO Pacienti VALUES (?,?,?,?)";

command.Parameters.Add("id", OleDbType.Integer).Value = id + 1;

command.Parameters.Add("nume", OleDbType.Char, 20).Value = p.Nume;

command.Parameters.Add("varsta", OleDbType.Integer).Value = p.Varsta;

command.Parameters.Add("tarif", OleDbType.Double).Value = p.Tarif\_consultatie;

command.ExecuteNonQuery();

}

MessageBox.Show("S-au salvat pacientii in bd");

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

conexiune.Close();

}

}

private void textBox2\_MouseDown(object sender, MouseEventArgs e)

{

textBox2.DoDragDrop(textBox2.Text, DragDropEffects.Copy | DragDropEffects.Move);

}

private void panel1\_DragEnter(object sender, DragEventArgs e)

{

e.Effect = DragDropEffects.None;

if((e.KeyState & 8) == 8

&& (e.AllowedEffect & DragDropEffects.Copy) == DragDropEffects.Copy)

{

e.Effect = DragDropEffects.Copy;

}

else

{

if((e.AllowedEffect & DragDropEffects.Move) == DragDropEffects.Move)

{

e.Effect = DragDropEffects.Move;

}

}

}

private void panel1\_DragDrop(object sender, DragEventArgs e)

{

Graphics g = Graphics.FromHwnd(panel1.Handle);

string text = e.Data.GetData(DataFormats.Text).ToString();

g.DrawString(text, new Font(FontFamily.GenericSansSerif, 12),

new SolidBrush(Color.Black), 10, 10);

if(e.Effect == DragDropEffects.Move)

{

textBox2.Text = "";

}

}

private void lvPacienti\_SelectedIndexChanged(object sender, EventArgs e)

{

}

}

}

ALTCEVA

PROGRAM

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Tutoring\_PAW\_05.\_06.\_2022

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

JOB

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tutoring\_PAW\_05.\_06.\_2022

{

class Job

{

private Interviu[] vectorInterviuri;

private string numeJob;

//private List<Interviu> listaInterviuri;

public Job(Interviu[] vectorInterviuri, string numeJob)

{

//this.vectorInterviuri = vectorInterviuri;

this.vectorInterviuri = new Interviu[vectorInterviuri.Length];

for(int i=0; i<vectorInterviuri.Length; i++)

{

this.vectorInterviuri[i] = vectorInterviuri[i];

}

this.numeJob = numeJob;

}

//public Job(string numeJob)

//{

// this.numeJob = numeJob;

// this.listaInterviuri = new List<Interviu>();

//}

public int numarInterviuri()

{

return vectorInterviuri.Length;

}

public Interviu[] VectorInterviuri

{

get

{

return vectorInterviuri;

}

set

{

vectorInterviuri = value;

}

}

public static Job operator+(Job j, Interviu i)

{

//j.vectorInterviuri.Append(i);

//j.listaInterviuri.Add(i);

Interviu[] interviuriNoi = new Interviu[j.vectorInterviuri.Length + 1];

for(int index=0; index<j.vectorInterviuri.Length; index++)

{

interviuriNoi[index] = j.vectorInterviuri[index];

}

interviuriNoi[interviuriNoi.Length - 1] = i;

j.vectorInterviuri = interviuriNoi;

return j;

}

public Interviu this[int index]

{

get

{

if(index >= 0 && index < vectorInterviuri.Length)

{

return vectorInterviuri[index];

}

else

{

throw new Exception();

}

//if (index >= 0 && index < listaInterviuri.Count)

//{

// return listaInterviuri[index];

//}

//else

//{

// throw new Exception();

//}

}

set

{

if (index >= 0 && index < vectorInterviuri.Length)

{

vectorInterviuri[index] = value;

}

else

{

throw new Exception();

}

//if (index >= 0 && index < listaInterviuri.Count)

//{

// listaInterviuri[index] = value;

//}

//else

//{

// throw new Exception();

//}

}

}

}

}

INTERVIU

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tutoring\_PAW\_05.\_06.\_2022

{

class Interviu:ICalculPunctaj

{

private DateTime data\_interviu;

private string specializare;

private string numeCandidat;

private float punctajTeorie;

private float punctajPractic;

public Interviu(DateTime data\_interviu, string specializare, string numeCandidat, float punctajTeorie, float punctajPractic)

{

this.data\_interviu = data\_interviu;

this.specializare = specializare ?? throw new ArgumentNullException(nameof(specializare));

this.numeCandidat = numeCandidat ?? throw new ArgumentNullException(nameof(numeCandidat));

this.punctajTeorie = punctajTeorie;

this.punctajPractic = punctajPractic;

}

public DateTime Data\_interviu { get => data\_interviu; set => data\_interviu = value; }

public string Specializare { get => specializare; set => specializare = value; }

public string NumeCandidat { get => numeCandidat; set => numeCandidat = value; }

public float PunctajTeorie { get => punctajTeorie; set => punctajTeorie = value; }

public float PunctajPractic { get => punctajPractic; set => punctajPractic = value; }

public float calculeazaPunctaj()

{

return (punctajPractic + punctajTeorie) / 2;

}

public override string ToString()

{

return $"Data interviu {data\_interviu}, nume candidat {numeCandidat}, " +

$"specializarea {specializare} a obtinut la teorie {punctajTeorie} si" +

$"la practic {punctajPractic}";

}

}

}

ICALCULPUNCTAJ

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tutoring\_PAW\_05.\_06.\_2022

{

public interface ICalculPunctaj

{

float calculeazaPunctaj();

}

}

FORM 1

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 Tutoring\_PAW\_05.\_06.\_2022

{

public partial class Form1 : Form

{

Job job;

public Form1()

{

InitializeComponent();

job = new Job(new Interviu[] { }, "IT");

job += new Interviu(DateTime.Now, "aa", "aa", 10, 10);

}

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

{

Form2 frm2 = new Form2();

frm2.ShowDialog();

if(frm2.DialogResult == DialogResult.OK)

{

DateTime data = frm2.dtpData.Value;

string nume = frm2.tbNume.Text;

string specializare = frm2.cbSpecializare.SelectedItem.ToString();

float punctajPractic = float.Parse(frm2.tbPctPractic.Text);

float punctajTeoretic = float.Parse(frm2.tbPctTeorie.Text);

Interviu i = new Interviu(data, specializare, nume, punctajTeoretic,

punctajPractic);

job += i;

ListViewItem itm = new ListViewItem(data.ToString());

itm.SubItems.Add(nume);

itm.SubItems.Add(specializare);

itm.SubItems.Add(i.calculeazaPunctaj().ToString());

listView1.Items.Add(itm);

listView1.Refresh();

}

}

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

{

if(listView1.SelectedItems.Count > 0)

{

listView1.SelectedItems[0].Remove();

var interviuriNesterse = new Interviu[] { };

foreach(Interviu i in job.VectorInterviuri)

{

if (!listView1.SelectedItems[0].SubItems["Nume"].ToString().Equals(i.NumeCandidat))

{

interviuriNesterse.Append(i);

}

}

job.VectorInterviuri = interviuriNesterse;

}

}

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

{

//chart1.ChartAreas.Add("Punctaje");

//chart1.Series.Clear();

//chart1.Series.Add("Series punctaje");

//for(int i=0; i<job.numarInterviuri(); i++)

//{

// chart1.Series[0].Points.AddXY(i + 1, job[i].calculeazaPunctaj());

//}

panel1.Invalidate();

//MessageBox.Show(job.numarInterviuri().ToString());

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

double margine = 10;

Graphics g = e.Graphics;

Rectangle r = new Rectangle((int)(panel1.ClientRectangle.X + margine),

(int)(panel1.ClientRectangle.Y + 4 \* margine), (int)(panel1.ClientRectangle.Width - 2 \* margine),

(int)(panel1.ClientRectangle.Height - 5 \* margine));

Pen pen = new Pen(Color.Red, 3);

g.DrawRectangle(pen, r);

double latime = r.Width / job.numarInterviuri() / 3;

double distanta = (r.Width - job.numarInterviuri() \* latime) / (job.numarInterviuri() + 1);

double hMax = 20;

Brush br = new SolidBrush(Color.Blue);

Rectangle[] rectangles = new Rectangle[job.numarInterviuri()];

for (int i = 0; i < job.numarInterviuri(); i++)

{

rectangles[i] = new Rectangle((int)(r.Location.X + (i + 1) \* distanta + i \* latime),

(int)(r.Location.Y + r.Height - job.VectorInterviuri[i].PunctajPractic / hMax \* r.Height),

(int)latime,

(int)(job.VectorInterviuri[i].PunctajPractic / hMax \* r.Height));

}

g.FillRectangles(br, rectangles);

}

private void listView1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

}

}