KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Objektinis programavimas II (P175B123)

Darbų aplankas

Atliko:

IFF-7/7 gr. studentas

Matas Goštautas

2018 m. kovo 19 d.

Priėmė:

Doc. Lina Narbutaitė

KAUNAS 2018

TURINYS

1. Rekursija (L1) 3

1.1. Darbo užduotis 3

1.2. Grafinės vartotojo sąsajos schema 3

1.3. Sąsajoje panaudotų komponentų keičiamos savybės 4

1.4. Klasių diagrama 4

1.5. Programos vartotojo vadovas 4

1.6. Programos tekstas 5

1.7. Pradiniai duomenys ir rezultatai 6

1.8. Dėstytojo pastabos 17

2. Dinaminis atminties valdymas (L2) 18

2.1. Darbo užduotis 18

2.2. Grafinės vartotojo sąsajos schema 18

2.3. Sąsajoje panaudotų komponentų keičiamos savybės 19

2.4. Klasių diagrama 20

2.5. Programos vartotojo vadovas 20

2.6. Programos tekstas 21

2.7. Pradiniai duomenys ir rezultatai 33

2.8. Dėstytojo pastabos 36

3. Bendrinės klasės ir sąsajos (L3) 41

3.1. Darbo užduotis 41

3.2. Grafinės vartotojo sąsajos schema 41

3.3. Sąsajoje panaudotų komponentų keičiamos savybės 41

3.4. Klasių diagrama 41

3.5. Programos vartotojo vadovas 41

3.6. Programos tekstas 41

3.7. Pradiniai duomenys ir rezultatai 41

3.8. Dėstytojo pastabos 41

4. Kolekcijos ir išimčių valdymas (L4) 42

4.1. Darbo užduotis 42

4.2. Grafinės vartotojo sąsajos schema 42

4.3. Sąsajoje panaudotų komponentų keičiamos savybės 42

4.4. Klasių diagrama 42

4.5. Programos vartotojo vadovas 42

4.6. Programos tekstas 42

4.7. Pradiniai duomenys ir rezultatai 42

4.8. Dėstytojo pastabos 42

5. Deklaratyvusis programavimas (L5) 43

5.1. Darbo užduotis 43

5.2. Grafinės vartotojo sąsajos schema 43

5.3. Sąsajoje panaudotų komponentų keičiamos savybės 43

5.4. Klasių diagrama 43

5.5. Programos vartotojo vadovas 43

5.6. Programos tekstas 43

5.7. Pradiniai duomenys ir rezultatai 43

5.8. Dėstytojo pastabos 43

# Rekursija (L1)

## Darbo užduotis

**LD\_14.Salos.**

Robotas iš didelio aukščio nufotografavo stačiakampį Žemės paviršiaus plotą. Gautas vaizdas buvo sudalintas kvadratiniais langeliais n × m (1 ≤ n, m ≤ 100). Langelis tamsus, jeigu jis atitinka sausumos dalelę ir šviesus, jeigu atitinka vandens plotelį.

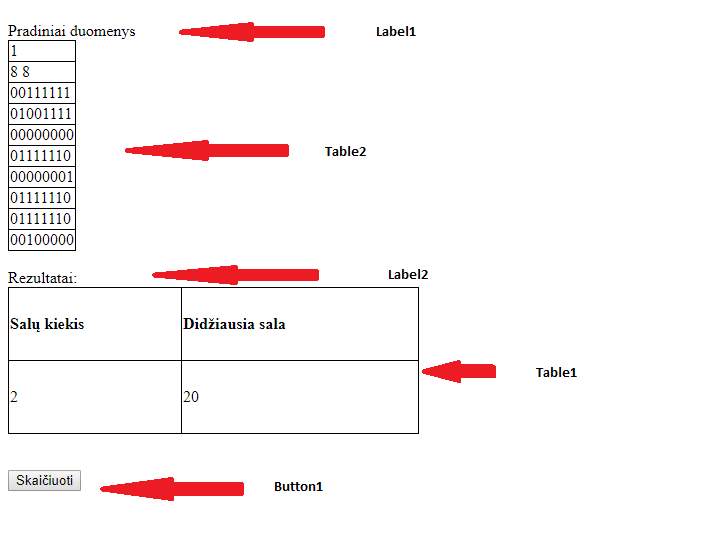
Parašykite programą, kuri suskaičiuotų, kiek yra salų ir kokio dydžio didžiausia sala. Sala – tai gretimų tamsių langelių visuma, iš visų pusių apribota šviesiais. Sala taip pat laikoma tamsių langelių visuma, esanti ant nufotografuoto ploto ribos (ne iš visų pusių supa vanduo).

Programa turi analizuoti k (1 ≤ k ≤ 10) tokių žemės plotų nuotraukų.

**Duomenys** surašyti tekstiniame faile U3.txt, kur pirmoje eilutėje yra k. Toliau – kiekvieno stačiakampio duomenys: pirmoje stačiakampio aprašo eilutėje yra n ir m reikšmės, o kitose eilutėse – stačiakampio langelių reikšmės eilutėmis. Tamsų langelį atitinka simbolis ‘1’, o šviesų ‘0‘.

**Rezultatai.** Ekrane spausdinti kiekvienoje eilutėje po du skaičius: kiekvieno paviršiaus ploto salų skaičių ir didžiausios salos langelių skaičių

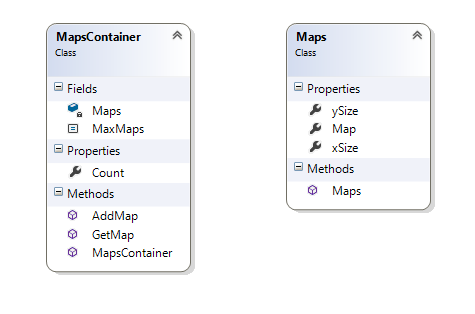
## Grafinės vartotojo sąsajos schema



## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Label1 | Text | Duomenys: |
| Label2 | Text | Rezultatai: |
| Table1 | BorderColor  BorderStyle  BorderWidth  GridLines | Black  Solid  1 px  Both |
| Table2 | BorderColor  BorderStyle  BorderWidth  GridLines | Black  Solid  1px  Both |
| Button1 | Text | Skaičiuoti |

## Klasių diagrama



## Programos vartotojo vadovas

Surašome duomenis tekstiniame faile U3.txt, kur pirmoje eilutėje yra nufotografuotų plotų skaičius. Toliau – kiekvieno stačiakampio duomenys: pirmoje stačiakampio aprašo eilutėje yra nuotraukų y ir x koordinatės, o kitose eilutėse – stačiakampio langelių reikšmės eilutėmis. Tamsų langelį atitinka simbolis ‘1’, o šviesų ‘0‘.

Įjungę programą paspaudžiame mygtuką „Skaičiuoti“. Lentelėje po užrašu „Duomenys:“ yra surašomi duomenys iš U3.txt tekstinio failo, o po „Rezultatai“ yra surašyti programos suskaičiuoti rezultatai.

## Programos tekstas

Map.cs klasė:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

public class Maps

{

public Maps(int[,] map, int xsize, int ysize)

{

Map = map;

xSize = xsize;

ySize = ysize;

}

public int[,] Map { get; set; }

public int xSize { get; set; }

public int ySize { get; set; }

}

MapsContainer.cs konteinerinė klasė:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

public class MapsContainer

{

public const int MaxMaps = 50;

private Maps[] Maps;

public int Count { get; private set; }

public MapsContainer(int size)

{

Maps = new Maps[size];

}

public void AddMap(Maps map)

{

Maps[Count++] = map;

}

public Maps GetMap(int index)

{

return Maps[index];

}

}

1lab.aspx:

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="1lab.aspx.cs" Inherits="\_1lab" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

</head>

<body>

<form id="form1" runat="server">

<div>

<br />

<asp:Label ID="Label1" runat="server" Text="Pradiniai duomenys"></asp:Label>

<asp:Table ID="Table2" runat="server" BorderColor="Black" BorderStyle="Solid" BorderWidth="1px" GridLines="Both" Height="30px" HorizontalAlign="Justify" Width="29px">

</asp:Table>

<br />

<asp:Label ID="Label2" runat="server" Text="Rezultatai:"></asp:Label>

<br />

<asp:Table ID="Table1" runat="server" Height="147px" Width="411px" BorderColor="Black" BorderStyle="Solid" BorderWidth="1px" ForeColor="Black" GridLines="Both">

</asp:Table>

<br />

<br />

<asp:Button ID="Button1" runat="server" Text="Skaičiuoti" OnClick="Button1\_Click" />

</div>

</form>

</body>

</html>

1lab.aspx.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.IO;

public partial class \_1lab : System.Web.UI.Page

{

public const int xMax = 100; //maksimalus x ašies dydis

public const int yMax = 100; //maksimalys y ašies dydis

const string data = "App\_Data/U2.txt"; //duomenų failas

protected void Page\_Load(object sender, EventArgs e)

{

}

/// <summary>

/// Spausdina salos rezultatus į Table1

/// </summary>

/// <param name="count"> salų skaičius plote</param>

/// <param name="max"> didžiausios salos plote dydis</param>

private void Table(string count, string max)

{

TableCell Max = new TableCell();

Max.Text = max;

TableCell Count = new TableCell();

Count.Text = count;

TableRow row = new TableRow();

row.Cells.Add(Count);

row.Cells.Add(Max);

Table1.Rows.Add(row);

}

/// <summary>

/// Spausdina rezultatus į Rezultatai.txt

/// </summary>

/// <param name="results"> rezultatai</param>

/// <param name="Count"> Plotų kiekis</param>

private void ToFile(int[,]results, int Count)

{

string path = Server.MapPath("App\_Data/Rezultatai.txt");

using (StreamWriter writer = new StreamWriter(path))

{

writer.WriteLine("{0,-15}{1,-15}", "Salų skaičius", "Didž. sala");

writer.WriteLine("------------------------------");

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

{

writer.WriteLine("{0,-15}{1,-15}", results[i,0], results[i,1]);

}

}

}

/// <summary>

/// Spausdina duomenis

/// </summary>

/// <param name="results"> rezultatai</param>

/// <param name="Count"> Plotų kiekis </param>

private void Print(int[,] results, int Count)

{

TableRow row = new TableRow();

TableCell pavadinimas = new TableCell();

pavadinimas.Text = "<b>Salų kiekis</b>";

row.Cells.Add(pavadinimas);

TableCell kaina = new TableCell();

kaina.Text = "<b>Didžiausia sala</b>";

row.Cells.Add(kaina);

Table1.Rows.Add(row);

ToFile(results, Count);

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

{

Table(results[i, 0].ToString(), results[i, 1].ToString());

}

}

/// <summary>

/// Gražina rezultatus

/// </summary>

/// <param name="count"> Žemėlapių kiekis</param>

/// <returns> gražina žemėlapius</returns>

public int[,] Islands(out int count)

{

MapsContainer Maps = Reading();

PrintDataFile(Maps);

int[,] results = new int[Maps.Count, 2];

Do(Maps, ref results);

count = Maps.Count;

PrintDataWeb();

return results;

}

/// <summary>

/// Skaito duomenų faila

/// </summary>

/// <returns> gražina žemėlapių konteinerį</returns>

private static MapsContainer Reading()

{

using (StreamReader reader = new StreamReader(@data))

{

int count = int.Parse(reader.ReadLine());

MapsContainer a = new MapsContainer(count);

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

{

string[] dims = reader.ReadLine().Split();

int[] dim = new int[3];

dim[1] = int.Parse(dims[0]) + 2; // dim[1] - Y ašis ilgis

dim[2] = int.Parse(dims[1]) + 2; // dim[2] - X ašis ilgis

int[,] Empty = new int[xMax, yMax];

int[,] area = new int[xMax, yMax];

area = Empty;

for (int j = 1; j < dim[1] - 1; j++)

{

string line = reader.ReadLine();

for (int k = 1; k < dim[2] - 1; k++)

{

area[k, j] = int.Parse(line[k - 1].ToString());

}

}

Maps map = new Maps(area, dim[2], dim[1]);

a.AddMap(map);

}

return a;

}

}

/// <summary>

/// Atlieka skaičiavimo veiksmus

/// </summary>

/// <param name="Map"> Žemėlapių konteineris</param>

/// <param name="results"> rezultatų dvimatys masyvas [salų skaičius,2]</param>

private static void Do(MapsContainer Map, ref int[,] results)

{

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

{

int[,] Empty = new int[xMax, yMax];

int[,] area = new int[xMax, yMax];

area = Empty;

area = Map.GetMap(i).Map;

int IslandCount = 0;

int MaxTwos = 0;

for (int j = 1; j < Map.GetMap(i).ySize - 1; j++)

{

for (int k = 1; k < Map.GetMap(i).xSize - 1; k++)

{

if (area[k, j] == 1)

{

area[k, j] = 2;

ConvertNearbySquares(ref area, k, j);

IslandCount++;

int Twos = CountTwos(area, Map.GetMap(i).xSize, Map.GetMap(i).ySize);

if (MaxTwos < Twos)

{

MaxTwos = Twos;

}

ClearTwos(ref area, Map.GetMap(i).xSize, Map.GetMap(i).ySize);

}

}

}

results[i, 0] = IslandCount;

results[i, 1] = MaxTwos;

}

}

/// <summary>

/// Metodas su rekursija. Suranda visus langelį liečiančius langelius, ir tuos langelius liečiančius langelius ir t.t.

/// </summary>

/// <param name="area"> perduodamas žemėlapis, kuriame 0 - vanduo, 1 - dar netikrinta žemė, 2 - jau patikrinta žemė</param>

/// <param name="xpos"> langelio x koordinate </param>

/// <param name="ypos"> langelio y koordinate </param>

private static void ConvertNearbySquares(ref int[,] area, int xpos, int ypos)

{

area[xpos, ypos] = 2;

for (int y = -1; y <= 1; y++)

{

for (int x = -1; x <= 1; x++)

{

if (area[xpos + x, ypos + y] == 1)

{

ConvertNearbySquares(ref area, xpos + x, ypos + y);

}

}

}

}

/// <summary>

/// Kad būtų galima skaičiuot kitos salos dydį, ši sala panaikinama iš žemėlapio masyvo (konteineris išlieka nepakeistas)

/// </summary>

/// <param name="Map"> Žemėlapio masyvas</param>

/// <param name="x"> Žemėlapio didžiausia x koordinate </param>

/// <param name="y"> Žemėlapio didžiausia y koordinate </param>

private static void ClearTwos(ref int[,] Map, int x, int y)

{

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

{

for (int j = 0; j < x; j++)

{

if (Map[j, i] == 2)

Map[j, i] = 0;

}

}

}

/// <summary>

/// Suskaičiuoja visus dvejetus dvimatyje masyve, t.y. suskaičiuoja salos dydį

/// </summary>

/// <param name="Map"> žemėlapio dvimatis masyvas </param>

/// <param name="x"> žemėlapio didžiausia x koordinatė </param>

/// <param name="y"> žemėlapio didžiausia y koordinatė </param>

/// <returns></returns>

private static int CountTwos(int[,] Map, int x, int y)

{

int count = 0;

{

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

{

for (int j = 0; j < x; j++)

{

if (Map[j, i] == 2)

count++;

}

}

}

return count;

}

/// <summary>

/// Paspaudus Button1 atlieka skaičiavimus ir spausdinimą

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

protected void Button1\_Click(object sender, EventArgs e)

{

int Count;

string path = Server.MapPath(data);

int[,] results = Islands(out Count);

Print(results, Count);

}

/// <summary>

/// spausdina duomenis ekrane

/// </summary>

private void PrintDataWeb()

{

string[] lines = File.ReadAllLines(Server.MapPath(data));

foreach (string line in lines)

{

TableRow row = new TableRow();

TableCell Line = new TableCell();

Line.Text = line;

row.Cells.Add(Line);

Table2.Rows.Add(row);

}

}

/// <summary>

/// spausdina duomenis faile

/// </summary>

/// <param name="Map"></param>

private void PrintDataFile(MapsContainer Map)

{

string path = Server.MapPath("App\_Data/Duomenys.txt");

using (StreamWriter writer = new StreamWriter(path))

{

writer.WriteLine("Salų kiekis: {0,10}", Map.Count);

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

{

writer.WriteLine("n ir m reikšmės: {0,5}{1,5}", Map.GetMap(i).ySize, Map.GetMap(i).xSize);

for (int j = 1; j < Map.GetMap(i).ySize-1; j++)

{

writer.Write("Stačiakampio duomenys: "); //24

for (int k = 1; k < Map.GetMap(i).xSize-1; k++)

{

writer.Write(Map.GetMap(i).Map[k, j]);

}

writer.WriteLine();

}

}

}

}

## Pradiniai duomenys ir rezultatai

**I variantas.**

|  |
| --- |
| U3.txt |
| 4  2 6  110100  000000  9 4  1000  1000  0110  1100  1100  0000  1010  1000  0000  10 7  0100110  0111000  0010110  1000000  1011100  1001110  0011000  1000100  0000100  0000000  2 2  00  01 |

|  |
| --- |
| Duomenys.txt |
| Salų kiekis: 4  n ir m reikšmės: 4 8  Stačiakampio duomenys: 110100  Stačiakampio duomenys: 000000  n ir m reikšmės: 11 6  Stačiakampio duomenys: 1000  Stačiakampio duomenys: 1000  Stačiakampio duomenys: 0110  Stačiakampio duomenys: 1100  Stačiakampio duomenys: 1100  Stačiakampio duomenys: 0000  Stačiakampio duomenys: 1010  Stačiakampio duomenys: 1000  Stačiakampio duomenys: 0000  n ir m reikšmės: 12 9  Stačiakampio duomenys: 0100110  Stačiakampio duomenys: 0111000  Stačiakampio duomenys: 0010110  Stačiakampio duomenys: 1000000  Stačiakampio duomenys: 1011100  Stačiakampio duomenys: 1001110  Stačiakampio duomenys: 0011000  Stačiakampio duomenys: 1000100  Stačiakampio duomenys: 0000100  Stačiakampio duomenys: 0000000  n ir m reikšmės: 4 4  Stačiakampio duomenys: 00  Stačiakampio duomenys: 00 |

|  |
| --- |
| Rezultatai.txt |
| Salų skaičius Didž. sala  ------------------------------  2 2  3 8  4 10  0 0 |

|  |
| --- |
| Vartotojo sąsaja |
| https://i.gyazo.com/9db3f28ab402c3557011f35f1151a758.png |

**II variantas.**

|  |
| --- |
| U3.txt |
| 1  8 8  00111111  01001111  00000000  01111110  00000001  01111110  01111110  00100000 |

|  |
| --- |
| Duomenys.txt |
| Salų kiekis: 1  n ir m reikšmės: 10 10  Stačiakampio duomenys: 00111111  Stačiakampio duomenys: 01001111  Stačiakampio duomenys: 00000000  Stačiakampio duomenys: 01111110  Stačiakampio duomenys: 00000001  Stačiakampio duomenys: 01111110  Stačiakampio duomenys: 01111110  Stačiakampio duomenys: 00100000 |

|  |
| --- |
| Rezultatai.txt |
| Salų skaičius Didž. sala  ------------------------------  2 20 |

|  |
| --- |
| Vartotojo sąsaja |
| https://i.gyazo.com/716d7f7c903ef89cb520fc6298608386.png |

## Dėstytojo pastabos

# Dinaminis atminties valdymas (L2)

## Darbo užduotis

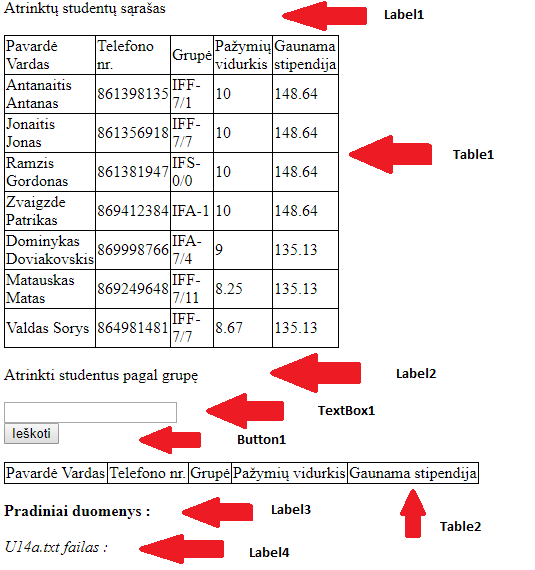
LD\_14. Stipendijos. Studentų stipendijoms yra skiriamas nurodyto dydžio fondas. Studentui skiriama stipendija, jei jo pažymių vidurkis viršija nurodytą dydį ir jis neturi skolų (visi pažymiai >4). Studentui skiriama 10% didesnė stipendija, jei jo visi pažymiai didesni už 8. Toks studentas vadinamas pirmūnu. Paskirstykite studentams stipendijas pagal duotą fondą. Fondą reikia maksimaliai išnaudoti, bet negalima viršyti fondo dydžio.

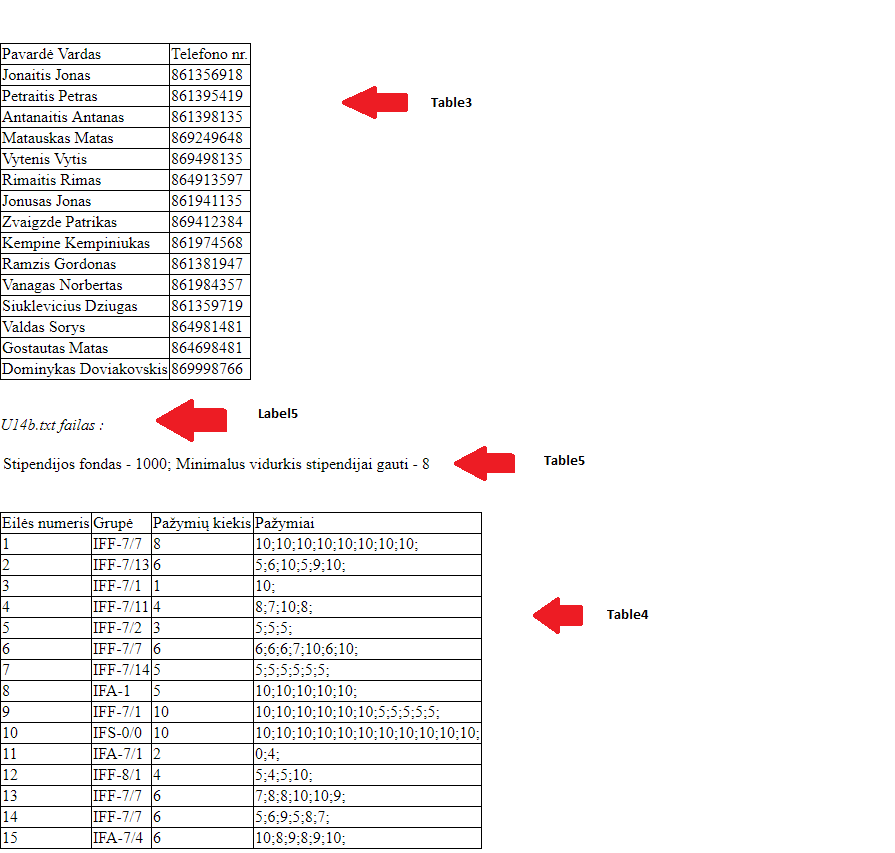
Duomenys:

* Tekstiniame faile U14a.txt duoti studentų asmeniniai duomenys: pavardė vardas, telefono numeris.
* Tekstiniame faile U14b.txt pateikta studentų tarnybinė informacija. Pirmojoje failo eilutėje nurodytas stipendijų fondo dydis ir pažymių vidurkis stipendijai gauti. Tolimesnėse eilutėse tokia informacija: studento eilės numeris, kuris sutampa su eilės numeriu U14a.txt faile, grupė, pažymių kiekis, pažymiai .

Spausdinamas sąrašas turi būti surikiuotas pagal stipendijų dydį, pavardes ir vardus. Iš sąrašo pašalinkite studentus, kurie negauna stipendijos. Suformuokite ir atspausdinkite nurodytos grupės (įvedama klaviatūra) pirmumų sąrašą.

## Grafinės vartotojo sąsajos schema

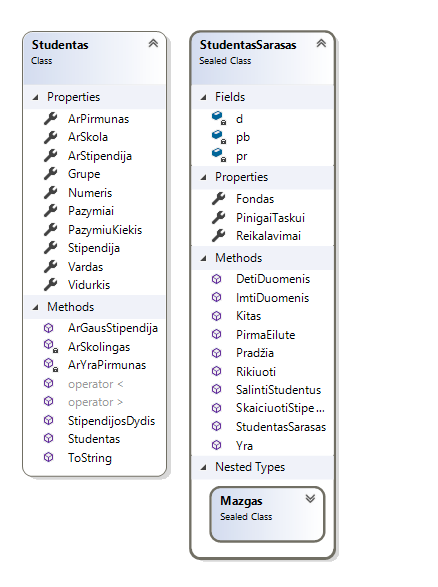




## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Label1 | Text | Atrinktų studentų sąrašas |
| Table1, Table2, Table3, Table4 | BorderColor  BorderStyle  BorderWidth  GridLines | Black  Solid  1px  Both |
| TextBox1 |  |  |
| Label2 | Text | Atrinkti studentus pagal grupę |
| Button1 | Text | Ieškoti |
| Label3 | Text | <b> Pradiniai duomenys :</b> |
| Label4 | CssClass  Text | newStyle1  U14a.txt failas |
| Label5 | CssClass  Text | newStyle2  U14b.txt failas : |
| Table5 |  |  |
| Image1 | CssClass  ImageUrl | newStyle3  https://media.giphy.com/media/X8omQqfFyeq1a/giphy.gif |

## Klasių diagrama



## Programos vartotojo vadovas

* U14a.txt duoti studentų asmeniniai duomenys: pavardė vardas, telefono numeris.
* Tekstiniame faile U14b.txt pateikta studentų tarnybinė informacija. Pirmojoje failo eilutėje nurodytas stipendijų fondo dydis ir pažymių vidurkis stipendijai gauti. Tolimesnėse eilutėse tokia informacija: studento eilės numeris, kuris sutampa su eilės numeriu U14a.txt faile, grupė, pažymių kiekis, pažymiai .

U14a.txt surašome studentų asmeniniai duomenys: pavardė vardas, telefono numeris. Tekstiniame faile U14b.txt pateikiamem studentų tarnybinė informacija. Pirmojoje failo eilutėje nurodytas stipendijų fondo dydis ir pažymių vidurkis stipendijai gauti. Tolimesnėse eilutėse tokia informacija: studento eilės numeris, kuris sutampa su eilės numeriu U14a.txt faile, grupė, pažymių kiekis, pažymiai. Duomenys turi būti atskirti simboliu “;”.

Įjungę programą po užrašu „Atrinktų studentų sąrašas“ matome duomenis studentų, kurie gaus stipendiją. Taip pat galima atsirinkti studentus pagal grupių pavadinimus, įvedus grupės pavadinimą po užrašu „Atrinkti studentus pagal grupę“ ir paspaudus mygtuką „Ieškoti“. Nurodytos grupės studentai bus spausdinami lentele po mygtuku „Ieškoti“. Lentelėse po užrašu „**Pradiniai duomenys :**“ yra spausdinami pradiniai duomenys.

## Programos tekstas

Studentas.cs klasė

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

/// <summary>

/// Summary description for Studentas

/// </summary>

public class Studentas

{

/// <summary>

/// konstruktorius

/// </summary>

/// <param name="vardas"> vardas ir pavardė</param>

/// <param name="numeris"> telefono numeris</param>

/// <param name="group"> grupes pavadinimas</param>

/// <param name="gradecount"> pažymių kiekis</param>

/// <param name="grades"> pažymių masyvas</param>

/// <param name="reikalavimas"> minimalus pažimys stipendijai</param>

public Studentas(string vardas, string numeris, string grupe, int pazymiuKiekis, int[] pazymiai, double reikalavimas)

{

Vardas = vardas;

Numeris = numeris;

Grupe = grupe;

PazymiuKiekis = pazymiuKiekis;

int[] tuscias = new int[PazymiuKiekis];

Pazymiai = tuscias;

Pazymiai = pazymiai;

ArPirmunas = ArYraPirmunas();

ArSkola = ArSkolingas();

ArGausStipendija(reikalavimas);

}

/// <summary>

/// Savybės

/// </summary>

public string Vardas { get; set; }

public string Numeris { get; set; }

public string Grupe { get; set; }

public int PazymiuKiekis { get; set; }

public int[] Pazymiai { get; set; }

/// <summary>

/// ar pirmūnas

/// </summary>

public bool ArPirmunas { get; set; }

/// <summary>

/// ar turi skolų

/// </summary>

public bool ArSkola { get; set; }

/// <summary>

/// gaunamos stipendijos dydis

/// </summary>

public double Stipendija { get; set; }

/// <summary>

/// ar gauna stipendija

/// </summary>

public bool ArStipendija { get; set; }

/// <summary>

/// Pažymių vidurkis

/// </summary>

public double Vidurkis { get; set; }

/// <summary>

/// Suranda ar studentas gaus stipendija

/// </summary>

/// <param name="reikalavimas"> minimalus pažimys stipendijai</param>

/// <returns></returns>

public bool ArGausStipendija(double reikalavimas)

{

double vidurkis = 0;

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

vidurkis = vidurkis + Pazymiai[i];

vidurkis = vidurkis / (PazymiuKiekis);

Vidurkis = Math.Round(vidurkis, 2);

if (vidurkis > reikalavimas)

{

ArStipendija = true;

return true;

}

else

{

ArStipendija = false;

return false;

}

}

/// <summary>

/// Suranda ar pirmūnas

/// </summary>

/// <returns></returns>

private bool ArYraPirmunas()

{

bool top = true;

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

{

if (Pazymiai[i] < 9)

top = false;

}

return top;

}

/// <summary>

/// patikrina ar studentas turi skolų

/// </summary>

/// <returns></returns>

private bool ArSkolingas()

{

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

{

if (Pazymiai[i] < 5)

{

return true;

}

}

return false;

}

/// <summary>

/// tikrina pagal stipendijos dydi ir vardą pavardė abėcėlės tvarka

/// </summary>

/// <param name="pirmas"> </param>

/// <param name="antras"></param>

/// <returns></returns>

static public bool operator >(Studentas pirmas, Studentas antras)

{

if (pirmas.Stipendija > antras.Stipendija)

return true;

else if (antras.Stipendija > pirmas.Stipendija)

return false;

else

{

int a = String.Compare(pirmas.Vardas, antras.Vardas, StringComparison.CurrentCulture);

return a < 0;

}

}

static public bool operator <(Studentas pirmas,

Studentas antras)

{

return !(pirmas > antras);

}

public override string ToString()

{

string eilute = string.Format("{0,-30};{1,-12};{2,-15};{3,-10};{4, 20}", Vardas, Numeris, Grupe, Vidurkis, Stipendija);

return eilute;

}

/// <summary>

/// Suskaičiuoja kokią stipendiją gaus studentas

/// </summary>

/// <param name="PinigaiTaskui"></param>

public void StipendijosDydis(double PinigaiTaskui)

{

int taskai = 0;

if (ArStipendija)

{

taskai = 10;

if (ArPirmunas)

taskai = 11;

}

Stipendija = Math.Floor(PinigaiTaskui \* taskai \* 100) / 100;

}

}

StudentasSarasas.cs sąrašo klasė

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

/// <summary>

/// Summary description for StudentasSarasas

/// </summary>

public sealed class StudentasSarasas

{

/// <summary>

/// vieno elemento klasė

/// </summary>

private sealed class Mazgas

{

public Studentas Duom { get; set; }

public Mazgas Kitas { get; set; }

public Mazgas(Studentas a, Mazgas adr)

{

Duom = a;

Kitas = adr;

}

}

private Mazgas pr; // sąrašo pradžią

private Mazgas pb; // sąrašo pabaiga

private Mazgas d; // sąrašo sąsaja

public double Fondas { get; private set; } // Stipendijos fondas

public double Reikalavimai { get; private set; } // Minimalus Pažimys stipendijai

/// <summary>

/// Įveda į studentų sąrašą // Stipendijos fondą ir Minimalų Pažimį stipendijai

/// </summary>

/// <param name="fondas"> Stipendijos fondas</param>

/// <param name="reikalavimai"> Minimalus Pažimys stipendijai</param>

public void PirmaEilute(double fondas, double reikalavimai)

{

Fondas = fondas;

Reikalavimai = reikalavimai;

}

/// <summary>

/// konstruktorius

/// </summary>

public StudentasSarasas()

{

pr = null;

pb = null;

d = null;

}

/// <summary>

/// Įdeda duomenis atvirkštine tvarka

/// </summary>

/// <param name="naujas"> studento duomenys</param>

public void DetiDuomenis(Studentas naujas)

{

pr = new Mazgas(naujas, pr);

}

/\*\* Sąsajai priskiriama sąrašo pradžia \*/

public void Pradžia()

{ d = pr; }

/\*\* Sąsajai priskiriamas tolesnis sąrašo elementas \*/

public void Kitas()

{ d = d.Kitas; }

/\*\* Grąžina true, jeigu sąrašas netuščias \*/

public bool Yra()

{ return d != null; }

/\*\* Grąžina sąrašo sąsajos elemento reikšmę \*/

public Studentas ImtiDuomenis()

{ return d.Duom; }

/// <summary>

/// rikiavimas pagal stipendijos dydį ir vardą pavardę

/// </summary>

public void Rikiuoti()

{

for (Mazgas d1 = pr; d1 != null; d1 = d1.Kitas)

{

Mazgas minv = d1;

for (Mazgas d2 = d1.Kitas; d2 != null; d2 = d2.Kitas)

if (d2.Duom > minv.Duom)

minv = d2;

// Informacinių dalių sukeitimas vietomis

Studentas St = d1.Duom;

d1.Duom = minv.Duom;

minv.Duom = St;

}

}

/// <summary>

/// Šalina studentus kurie negaus stipendijos

/// </summary>

public void SalintiStudentus()

{

for (Mazgas d1 = pr; d1 != null; /\*d1 = d1.Kitas\*/)

{

d1.Duom.StipendijosDydis(PinigaiTaskui);

if (d1.Kitas != null)

if (d1.Kitas.Duom.ArSkola|| !d1.Kitas.Duom.ArStipendija)

{

d1.Kitas = d1.Kitas.Kitas;

}

else

d1 = d1.Kitas;

else

d1 = d1.Kitas;

}

if (pr != null)

if (pr.Duom.ArSkola || !pr.Duom.ArStipendija)

pr = pr.Kitas;

}

public double PinigaiTaskui { get; private set; } // 10% stipendijos

/// <summary>

/// Suskaičiuoja kiek yra 10% stipendijos

/// </summary>

public void SkaiciuotiStipendijosTaskus()

{

int Taskai = 0;

for (Mazgas d1 = pr; d1 != null; d1 = d1.Kitas)

{

if (d1.Duom.ArGausStipendija(Reikalavimai))

{

Taskai = Taskai + 10;

if (d1.Duom.ArPirmunas)

Taskai++;

}

}

PinigaiTaskui = Fondas / Taskai;

}

}

Labaratorinis2.aspx

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Labaratorinis2.aspx.cs" Inherits="Labaratorinis2" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

<style type="text/css">

.newStyle1 {

font-style: italic;

}

.newStyle2 {

font-style: italic;

}

.newStyle3 {

position: fixed;

top: 43px;

left: 660px;

height: 343px;

width: 399px;

}

</style>

</head>

<body>

<form id="form1" runat="server">

<asp:Label ID="Label1" runat="server" Text=" Atrinktų studentų sąrašas"></asp:Label>

<br />

<br />

<asp:Table ID="Table1" runat="server" BorderColor="Black" BorderStyle="Solid" BorderWidth="1px" GridLines="Both" style="width: 30px">

</asp:Table>

<br />

<asp:Label ID="Label2" runat="server" Text=" Atrinkti studentus pagal grupę"></asp:Label>

<asp:Image ID="Image1" runat="server" CssClass="newStyle3" ImageUrl="https://media.giphy.com/media/X8omQqfFyeq1a/giphy.gif" />

<br />

<br />

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

<br />

<asp:Button ID="Button1" runat="server" OnClick="Button1\_Click" Text="Ieškoti" />

<br />

<br />

<asp:Table ID="Table2" runat="server" BorderColor="Black" BorderStyle="Solid" BorderWidth="1px" GridLines="Both">

</asp:Table>

<br />

<asp:Label ID="Label3" runat="server" Text="<b> Pradiniai duomenys :</b>"></asp:Label>

<br />

<br />

<asp:Label ID="Label4" runat="server" Text="U14a.txt failas :" CssClass="newStyle1"></asp:Label>

<br />

<br />

<asp:Table ID="Table3" runat="server" BorderColor="Black" BorderStyle="Solid" BorderWidth="1px" GridLines="Both">

</asp:Table>

<br />

<br />

<asp:Label ID="Label5" runat="server" Text="U14b.txt failas :" CssClass="newStyle2"></asp:Label>

<br />

<br />

<asp:Table ID="Table5" runat="server">

</asp:Table>

<br />

<br />

<asp:Table ID="Table4" runat="server" BorderColor="Black" BorderStyle="Solid" BorderWidth="1px" GridLines="Both">

</asp:Table>

</form>

</body>

</html>

Labaratorinis2.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.IO;

public partial class Labaratorinis2 : System.Web.UI.Page

{

//duomenų failai

public const string stduoma = "App\_Data/U14a.txt";

public const string stduomb = "App\_Data/U14b.txt";

//public const string stduoma = "App\_Data/U14aa.txt";

//public const string stduomb = "App\_Data/U14ba.txt";

protected void Page\_Load(object sender, EventArgs e)

{

StudentasSarasas Sarasas = skaitymas();

Sarasas.SkaiciuotiStipendijosTaskus();

Sarasas.SalintiStudentus();

Sarasas.Rikiuoti();

Spausdinti(Sarasas, Table1);

SpausdintiPradiniusDuomenis(Table4, Table3, Table5);

string AtrinktiGrupe = TextBox1.Text;

SpausdintiAtrinkta(Sarasas, Table2, AtrinktiGrupe);

SpausdintiRezultatus(Sarasas);

SpausdintiDuomenis();

}

/// <summary>

/// Skaitomo duomenis

/// </summary>

/// <returns> Gražina studentų sąrašą</returns>

private StudentasSarasas skaitymas()

{

StudentasSarasas sarasas = new StudentasSarasas();

string path = Server.MapPath(stduomb);

using (StreamReader reader1 = new StreamReader(@path))

{

string[] duomenys = reader1.ReadLine().Split(';');

sarasas.PirmaEilute(double.Parse(duomenys[0]), double.Parse(duomenys[1]));

string kitaspath = Server.MapPath(stduoma);

string[] eilutes = File.ReadAllLines(kitaspath);

foreach (string eilute in eilutes)

{

string[] StudentoDuomenys = reader1.ReadLine().Split(';');

string[] eiluteA = eilute.Split(';');

int[] pazymiai = new int[int.Parse(StudentoDuomenys[2])];

int j = 0;

for (int i = 3; i < 3 + int.Parse(StudentoDuomenys[2]); i++)

{

pazymiai[j++] = int.Parse(StudentoDuomenys[i]);

}

Studentas studentas = new Studentas(eiluteA[0], eiluteA[1], StudentoDuomenys[1], int.Parse(StudentoDuomenys[2]), pazymiai, sarasas.Reikalavimai);

sarasas.DetiDuomenis(studentas);

}

}

return sarasas;

}

/// <summary>

/// Spausdina duomenis į lentelę

/// </summary>

/// <param name="A"> Perduodamas studentų sąrašas</param>

/// <param name="lentele"> Į kurią lentelę vesti</param>

static void Spausdinti(StudentasSarasas A, Table lentele)

{

// Sąrašo peržiūra, panaudojant sąsajos metodus

LentelesAntraste(lentele);

for (A.Pradžia(); A.Yra(); A.Kitas())

{

IterptiIrasa(A.ImtiDuomenis(), lentele);

}

}

/// <summary>

/// Spausdina nurodytls grupes studentus

/// </summary>

/// <param name="A"> Studentų sąrašus</param>

/// <param name="lentele"> į kurią lentelę vesti</param>

/// <param name="grupe"> grupės pavadinimas</param>

static void SpausdintiAtrinkta(StudentasSarasas A, Table lentele, string grupe)

{

// Sąrašo peržiūra, panaudojant sąsajos metodus

LentelesAntraste(lentele);

for (A.Pradžia(); A.Yra(); A.Kitas())

{

if (A.ImtiDuomenis().Grupe == grupe)

{

IterptiIrasa(A.ImtiDuomenis(), lentele);

}

}

}

protected void Button1\_Click(object sender, EventArgs e)

{

}

/// <summary>

/// Išveda studentą į lentelę

/// </summary>

/// <param name="a"> studentų sąrašas</param>

/// <param name="lentele"> į kurią lentelę vesti</param>

static void IterptiIrasa(Studentas a, Table lentele)

{

TableRow eilute = new TableRow();

string[] Duomenys = a.ToString().Split(';');

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

{

eilute.Cells.Add(Elementas(Duomenys[i]));

}

lentele.Rows.Add(eilute);

}

/// <summary>

/// sukuriamas lenteles langelis

/// </summary>

/// <param name="tekstas"> langelio tekstas</param>

/// <returns> grtažina langelį</returns>

static TableCell Elementas(string tekstas)

{

TableCell elementas = new TableCell();

elementas.Text = tekstas;

return elementas;

}

/// <summary>

/// Sukuriama rezultatų lentelės antraštė

/// </summary>

/// <param name="lentele"> į kurią lentelę vesti</param>

static void LentelesAntraste(Table lentele)

{

TableRow Eilute = new TableRow();

Eilute.Cells.Add(Elementas("Pavardė Vardas"));

Eilute.Cells.Add(Elementas("Telefono nr."));

Eilute.Cells.Add(Elementas("Grupė"));

Eilute.Cells.Add(Elementas("Pažymių vidurkis"));

Eilute.Cells.Add(Elementas("Gaunama stipendija"));

lentele.Rows.Add(Eilute);

}

/// <summary>

/// Spausdina pradinius duomenis į lenteles

/// </summary>

/// <param name="table1"> į kurią lentelę vesti</param>

/// <param name="table2"> į kurią lentelę vesti</param>

void SpausdintiPradiniusDuomenis(Table table1, Table table2, Table table3)

{

string path = Server.MapPath(stduomb);

using (StreamReader reader = new StreamReader(path))

{

string[] eilute = reader.ReadLine().Split(';');

TableRow Eilute = new TableRow();

Eilute.Cells.Add(Elementas(String.Format("{0}{1};", "Stipendijos fondas - ", eilute[0])));

Eilute.Cells.Add(Elementas(String.Format("{0}{1}", "Minimalus vidurkis stipendijai gauti - ", eilute[1])));

table3.Rows.Add(Eilute);

AntrosPradinesLentelesAntraste(table1);

string line = reader.ReadLine();

while (line != null)

{

string[] duomenys = line.Split(';');

TableRow LentelesEilute = new TableRow();

for (int i = 0; i < 3; i++)

{

LentelesEilute.Cells.Add(Elementas(duomenys[i]));

}

string Pazymiai = line.Substring(1 + line.IndexOf(String.Format(";{0};", duomenys[3])));

LentelesEilute.Cells.Add(Elementas(Pazymiai));

table1.Rows.Add(LentelesEilute);

line = reader.ReadLine();

}

}

string kitaspath = Server.MapPath(stduoma);

string[] lines = File.ReadAllLines(kitaspath);

PirmosPradinesLentelesAntraste(table2);

foreach (string line in lines)

{

TableRow KitaLentelesEilute = new TableRow();

string[] KitiDuomenys = line.Split(';');

KitaLentelesEilute.Cells.Add(Elementas(KitiDuomenys[0]));

KitaLentelesEilute.Cells.Add(Elementas(KitiDuomenys[1]));

table2.Rows.Add(KitaLentelesEilute);

}

}

/// <summary>

/// Antros pradinės lentelės antraštė

/// </summary>

/// <param name="lentele"> į kurią lentelę vesti</param>

static void AntrosPradinesLentelesAntraste(Table lentele)

{

TableRow Eilute = new TableRow();

Eilute.Cells.Add(Elementas("Eilės numeris"));

Eilute.Cells.Add(Elementas("Grupė"));

Eilute.Cells.Add(Elementas("Pažymių kiekis"));

Eilute.Cells.Add(Elementas("Pažymiai"));

lentele.Rows.Add(Eilute);

}

/// <summary>

/// Pirmos pradinės lentelės antraštė

/// </summary>

/// <param name="lentele"> į kurią lentelę vesti</param>

static void PirmosPradinesLentelesAntraste(Table lentele)

{

TableRow Eilute = new TableRow();

Eilute.Cells.Add(Elementas("Pavardė Vardas"));

Eilute.Cells.Add(Elementas("Telefono nr."));

lentele.Rows.Add(Eilute);

}

/// <summary>

/// Spausdina rezultatus

/// </summary>

/// <param name="A"> Studentų sąrašas</param>

void SpausdintiRezultatus(StudentasSarasas A)

{

string path = Server.MapPath("App\_Data/Rezultatai.txt");

using (StreamWriter writer = new StreamWriter(path))

{

writer.WriteLine("----------------------------------------------------------------------------------------------------------");

writer.WriteLine("{0,-35}{1,-16}{2,-15}{3,-20}{4,-30}", "Pavardė Vardas", "Telefono nr.", "Grupė", "Pažymių vidurkis", "Gaunama stipendija");

writer.WriteLine("----------------------------------------------------------------------------------------------------------");

for (A.Pradžia(); A.Yra(); A.Kitas())

{

string[] duomenys = A.ImtiDuomenis().ToString().Split(';');

writer.WriteLine("{0,-35}{1,-16}{2,-15}{3,-20}{4,-30}", duomenys[0], duomenys[1], duomenys[2], duomenys[3], duomenys[4]);

}

writer.WriteLine("----------------------------------------------------------------------------------------------------------");

}

}

/// <summary>

/// spausdinami duomenys

/// </summary>

void SpausdintiDuomenis()

{

string path = Server.MapPath("App\_Data/Duomenys.txt");

using (StreamWriter writer = new StreamWriter(path))

{

string path1 = Server.MapPath(stduoma);

string path2 = Server.MapPath(stduomb);

using (StreamReader reader = new StreamReader(path2))

{

writer.WriteLine("U14a.txt :");

writer.WriteLine();

writer.WriteLine("--------------------------------------------------");

writer.WriteLine("{0,-30}{1,20}", "Pavardė Vardas", "Telefono nr.");

writer.WriteLine("--------------------------------------------------");

string[] eilutes = File.ReadAllLines(path1);

foreach (string eilute in eilutes)

{

string[] duomenys = eilute.Split(';');

writer.WriteLine("{0,-30}{1,20}", duomenys[0], duomenys[1]);

}

writer.WriteLine("--------------------------------------------------");

string Eilute = reader.ReadLine();

string[] Duomenys = Eilute.Split(';');

writer.WriteLine();

writer.WriteLine("U14b.txt :");

writer.WriteLine();

writer.WriteLine("---------------------------------------------------------------------------------------------");

writer.WriteLine("Fondas stipendijai: {0} Reikalavimas Stipendijai: {1}", Duomenys[0], Duomenys[1]);

writer.WriteLine("---------------------------------------------------------------------------------------------");

writer.WriteLine("{0,-20}{1,-15}{2,-15}{3,-20}", "Eilės numeris", "Grupė", "Pažymių kiekis", "Pažymiai");

writer.WriteLine("---------------------------------------------------------------------------------------------");

Eilute = reader.ReadLine();

while (Eilute != null)

{

string[] duomenys = Eilute.Split(';');

string Pazymiai = Eilute.Substring(1 + Eilute.IndexOf(String.Format(";{0};", duomenys[3])));

writer.WriteLine("{0,-20}{1,-15}{2,-15}{3,-20}", duomenys[0], duomenys[1], duomenys[2], Pazymiai);

Eilute = reader.ReadLine();

}

writer.WriteLine("---------------------------------------------------------------------------------------------");

}

}

}

}

## Pradiniai duomenys ir rezultatai

**I variantas**

|  |
| --- |
| U14a.txt |
| Jonaitis Jonas;861356918;  Petraitis Petras;861395419;  Antanaitis Antanas;861398135;  Matauskas Matas;869249648;  Vytenis Vytis;869498135;  Rimaitis Rimas;864913597;  Jonusas Jonas;861941135;  Zvaigzde Patrikas;869412384;  Kempine Kempiniukas;861974568;  Ramzis Gordonas;861381947;  Vanagas Norbertas;861984357;  Siuklevicius Dziugas;861359719;  Valdas Sorys;864981481;  Gostautas Matas;864698481;  Dominykas Doviakovskis;869998766; |

|  |
| --- |
| U14b.txt |
| 1000;8  1;IFF-7/7;8;10;10;10;10;10;10;10;10;  2;IFF-7/13;6;5;6;10;5;9;10;  3;IFF-7/1;1;10;  4;IFF-7/11;4;8;7;10;8;  5;IFF-7/2;3;5;5;5;  6;IFF-7/7;6;6;6;7;10;6;10;  7;IFF-7/14;5;5;5;5;5;5;  8;IFA-1;5;10;10;10;10;10;  9;IFF-7/1;10;10;10;10;10;10;5;5;5;5;5;  10;IFS-0/0;10;10;10;10;10;10;10;10;10;10;10;  11;IFA-7/1;2;0;4;  12;IFF-8/1;4;5;4;5;10;  13;IFF-7/7;6;7;8;8;10;10;9;  14;IFF-7/7;6;5;6;9;5;8;7;  15;IFA-7/4;6;10;8;9;8;9;10; |

|  |
| --- |
| Duomenys.txt |
| U14a.txt :  --------------------------------------------------  Pavardė Vardas Telefono nr.  --------------------------------------------------  Jonaitis Jonas 861356918  Petraitis Petras 861395419  Antanaitis Antanas 861398135  Matauskas Matas 869249648  Vytenis Vytis 869498135  Rimaitis Rimas 864913597  Jonusas Jonas 861941135  Zvaigzde Patrikas 869412384  Kempine Kempiniukas 861974568  Ramzis Gordonas 861381947  Vanagas Norbertas 861984357  Siuklevicius Dziugas 861359719  Valdas Sorys 864981481  Gostautas Matas 864698481  Dominykas Doviakovskis 869998766  --------------------------------------------------  U14b.txt :  ---------------------------------------------------------------------------------------------  Fondas stipendijai: 1000 Reikalavimas Stipendijai: 8  ---------------------------------------------------------------------------------------------  Eilės numeris Grupė Pažymių kiekis Pažymiai  ---------------------------------------------------------------------------------------------  1 IFF-7/7 8 10;10;10;10;10;10;10;10;  2 IFF-7/13 6 5;6;10;5;9;10;  3 IFF-7/1 1 10;  4 IFF-7/11 4 8;7;10;8;  5 IFF-7/2 3 5;5;5;  6 IFF-7/7 6 6;6;6;7;10;6;10;  7 IFF-7/14 5 5;5;5;5;5;5;  8 IFA-1 5 10;10;10;10;10;  9 IFF-7/1 10 10;10;10;10;10;10;5;5;5;5;5;  10 IFS-0/0 10 10;10;10;10;10;10;10;10;10;10;10;  11 IFA-7/1 2 0;4;  12 IFF-8/1 4 5;4;5;10;  13 IFF-7/7 6 7;8;8;10;10;9;  14 IFF-7/7 6 5;6;9;5;8;7;  15 IFA-7/4 6 10;8;9;8;9;10;  --------------------------------------------------------------------------------------------- |

|  |
| --- |
| Rezultatai.txt |
| ----------------------------------------------------------------------------------------------------------  Pavardė Vardas Telefono nr. Grupė Pažymių vidurkis Gaunama stipendija  ----------------------------------------------------------------------------------------------------------  Antanaitis Antanas 861398135 IFF-7/1 10 148.64  Jonaitis Jonas 861356918 IFF-7/7 10 148.64  Ramzis Gordonas 861381947 IFS-0/0 10 148.64  Zvaigzde Patrikas 869412384 IFA-1 10 148.64  Dominykas Doviakovskis 869998766 IFA-7/4 9 135.13  Matauskas Matas 869249648 IFF-7/11 8.25 135.13  Valdas Sorys 864981481 IFF-7/7 8.67 135.13  ---------------------------------------------------------------------------------------------------------- |

|  |
| --- |
| Vartotojo sąsaja |
| https://i.gyazo.com/94434249a801902dd8a58379f58c634f.png |

|  |
| --- |
| Vartotojo sąsaja atrinkus „IFF-7/7“ grupės studentus |
| https://i.gyazo.com/dc5a2ecbf7b009a8fcfe6faf4b8f521a.png |

**II variantas**

Pakeisti studentų pažymiai taip, kad liktu tik vienas pirmūnas ir vienas ne pirmūnas.

|  |
| --- |
| U14a.txt |
| Jonaitis Jonas;861356918;  Petraitis Petras;861395419;  Antanaitis Antanas;861398135;  Matauskas Matas;869249648;  Vytenis Vytis;869498135;  Rimaitis Rimas;864913597;  Jonusas Jonas;861941135;  Zvaigzde Patrikas;869412384;  Kempine Kempiniukas;861974568;  Ramzis Gordonas;861381947;  Vanagas Norbertas;861984357;  Siuklevicius Dziugas;861359719;  Valdas Sorys;864981481;  Gostautas Matas;864698481;  Dominykas Doviakovskis;869998766; |

|  |
| --- |
| U14b.txt |
| 1000;9  1;IFF-7/7;8;10;10;10;10;10;10;10;10;  2;IFF-7/13;6;5;6;10;5;9;10;  3;IFF-7/1;1;8;  4;IFF-7/11;4;8;7;10;8;  5;IFF-7/2;3;5;5;5;  6;IFF-7/7;6;6;6;7;4;6;10;  7;IFF-7/14;5;5;5;5;5;5;  8;IFA-1;5;10;10;10;10;4;  9;IFF-7/1;10;10;10;10;10;10;5;5;5;5;5;  10;IFS-0/0;10;10;10;10;10;10;10;10;10;10;8;  11;IFA-7/1;2;0;4;  12;IFF-8/1;4;5;4;5;10;  13;IFF-7/7;6;7;8;8;4;10;9;  14;IFF-7/7;6;5;6;9;5;8;7;  15;IFA-7/4;6;10;8;9;8;9;4; |

|  |
| --- |
| Duomenys.txt |
| U14a.txt :  --------------------------------------------------  Pavardė Vardas Telefono nr.  --------------------------------------------------  Jonaitis Jonas 861356918  Petraitis Petras 861395419  Antanaitis Antanas 861398135  Matauskas Matas 869249648  Vytenis Vytis 869498135  Rimaitis Rimas 864913597  Jonusas Jonas 861941135  Zvaigzde Patrikas 869412384  Kempine Kempiniukas 861974568  Ramzis Gordonas 861381947  Vanagas Norbertas 861984357  Siuklevicius Dziugas 861359719  Valdas Sorys 864981481  Gostautas Matas 864698481  Dominykas Doviakovskis 869998766  --------------------------------------------------  U14b.txt :  ---------------------------------------------------------------------------------------------  Fondas stipendijai: 1000 Reikalavimas Stipendijai: 9  ---------------------------------------------------------------------------------------------  Eilės numeris Grupė Pažymių kiekis Pažymiai  ---------------------------------------------------------------------------------------------  1 IFF-7/7 8 10;10;10;10;10;10;10;10;  2 IFF-7/13 6 5;6;10;5;9;10;  3 IFF-7/1 1 8;  4 IFF-7/11 4 8;7;10;8;  5 IFF-7/2 3 5;5;5;  6 IFF-7/7 6 6;6;6;7;4;6;10;  7 IFF-7/14 5 5;5;5;5;5;5;  8 IFA-1 5 10;10;10;10;4;  9 IFF-7/1 10 10;10;10;10;10;10;5;5;5;5;5;  10 IFS-0/0 10 10;10;10;10;10;10;10;10;10;10;8;  11 IFA-7/1 2 0;4;  12 IFF-8/1 4 5;4;5;10;  13 IFF-7/7 6 7;8;8;4;10;9;  14 IFF-7/7 6 5;6;9;5;8;7;  15 IFA-7/4 6 10;8;9;8;9;4;  --------------------------------------------------------------------------------------------- |

|  |
| --- |
| Rezultatai.txt |
| ----------------------------------------------------------------------------------------------------------  Pavardė Vardas Telefono nr. Grupė Pažymių vidurkis Gaunama stipendija  ----------------------------------------------------------------------------------------------------------  Jonaitis Jonas 861356918 IFF-7/7 10 523.8  Ramzis Gordonas 861381947 IFS-0/0 9.8 476.19  ---------------------------------------------------------------------------------------------------------- |

|  |
| --- |
| Vartotojo sąsaja |
| https://i.gyazo.com/6b7ea0fedb651fcf3a8552afba9901b2.png |

|  |
| --- |
| Vartotojo sąsaja atrinkus „IFS-0/0“ grupės studentus |
| https://i.gyazo.com/0adbc7d05bf5c66238a282ebcaa19048.png |

## Dėstytojo pastabos

# Bendrinės klasės ir sąsajos (L3)

## Darbo užduotis

## Grafinės vartotojo sąsajos schema

## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
|  |  |  |
|  |  |  |
|  |  |  |

## Klasių diagrama

## Programos vartotojo vadovas

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos

# Kolekcijos ir išimčių valdymas (L4)

## Darbo užduotis

## Grafinės vartotojo sąsajos schema

## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
|  |  |  |
|  |  |  |
|  |  |  |

## Klasių diagrama

## Programos vartotojo vadovas

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos

# Deklaratyvusis programavimas (L5)

## Darbo užduotis

## Grafinės vartotojo sąsajos schema

## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
|  |  |  |
|  |  |  |
|  |  |  |

## Klasių diagrama

## Programos vartotojo vadovas

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos