KAUNO TECHNOLOGIJOS UNIVERSITETAS

Objektinis programavimas (P175B502)

Laboratorinio darbo ataskaita

Atliko:

IF-1/9gr. studentas

Tautvydas Petkus

2012 m. gegužės 23 d.

Priėmė:

Prof. Vacius Jusas

KAUNAS 2012

TURINYS

1. Dinaminis objektų masyvas (L1) 3

1.1. Darbo užduotis 3

1.2. Grafinės vartotojo sąsajos schema 3

1.3. Sąsajoje panaudotų komponentų keičiamos savybės 3

1.4. Programos vartotojo vadovas 4

1.5. Programos tekstas 4

1.6. Pradiniai duomenys ir rezultatai 10

2. Vienkryptis sąrašas (L2) 12

2.1. Darbo užduotis 12

2.2. Grafinės vartotojo sąsajos schema 12

2.3. Sąsajoje panaudotų komponentų keičiamos savybės 12

2.4. Programos vartotojo vadovas 13

2.5. Programos tekstas 13

2.6. Pradiniai duomenys ir rezultatai 23

3. Dvikryptis sąrašas (L3) 25

3.1. Darbo užduotis 25

3.2. Grafinės vartotojo sąsajos schema 25

3.3. Sąsajoje panaudotų komponentų keičiamos savybės 25

3.4. Programos vartotojo vadovas 26

3.5. Programos tekstas 26

3.6. Pradiniai duomenys ir rezultatai 39

4. Objektų sąrašas (L4) 40

4.1. Darbo užduotis 40

4.2. Grafinės vartotojo sąsajos schema 40

4.3. Sąsajoje panaudotų komponentų keičiamos savybės 40

4.4. Programos vartotojo vadovas 41

4.5. Programos tekstas 41

4.6. Pradiniai duomenys ir rezultatai 55

5. Šakotas sąrašas (L5) 57

5.1. Darbo užduotis 57

5.2. Grafinės vartotojo sąsajos schema 57

5.3. Sąsajoje panaudotų komponentų keičiamos savybės 58

5.4. Programos vartotojo vadovas 58

5.5. Programos tekstas 58

5.6. Pradiniai duomenys ir rezultatai 79

# Dinaminis objektų masyvas (L1)

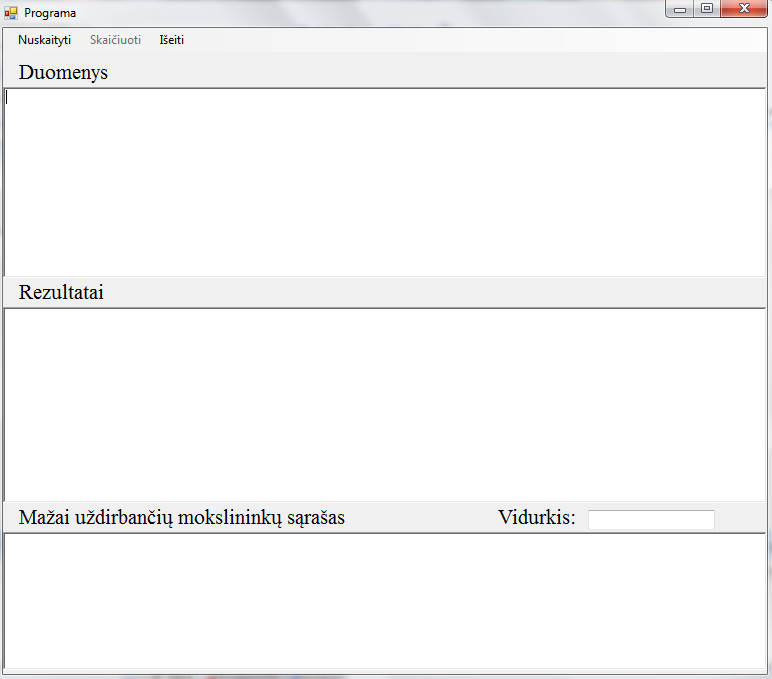
## Darbo užduotis

39. Moksliniai darbuotojai atliko darbus 4 skirtingose temose. Visos temos gavo premijas. Duomenų faile tokia informacija:

Pirmoje eilutėje – premijų dydžiai.

Tolesnėse eilutėse – darbuotojų indėliai, kurie išreikšti naudingumo koeficientu, į eilinę temą. Informacija apie darbuotoją užima vieną eilutę. Remiantis darbuotojų indėliais, suskaičiuoti kiekvienam darbuotojui priklausančios premijos dydį pagal kiekvieną temą atskirai ir bendrą premijų sumą. Suformuoti sąrašą darbuotojų, kurie uždirbo mažiau už vidurkį. Rikiuoti(pavardė, vardas).

## Grafinės vartotojo sąsajos schema



## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Meniu juosta  (ToolStripMenu) | Rodo, kokias funkcijas atlieka programa | Ja pasinaudojus galima nuskaityti duomenis, juos apsikačiuoti, baigti darbą su programa |
| Tekstinių failų laukeliai (richtextbox1, richtextbox2, richtextbox3) | Pavaizduoja atitinkamų tekstinių failų turinį | Parodo nuskaitytų pradinių duomenų failą, rezultatų failą su skaičiavimais ir rikiavimu. |
| Vidurkio reikšmės laukelis (textbox1) | Pavaizduoja, koks buvo vidurkis, pagal kurį buvo atrinktas mažai uždirbančių mokslininkų sąrašas | Padeda vartotojui lengviau įvertinti duomenis |
| Laukelių pavadinimai (label1, label2, label3, label4) | Skirti pavaizduoti antraštėms ar tekstui programos lauke | Skirta vartotojui atskirti duomenų, rezultatų failą, taip pat atskirti atrinktą rezultatų failą bei nurodant textbox1 reikšmę programai. |

## Programos vartotojo vadovas

1. Visi duomenys turi būti surašyti tekstiniame faile Duomenys.txt, pagal užduoties sąlygoje nurodytus reikalavimus.
2. Spauskite mygtuką „Nuskaityti“, kad nuskaitytumėte duomenis iš failo.
3. Atsilaisvinus mygtukui „Skaičiuoti“, spauskite jį, norėdami apskaičiuoti kiekvieno mokslininko uždirbtą premiją bei visas premijas kartu.

Tuo pačiu bus apskaičiuotas vidurkis visų mokslininkų uždirbtų pinigų, kuris bus pavaizduotas laukelyje „Vidurkis“.

3-iajame teksto lauke bus pavaizduotas sąrašas mokslininkų, kurių premijų suma nesiekė vidurkio.

1. Norėdami baigti darbą, spauskite „Išeiti“.

## Programos tekstas

**Mokslininkai.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Mokslininkai

{

public:

staticconstint Cpremijos=4;

private:

string kodas, vardas, pavarde;

double ind[Cpremijos], pin[Cpremijos] , S;

public:

Mokslininkai() {}

Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]);

string Spausdinti();

string SpausdintiPapildomai();

void ApskaiciuotiS();

void DetiUzdirbius(double a, int b);

double ImtiS();

double ImtiIndelius(int i);

booloperator> (Mokslininkai & C);

};

**Mokslininkai.cpp:**

#include"StdAfx.h"

#include"Mokslininkai.h"

usingnamespace std;

Mokslininkai::Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]) :

kodas(kodasR), vardas(vardasR), pavarde(pavardeR), S(0)

{

for (int i = 0; i < Cpremijos; i++){

ind[i] = indR[i];

pin[i] = 0;

}

}

string Mokslininkai::Spausdinti()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

return ss.str();

}

string Mokslininkai::SpausdintiPapildomai()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

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

ss <<' '<< right << setw(6) << ((int)pin[i]);

ss << right << setw(10) << ((int)S);

return ss.str();

}

//----------------------------------

double Mokslininkai::ImtiS() { return S; }

//----------------------------------

double Mokslininkai::ImtiIndelius(int i)

{

return ind[i];

}

//----------------------------------

void Mokslininkai::ApskaiciuotiS(){

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

S+= pin[i];

}

//----------------------------------

void Mokslininkai::DetiUzdirbius(double a, int b)

{

pin[b] = a;

}

//------------------------------------

bool Mokslininkai::operator> (Mokslininkai & C)

{

return pavarde > C.pavarde || ((pavarde == C.pavarde) && (vardas > C.vardas));

}

**Fondas.h:**

#pragmaonce

#include"Mokslininkai.h"

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Fondas

{

public:

staticconstint Cpremijos=4;

staticconstint Cmoksl=32;

private:

double Premija[Cpremijos];

Mokslininkai \*M;

int n;

public:

Fondas();

~Fondas();

void DetiKiek(int a);

int Kiek();

Mokslininkai Imti(int i);

void DetiP(double p, int i);

void Deti(Mokslininkai p);

double ImtiP (int i);

void Paskirstymas();

void Rikiavimas();

};

**Fondas.cpp:**

#include"StdAfx.h"

#include"Fondas.h"

usingnamespace std;

Fondas::Fondas() : n(0), M(NULL)

{

M = new Mokslininkai[Cmoksl];

}

Fondas::~Fondas()

{

delete [] M;

}

void Fondas::DetiKiek(int a) { n = a; }

//--------------------------------------------------------

int Fondas::Kiek()

{ return n; }

//--------------------------------------------------------

Mokslininkai Fondas::Imti(int i)

{ return M[i];}

//--------------------------------------------------------

void Fondas::Deti(Mokslininkai p) {

if (n < Cmoksl)

M[n] = p;

}

//--------------------------------------------------------

void Fondas::DetiP(double p, int i)

{

Premija[i] = p;

}

//--------------------------------------------------------

double Fondas::ImtiP(int i)

{

return Premija[i];

}

//--------------------------------------------------------

void Fondas::Paskirstymas() {

double indS = 0;

double pinigai;

for (int i = 0; i < Cpremijos; i++){

indS = 0;

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

indS+= M[j].ImtiIndelius(i);

for (int j = 0; j < n; j++){

pinigai = (M[j].ImtiIndelius(i) / indS) \* Premija[i];

M[j].DetiUzdirbius(pinigai, i);

}

}

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

M[i].ApskaiciuotiS();

}

//-------------------------------------------------------

void Fondas::Rikiavimas()

{

Mokslininkai CC;

for (int i = 0; i < n - 1; i++)

for (int j = i + 1; j < n; j++)

if (M[i] > M[j])

{

CC = M[i];

M[i] = M[j];

M[j] = CC;

}

}

**Form1.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

#include"Mokslininkai.h"

#include"Fondas.h"

const string Cduom = "Duomenys.txt";

const string Crez = "Rezultatai.txt";

const string Crez2= "Rezultatai2.txt";

const string Crez3= "Rezultatai3.txt";

constdouble Cprem = 4;

namespace S2L1 {

usingnamespace System;

usingnamespace System::ComponentModel;

usingnamespace System::Collections;

usingnamespace System::Windows::Forms;

usingnamespace System::Data;

usingnamespace System::Drawing;

void Duomenys(Fondas \* F)

{

ifstream duom(Cduom.c\_str());

string kod, pav, vard;

double indelis[Mokslininkai::Cpremijos];

double p[Fondas::Cpremijos];

F->DetiKiek(0);

for (int i = 0; i < Fondas::Cpremijos; i++)

duom >> p[i];

for (int i = 0; i < Fondas::Cpremijos; i++)

F->DetiP(p[i], i);

while(!duom.eof()) {

getline (duom, kod, ',');

getline (duom, pav, ',');

getline (duom, vard, ',');

for (int i=0; i<Mokslininkai::Cpremijos; i++)

duom >> indelis[i];

F->Deti(Mokslininkai(kod, pav, vard, indelis));

F->DetiKiek(F->Kiek() + 1);

}

duom.close();

}

//-----------------------------------------------------

// Spausdinami objekto F duomenys

void Isvesti(Fondas \* F)

{

ofstream rez(Crez.c\_str(), ios::app);

for (int i = 0; i < Fondas::Cpremijos; i++)

rez <<' '<< F->ImtiP(i);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(15) <<"Indeliai \n";

rez <<"-----------------------------------------\n";

for (int i = 0; i < F->Kiek(); i++)

rez << F->Imti(i).Spausdinti();

rez << endl <<"--------------------------------------\n\n\n";

rez.close();

}

//-----------------------------------------------------

void Apskaiciavimas(Fondas \* F, Fondas \* R)

{

R->DetiKiek(0);

for (int i = 0; i < Fondas::Cpremijos; i++)

R->DetiP(F->ImtiP(i), i);

for (int i = 0; i < F->Kiek(); i++){

R->Deti(F->Imti(i));

R->DetiKiek(R->Kiek() + 1);

}

R->Paskirstymas();

R->Rikiavimas();

}

//-----------------------------------------------------

void RezultatuIsvedimas(Fondas \* R, string rezFailas)

{

ofstream rez(rezFailas.c\_str(), ios::app);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(20) <<"Indeliai"<<

setw(30) <<"Premijos"<< setw(10) <<"Suma"<< endl;

rez <<"-----------------------------------------------------------------------------------------\n";

for (int i = 0; i < R->Kiek(); i++)

rez << R->Imti(i).SpausdintiPapildomai();

rez << endl;

rez <<"-----------------------------------------------------------------------------------------\n";

rez.close();

}

//----------------------------------------------------

double Vidurkiai(Fondas \* R, Fondas \* U)

{

double PremijuV = 0;

U->DetiKiek(0);

for (int i = 0; i < R->Kiek(); i++)

PremijuV += R->Imti(i).ImtiS();

PremijuV = PremijuV / R->Kiek();

for (int i = 0; i < R->Kiek(); i++)

if (R->Imti(i).ImtiS() < PremijuV)

{

U->Deti(R->Imti(i));

U->DetiKiek(U->Kiek() + 1);

}

return PremijuV;

}

//-----------------------------------------------------

/// <summary>

/// Summary for Form1

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

publicrefclass Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

info = new Fondas;

infoc = new Fondas;

infoc2 = new Fondas;

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

}

private: Fondas \*info, \*infoc, \*infoc2;

private: System::Windows::Forms::MenuStrip^ menuStrip1;

protected:

private: System::Windows::Forms::ToolStripMenuItem^ nuskaitytiToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ išeitiToolStripMenuItem;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::RichTextBox^ richTextBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::RichTextBox^ richTextBox3;

private: System::Windows::Forms::ToolStripMenuItem^ skaičiuotiToolStripMenuItem;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox1;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container ^components;

#pragmaregion Windows Form Designer generated code {.....}

#pragmaendregion

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void išeitiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void nuskaitytiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

ofstream rez(Crez.c\_str());

rez.close();

ofstream rez2(Crez2.c\_str());

rez2.close();

ofstream rez3(Crez3.c\_str());

rez3.close();

Duomenys(info);

Isvesti(info);

richTextBox1->LoadFile("Rezultatai.txt", RichTextBoxStreamType::PlainText);

skaičiuotiToolStripMenuItem->Enabled = true;

}

private: System::Void richTextBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void menuStrip1\_ItemClicked(System::Object^ sender, System::Windows::Forms::ToolStripItemClickedEventArgs^ e) {

}

private: System::Void richTextBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void skaičiuotiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Apskaiciavimas(info, infoc);

double vidurkis = Vidurkiai(infoc, infoc2);

RezultatuIsvedimas(infoc, Crez2);

RezultatuIsvedimas(infoc2, Crez3);

richTextBox2->LoadFile("Rezultatai2.txt", RichTextBoxStreamType::PlainText);

richTextBox3->LoadFile("Rezultatai3.txt", RichTextBoxStreamType::PlainText);

textBox1->Text = ((int)vidurkis).ToString();

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

};

}

## Pradiniai duomenys ir rezultatai

**Duomenys.txt:**

5000 2500 8000 1000

38604163754, Hhhhhh, Hhhhhhhhh, 5 9 7 2

37407011234, Gggggg, Ggggggggg, 2 1 0 3

39012128243, Ffffff, Fffffffff, 6 7 1 9

36401288430, Eeeeee, Eeeeeeeee, 1 2 3 4

37702131348, Dddddd, Ddddddddd, 9 4 3 6

37103044686, Cccccc, Ccccccccc, 4 5 4 6

33008141568, Aaaaaa, Aaaaaaaaa, 7 2 2 3

39011128434, Bbbbbb, Bbbbbbbbb, 7 6 1 9

**Rezultatai.txt:**

5000 2500 8000 1000

Kodas Pavarde Vardas Indeliai

-----------------------------------------

38604163754 HhhhhhHhhhhhhhh 5 9 7 2

37407011234 GgggggGgggggggg 2 1 0 3

39012128243 FfffffFffffffff 6 7 1 9

36401288430 EeeeeeEeeeeeeee 1 2 3 4

37702131348 DdddddDdddddddd 9 4 3 6

37103044686 CcccccCcccccccc 4 5 4 6

33008141568 AaaaaaAaaaaaaaa 7 2 2 3

39011128434 BbbbbbBbbbbbbbb 7 6 1 9

--------------------------------------

**Rezultatai2.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

37103044686 CcccccCcccccccc 4 5 4 6 487 347 1523 142 2501

37702131348 DdddddDdddddddd 9 4 3 6 1097 277 1142 142 2661

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

38604163754 HhhhhhHhhhhhhhh 5 9 7 2 609 625 2666 47 3949

-----------------------------------------------------------------------------------------

**Rezultatai3.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

-----------------------------------------------------------------------------------------

# Vienkryptis sąrašas (L2)

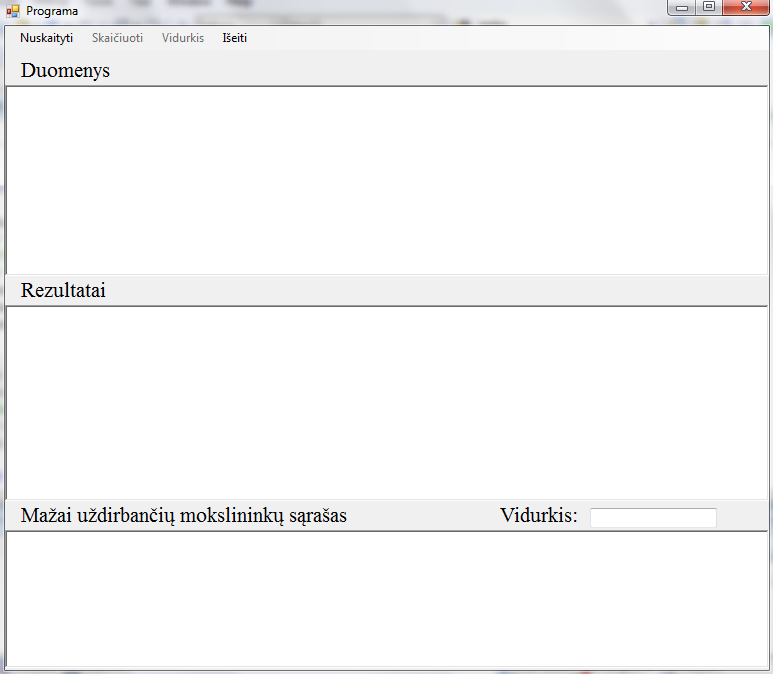
## Darbo užduotis

39. Moksliniai darbuotojai atliko darbus 4 skirtingose temose. Visos temos gavo premijas. Duomenų faile tokia informacija:

Pirmoje eilutėje – premijų dydžiai.

Tolesnėse eilutėse – darbuotojų indėliai, kurie išreikšti naudingumo koeficientu, į eilinę temą. Informacija apie darbuotoją užima vieną eilutę. Remiantis darbuotojų indėliais, suskaičiuoti kiekvienam darbuotojui priklausančios premijos dydį pagal kiekvieną temą atskirai ir bendrą premijų sumą. Suformuoti sąrašą darbuotojų, kurie uždirbo mažiau už vidurkį. Rikiuoti(pavardė, vardas).

## Grafinės vartotojo sąsajos schema



## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Meniu juosta  (ToolStripMenu) | Rodo, kokias funkcijas atlieka programa | Ja pasinaudojus galima nuskaityti duomenis, juos apsikačiuoti, išvesti vidurkius, baigti darbą su programa |
| Tekstinių failų laukeliai (richtextbox1, richtextbox2, richtextbox3) | Pavaizduoja atitinkamų tekstinių failų turinį | Parodo nuskaitytų pradinių duomenų failą, rezultatų failą su skaičiavimais ir rikiavimu. |
| Vidurkio reikšmės laukelis (textbox1) | Pavaizduoja, koks buvo vidurkis, pagal kurį buvo atrinktas mažai uždirbančių mokslininkų sąrašas | Padeda vartotojui lengviau įvertinti duomenis |
| Laukelių pavadinimai (label1, label2, label3, label4) | Skirti pavaizduoti antraštėms ar tekstui programos lauke | Skirta vartotojui atskirti duomenų, rezultatų failą, taip pat atskirti atrinktą rezultatų failą bei nurodant textbox1 reikšmę programai. |

## Programos vartotojo vadovas

1. Visi duomenys turi būti surašyti tekstiniame faile Duomenys.txt, pagal užduoties sąlygoje nurodytus reikalavimus.

2. Spauskite mygtuką „Nuskaityti“, kad nuskaitytumėte duomenis iš failo.

3. Atsilaisvinus mygtukui „Skaičiuoti“, spauskite jį, norėdami apskaičiuoti kiekvieno mokslininko uždirbtą premiją bei visas premijas kartu.

4. Spauskite mygtuką skaičiuoti, jei norite apskaičiuoti premijų vidurkį

3-iajame teksto lauke bus pavaizduotas sąrašas mokslininkų, kurių premijų suma nesiekė vidurkio.

4. Norėdami baigti darbą, spauskite „Išeiti“.

## Programos tekstas

**Mokslininkai.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Mokslininkai

{

public:

staticconstint Cpremijos=4;

private:

string kodas, vardas, pavarde;

double ind[Cpremijos], pin[Cpremijos] , S;

public:

Mokslininkai() {}

Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]);

string Spausdinti();

string SpausdintiPapildomai();

void ApskaiciuotiS();

void DetiUzdirbius(double a, int b);

double ImtiS();

double ImtiIndelius(int i);

booloperator> (constMokslininkai & C);

};

**Mokslininkai.cpp:**

#include"StdAfx.h"

#include"Mokslininkai.h"

usingnamespace std;

Mokslininkai::Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]) :

kodas(kodasR), vardas(vardasR), pavarde(pavardeR), S(0)

{

for (int i = 0; i < Cpremijos; i++){

ind[i] = indR[i];

pin[i] = 0;

}

}

string Mokslininkai::Spausdinti()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

return ss.str();

}

string Mokslininkai::SpausdintiPapildomai()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

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

ss <<' '<< right << setw(6) << ((int)pin[i]);

ss << right << setw(10) << ((int)S);

return ss.str();

}

//----------------------------------

double Mokslininkai::ImtiS() { return S; }

//----------------------------------

double Mokslininkai::ImtiIndelius(int i)

{

return ind[i];

}

//----------------------------------

void Mokslininkai::ApskaiciuotiS(){

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

S+= pin[i];

}

//----------------------------------

void Mokslininkai::DetiUzdirbius(double a, int b)

{

pin[b] = a;

}

//------------------------------------

bool Mokslininkai::operator> (const Mokslininkai & C)

{

return pavarde < C.pavarde || ((pavarde == C.pavarde) && (vardas < C.vardas));

}

**Fondas.h:**

#pragmaonce

#include"Mokslininkai.h"

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Fondas

{

public:

staticconstint Cpremijos=4;

private:

struct Mazgas{

Mokslininkai duom;

Mazgas \*sek;

};

double Premija[Cpremijos];

Mazgas \*p;

Mazgas \*d;

public:

Fondas();

~Fondas();

void Pradzia();

void Sekantis();

bool Yra();

Mokslininkai ImtiDuomenis();

void DetiDuomenis(Mokslininkai x);

void DetiIPradzia(Mokslininkai x);

void DetiIPabaiga(Mokslininkai x);

void DetiP(double p, int i);

double ImtiP (int i);

void Paskirstymas();

void Rikiavimas();

};

**Fondas.cpp:**

#include"StdAfx.h"

#include"Fondas.h"

usingnamespace std;

Fondas::Fondas() : p(NULL), d(NULL) {}

Fondas::~Fondas()

{

while (p != NULL){

d = p;

p = p->sek;

delete d;

}

}

//-----------------Baziniai saraso veiksmai---------------------

void Fondas::Pradzia() { d = p; }

void Fondas::Sekantis() {d = d->sek;}

bool Fondas::Yra() {return d != NULL; }

Mokslininkai Fondas::ImtiDuomenis() {return d->duom; }

void Fondas::DetiDuomenis(Mokslininkai x) { d->duom = x;}

//-----------------Naujas elementas dedamas i pradzia----------

void Fondas::DetiIPradzia(Mokslininkai x)

{

Mazgas \*t = new Mazgas;

t->duom = x;

t->sek = p;

p = t;

}

//----------------Naujas elementas dedamas i pabaiga-----------

void Fondas::DetiIPabaiga(Mokslininkai x)

{

Mazgas \*t = new Mazgas;

t->duom = x;

t->sek = NULL;

if (p != NULL){

Mazgas \*v = new Mazgas;

for (v = p; v->sek != NULL; v = v->sek);

v->sek = t;

}

else

p = t;

}

//--------------------------------------------------------\*/

void Fondas::DetiP(double p, int i)

{

Premija[i] = p;

}

//--------------------------------------------------------

double Fondas::ImtiP(int i)

{

return Premija[i];

}

//--------------------------------------------------------

void Fondas::Paskirstymas() {

double indS = 0;

double pinigai;

for (int i = 0; i < Cpremijos; i++){

indS = 0;

Pradzia();

while (Yra()){

indS+= ImtiDuomenis().ImtiIndelius(i);

Sekantis();

}

Pradzia();

while (Yra()){

pinigai = (ImtiDuomenis().ImtiIndelius(i) / indS) \* Premija[i];

Mokslininkai x;

x = Mokslininkai(ImtiDuomenis());

x.DetiUzdirbius(pinigai, i);

DetiDuomenis(x);

Sekantis();

}

}

Pradzia();

while (Yra()){

Mokslininkai x;

x = Mokslininkai(ImtiDuomenis());

x.ApskaiciuotiS();

DetiDuomenis(x);

Sekantis();

}

}

//-------------------------------------------------------

void Fondas::Rikiavimas()

{

bool yra = true;

Mazgas \*d, \*d1, \*r;

while (yra) {

yra = false;

d = d1 = r = p;

while (d1 != NULL) {

if (d1->duom > d->duom) {

yra = true;

if (d == p) { // pirmas elementas su kaimynu

p = p->sek;

d->sek = d1->sek;

d1->sek = d;

}

else { // pora tolesnių elementų

d->sek = d1->sek;

d1->sek = d;

r->sek = d1;

}

}

r = d;

d = d1;

d1 = d1->sek;

}

}

}

**Form1.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

#include"Mokslininkai.h"

#include"Fondas.h"

const string Cduom = "Duomenys.txt";

const string Crez = "Rezultatai.txt";

const string Crez2= "Rezultatai2.txt";

const string Crez3= "Rezultatai3.txt";

constdouble Cprem = 4;

namespace S2L1 {

usingnamespace System;

usingnamespace System::ComponentModel;

usingnamespace System::Collections;

usingnamespace System::Windows::Forms;

usingnamespace System::Data;

usingnamespace System::Drawing;

void Duomenys(Fondas \* F)

{

ifstream duom(Cduom.c\_str());

string kod, pav, vard;

double indelis[Mokslininkai::Cpremijos];

double p[Fondas::Cpremijos];

F->Pradzia();

for (int i = 0; i < Fondas::Cpremijos; i++){

duom >> p[i];

F->DetiP(p[i], i);

}

while(!duom.eof()) {

getline (duom, kod, ',');

getline (duom, pav, ',');

getline (duom, vard, ',');

for (int i=0; i<Mokslininkai::Cpremijos; i++)

duom >> indelis[i];

Mokslininkai x;

x = Mokslininkai(kod, pav, vard, indelis);

F->DetiIPabaiga(x);

}

duom.close();

}

//-----------------------------------------------------

// Spausdinami objekto F duomenys

void Isvesti(Fondas \* F)

{

ofstream rez(Crez.c\_str(), ios::app);

for (int i = 0; i < Fondas::Cpremijos; i++)

rez <<' '<< F->ImtiP(i);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(15) <<"Indeliai \n";

rez <<"-----------------------------------------\n";

F->Pradzia();

while (F->Yra()){

rez << F->ImtiDuomenis().Spausdinti();

F->Sekantis();

}

rez << endl <<"--------------------------------------\n\n\n";

rez.close();

}

//-----------------------------------------------------

void Apskaiciavimas(Fondas \* F, Fondas \* R)

{

for (int i = 0; i < Fondas::Cpremijos; i++)

R->DetiP(F->ImtiP(i), i);

F->Pradzia();

while (F->Yra()){

R->DetiIPradzia(F->ImtiDuomenis());

F->Sekantis();

}

R->Paskirstymas();

R->Rikiavimas();

}

//-----------------------------------------------------

void RezultatuIsvedimas(Fondas \* R, string rezFailas)

{

ofstream rez(rezFailas.c\_str(), ios::app);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(20) <<"Indeliai"<<

setw(30) <<"Premijos"<< setw(10) <<"Suma"<< endl;

rez <<"-----------------------------------------------------------------------------------------\n";

R->Pradzia();

while (R->Yra()){

rez << R->ImtiDuomenis().SpausdintiPapildomai();

R->Sekantis();

}

rez << endl;

rez <<"-----------------------------------------------------------------------------------------\n";

rez.close();

}

//----------------------------------------------------

double Vidurkiai(Fondas \* R, Fondas \* U)

{

for (int i = 0; i < Fondas::Cpremijos; i++)

U->DetiP(R->ImtiP(i), i);

R->Pradzia();

double PremijuV = 0;

int n = 0;

while (R->Yra()){

PremijuV += R->ImtiDuomenis().ImtiS();

R->Sekantis();

n++;

}

R->Pradzia();

PremijuV = PremijuV / n;

while(R->Yra()){

if (R->ImtiDuomenis().ImtiS() < PremijuV)

U->DetiIPabaiga(R->ImtiDuomenis());

R->Sekantis();

}

return PremijuV;

}

//-----------------------------------------------------

/// <summary>

/// Summary for Form1

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

publicrefclass Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

info = new Fondas;

infoc = new Fondas;

infoc2 = new Fondas;

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

delete info;

delete infoc;

delete infoc2;

}

private: Fondas \*info, \*infoc, \*infoc2;

private: System::Windows::Forms::MenuStrip^ menuStrip1;

protected:

private: System::Windows::Forms::ToolStripMenuItem^ nuskaitytiToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ išeitiToolStripMenuItem;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::RichTextBox^ richTextBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::RichTextBox^ richTextBox3;

private: System::Windows::Forms::ToolStripMenuItem^ skaičiuotiToolStripMenuItem;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::ToolStripMenuItem^ vidurkisToolStripMenuItem;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container ^components;

#pragmaregion Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());

this->nuskaitytiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->skaičiuotiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->vidurkisToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->išeitiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->richTextBox2 = (gcnew System::Windows::Forms::RichTextBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->richTextBox3 = (gcnew System::Windows::Forms::RichTextBox());

this->label4 = (gcnew System::Windows::Forms::Label());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->menuStrip1->SuspendLayout();

this->SuspendLayout();

//

// menuStrip1

//

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(4) {this->nuskaitytiToolStripMenuItem,

this->skaičiuotiToolStripMenuItem, this->vidurkisToolStripMenuItem, this->išeitiToolStripMenuItem});

this->menuStrip1->Location = System::Drawing::Point(0, 0);

this->menuStrip1->Name = L"menuStrip1";

this->menuStrip1->Size = System::Drawing::Size(764, 24);

this->menuStrip1->TabIndex = 0;

this->menuStrip1->Text = L"menuStrip1";

this->menuStrip1->ItemClicked += gcnew System::Windows::Forms::ToolStripItemClickedEventHandler(this, &Form1::menuStrip1\_ItemClicked);

//

// nuskaitytiToolStripMenuItem

//

this->nuskaitytiToolStripMenuItem->Name = L"nuskaitytiToolStripMenuItem";

this->nuskaitytiToolStripMenuItem->Size = System::Drawing::Size(72, 20);

this->nuskaitytiToolStripMenuItem->Text = L"Nuskaityti";

this->nuskaitytiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::nuskaitytiToolStripMenuItem\_Click);

//

// skaičiuotiToolStripMenuItem

//

this->skaičiuotiToolStripMenuItem->Enabled = false;

this->skaičiuotiToolStripMenuItem->Name = L"skaičiuotiToolStripMenuItem";

this->skaičiuotiToolStripMenuItem->Size = System::Drawing::Size(70, 20);

this->skaičiuotiToolStripMenuItem->Text = L"Skaičiuoti";

this->skaičiuotiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::skaičiuotiToolStripMenuItem\_Click);

//

// vidurkisToolStripMenuItem

//

this->vidurkisToolStripMenuItem->Enabled = false;

this->vidurkisToolStripMenuItem->Name = L"vidurkisToolStripMenuItem";

this->vidurkisToolStripMenuItem->Size = System::Drawing::Size(61, 20);

this->vidurkisToolStripMenuItem->Text = L"Vidurkis";

this->vidurkisToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::vidurkisToolStripMenuItem\_Click);

//

// išeitiToolStripMenuItem

//

this->išeitiToolStripMenuItem->Name = L"išeitiToolStripMenuItem";

this->išeitiToolStripMenuItem->Size = System::Drawing::Size(43, 20);

this->išeitiToolStripMenuItem->Text = L"Išeiti";

this->išeitiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::išeitiToolStripMenuItem\_Click);

//

// richTextBox1

//

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox1->Location = System::Drawing::Point(0, 59);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->Size = System::Drawing::Size(764, 191);

this->richTextBox1->TabIndex = 1;

this->richTextBox1->Text = L"";

this->richTextBox1->TextChanged += gcnew System::EventHandler(this, &Form1::richTextBox1\_TextChanged);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label1->Location = System::Drawing::Point(12, 33);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(99, 23);

this->label1->TabIndex = 2;

this->label1->Text = L"Duomenys";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label2->Location = System::Drawing::Point(12, 253);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(95, 23);

this->label2->TabIndex = 4;

this->label2->Text = L"Rezultatai";

//

// richTextBox2

//

this->richTextBox2->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox2->Location = System::Drawing::Point(0, 279);

this->richTextBox2->Name = L"richTextBox2";

this->richTextBox2->Size = System::Drawing::Size(764, 196);

this->richTextBox2->TabIndex = 3;

this->richTextBox2->Text = L"";

this->richTextBox2->TextChanged += gcnew System::EventHandler(this, &Form1::richTextBox2\_TextChanged);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label3->Location = System::Drawing::Point(12, 478);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(336, 23);

this->label3->TabIndex = 6;

this->label3->Text = L"Mažai uždirbančių mokslininkų sąrašas";

//

// richTextBox3

//

this->richTextBox3->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox3->Location = System::Drawing::Point(0, 504);

this->richTextBox3->Name = L"richTextBox3";

this->richTextBox3->Size = System::Drawing::Size(764, 138);

this->richTextBox3->TabIndex = 5;

this->richTextBox3->Text = L"";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label4->Location = System::Drawing::Point(491, 478);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(88, 23);

this->label4->TabIndex = 7;

this->label4->Text = L"Vidurkis:";

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(585, 482);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(127, 20);

this->textBox1->TabIndex = 8;

this->textBox1->TextChanged += gcnew System::EventHandler(this, &Form1::textBox1\_TextChanged);

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(764, 644);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->richTextBox3);

this->Controls->Add(this->label2);

this->Controls->Add(this->richTextBox2);

this->Controls->Add(this->label1);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->menuStrip1);

this->MainMenuStrip = this->menuStrip1;

this->Name = L"Form1";

this->Text = L"Programa";

this->Load += gcnew System::EventHandler(this, &Form1::Form1\_Load);

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragmaendregion

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void išeitiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void nuskaitytiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

ofstream rez(Crez.c\_str());

rez.close();

ofstream rez2(Crez2.c\_str());

rez2.close();

ofstream rez3(Crez3.c\_str());

rez3.close();

Duomenys(info);

Isvesti(info);

richTextBox1->LoadFile("Rezultatai.txt", RichTextBoxStreamType::PlainText);

skaičiuotiToolStripMenuItem->Enabled = true;

}

private: System::Void richTextBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void menuStrip1\_ItemClicked(System::Object^ sender, System::Windows::Forms::ToolStripItemClickedEventArgs^ e) {

}

private: System::Void richTextBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void skaičiuotiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Apskaiciavimas(info, infoc);

RezultatuIsvedimas(infoc, Crez2);

vidurkisToolStripMenuItem->Enabled = true;

richTextBox2->LoadFile("Rezultatai2.txt", RichTextBoxStreamType::PlainText);

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void vidurkisToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

double vidurkis = Vidurkiai(infoc, infoc2);

RezultatuIsvedimas(infoc2, Crez3);

richTextBox3->LoadFile("Rezultatai3.txt", RichTextBoxStreamType::PlainText);

textBox1->Text = ((int)vidurkis).ToString();

}

};

}

## Pradiniai duomenys ir rezultatai

**Duomenys.txt:**

5000 2500 8000 1000

38604163754, Hhhhhh, Hhhhhhhhh, 5 9 7 2

37407011234, Gggggg, Ggggggggg, 2 1 0 3

39012128243, Ffffff, Fffffffff, 6 7 1 9

36401288430, Eeeeee, Eeeeeeeee, 1 2 3 4

37702131348, Dddddd, Ddddddddd, 9 4 3 6

37103044686, Cccccc, Ccccccccc, 4 5 4 6

33008141568, Aaaaaa, Aaaaaaaaa, 7 2 2 3

39011128434, Bbbbbb, Bbbbbbbbb, 7 6 1 9

**Rezultatai.txt:**

5000 2500 8000 1000

Kodas Pavarde Vardas Indeliai

-----------------------------------------

38604163754 HhhhhhHhhhhhhhh 5 9 7 2

37407011234 GgggggGgggggggg 2 1 0 3

39012128243 FfffffFffffffff 6 7 1 9

36401288430 EeeeeeEeeeeeeee 1 2 3 4

37702131348 DdddddDdddddddd 9 4 3 6

37103044686 CcccccCcccccccc 4 5 4 6

33008141568 AaaaaaAaaaaaaaa 7 2 2 3

39011128434 BbbbbbBbbbbbbbb 7 6 1 9

--------------------------------------

**Rezultatai2.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

37103044686 CcccccCcccccccc 4 5 4 6 487 347 1523 142 2501

37702131348 DdddddDdddddddd 9 4 3 6 1097 277 1142 142 2661

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

38604163754 HhhhhhHhhhhhhhh 5 9 7 2 609 625 2666 47 3949

-----------------------------------------------------------------------------------------

**Rezultatai3.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

**-----------------------------------------------------------------------------------------**

# Dvikryptis sąrašas (L3)

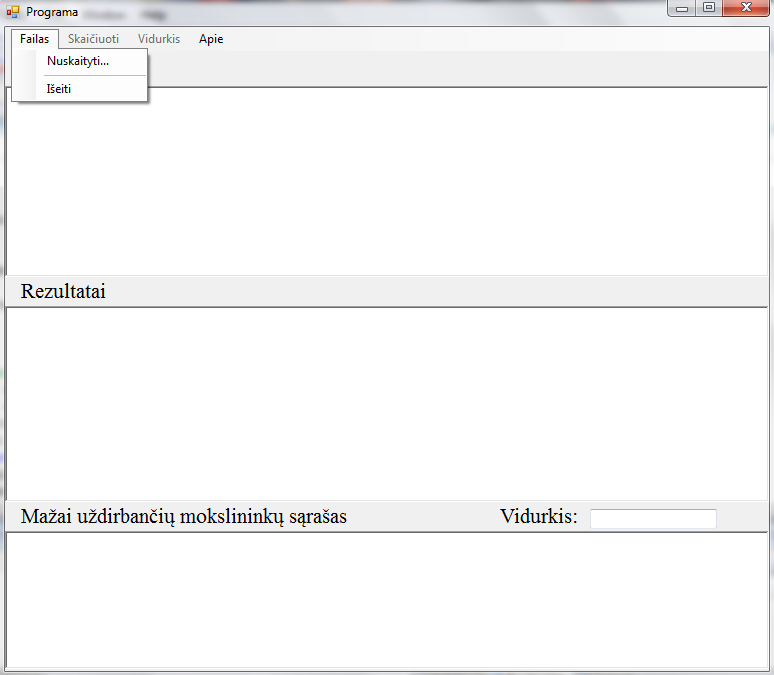
## Darbo užduotis

39. Moksliniai darbuotojai atliko darbus 4 skirtingose temose. Visos temos gavo premijas. Duomenų faile tokia informacija:

Pirmoje eilutėje – premijų dydžiai.

Tolesnėse eilutėse – darbuotojų indėliai, kurie išreikšti naudingumo koeficientu, į eilinę temą. Informacija apie darbuotoją užima vieną eilutę. Remiantis darbuotojų indėliais, suskaičiuoti kiekvienam darbuotojui priklausančios premijos dydį pagal kiekvieną temą atskirai ir bendrą premijų sumą. Suformuoti sąrašą darbuotojų, kurie uždirbo mažiau už vidurkį. Rikiuoti(pavardė, vardas).

## Grafinės vartotojo sąsajos schema



## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Meniu juosta  (ToolStripMenu) | Rodo, kokias funkcijas atlieka programa | Ja pasinaudojus galima nuskaityti duomenis, juos apsikačiuoti, išvesti vidurkius, baigti darbą su programa |
| Tekstinių failų laukeliai (richtextbox1, richtextbox2, richtextbox3) | Pavaizduoja atitinkamų tekstinių failų turinį | Parodo nuskaitytų pradinių duomenų failą, rezultatų failą su skaičiavimais ir rikiavimu. |
| Vidurkio reikšmės laukelis (textbox1) | Pavaizduoja, koks buvo vidurkis, pagal kurį buvo atrinktas mažai uždirbančių mokslininkų sąrašas | Padeda vartotojui lengviau įvertinti duomenis |
| Laukelių pavadinimai (label1, label2, label3, label4) | Skirti pavaizduoti antraštėms ar tekstui programos lauke | Skirta vartotojui atskirti duomenų, rezultatų failą, taip pat atskirti atrinktą rezultatų failą bei nurodant textbox1 reikšmę programai. |
| Dialogo langas | Skirtas pateikti svarbią informaciją vartotojui | Skirtas bendrai informacijai apie programos pavadinimą, kūrėją ir datą |

## Programos vartotojo vadovas

1. Pasirinkite failą, kurį norite nuskaityti, paspaudę mygtuką Failas->Nuskaityti...

2. Atsilaisvinus mygtukui „Skaičiuoti“, spauskite jį, norėdami apskaičiuoti kiekvieno mokslininko uždirbtą premiją bei visas premijas kartu.

3. Spauskite mygtuką „Vidurkis“, jei norite apskaičiuoti premijų vidurkį

3-iajame teksto lauke bus pavaizduotas sąrašas mokslininkų, kurių premijų suma nesiekė vidurkio.

4. Mygtukas „Apie“ pateikia bendrąją informaciją apie programą.

5. Norėdami baigti darbą, spauskite „Išeiti“.

## Programos tekstas

**Mokslininkai.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Mokslininkai

{

public:

staticconstint Cpremijos=4;

private:

string kodas, vardas, pavarde;

double ind[Cpremijos], pin[Cpremijos] , S;

public:

Mokslininkai() {}

Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]);

string Spausdinti();

string SpausdintiPapildomai();

void ApskaiciuotiS();

void DetiUzdirbius(double a, int b);

double ImtiS();

double ImtiIndelius(int i);

booloperator> (constMokslininkai & C);

};

**Mokslininkai.cpp:**

#include"StdAfx.h"

#include"Mokslininkai.h"

usingnamespace std;

Mokslininkai::Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]) :

kodas(kodasR), vardas(vardasR), pavarde(pavardeR), S(0)

{

for (int i = 0; i < Cpremijos; i++){

ind[i] = indR[i];

pin[i] = 0;

}

}

string Mokslininkai::Spausdinti()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

return ss.str();

}

string Mokslininkai::SpausdintiPapildomai()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

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

ss <<' '<< right << setw(6) << ((int)pin[i]);

ss << right << setw(10) << ((int)S);

return ss.str();

}

//----------------------------------

double Mokslininkai::ImtiS() { return S; }

//----------------------------------

double Mokslininkai::ImtiIndelius(int i)

{

return ind[i];

}

//----------------------------------

void Mokslininkai::ApskaiciuotiS(){

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

S+= pin[i];

}

//----------------------------------

void Mokslininkai::DetiUzdirbius(double a, int b)

{

pin[b] = a;

}

//------------------------------------

bool Mokslininkai::operator> (const Mokslininkai & C)

{

return pavarde < C.pavarde || ((pavarde == C.pavarde) && (vardas < C.vardas));

}

**Fondas.h:**

#pragmaonce

#include"Mokslininkai.h"

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Fondas

{

public:

staticconstint Cpremijos=4;

private:

struct Mazgas{

Mokslininkai duom;

Mazgas \*pirmyn;

Mazgas \*atgal;

};

double Premija[Cpremijos];

Mazgas \*pradzia;

Mazgas \*pabaiga;

Mazgas \*d;

public:

Fondas();

~Fondas();

void Pradzia();

void Pabaiga();

void Sekantis();

void Ankstesnis();

bool Yra();

Mokslininkai ImtiDuomenis();

void DetiDuomenis(Mokslininkai x);

void DetiIPradzia(Mokslininkai x);

void DetiIPabaiga(Mokslininkai x);

void DetiP(double p, int i);

double ImtiP (int i);

void Paskirstymas();

void Rikiavimas();

};

**Fondas.cpp:**

#include"StdAfx.h"

#include"Fondas.h"

usingnamespace std;

Fondas::Fondas() : pradzia(NULL), pabaiga(NULL), d(NULL) {}

Fondas::~Fondas()

{

while (pradzia != pabaiga){

d = pradzia;

pradzia = pradzia->pirmyn;

delete d;

}

}

//-----------------Baziniai saraso veiksmai---------------------

void Fondas::Pradzia() { d = pradzia->pirmyn; }

void Fondas::Pabaiga() { d = pabaiga->atgal; }

void Fondas::Sekantis() {d = d->pirmyn;}

void Fondas::Ankstesnis() {d = d->atgal;}

bool Fondas::Yra() {return d != pabaiga; }

Mokslininkai Fondas::ImtiDuomenis() {return d->duom; }

void Fondas::DetiDuomenis(Mokslininkai x) { d->duom = x;}

//-----------------Naujas elementas dedamas i pradzia----------

void Fondas::DetiIPradzia(Mokslininkai x)

{

if (pradzia == NULL || pabaiga == NULL){

pradzia = new Mazgas;

pabaiga = new Mazgas;

d = pradzia;

Mazgas \*t = new Mazgas;

pradzia->pirmyn = t;

t->pirmyn = pabaiga;

pabaiga->atgal = t;

t->atgal = pradzia;

t->duom = x;

}

else{

Mazgas \*t = new Mazgas;

t->duom = x;

t->atgal = pradzia;

(pradzia->pirmyn)->atgal = t;

t->pirmyn = pradzia->pirmyn;

pradzia->pirmyn = t;

}

}

//----------------Naujas elementas dedamas i pabaiga-----------

void Fondas::DetiIPabaiga(Mokslininkai x)

{

if (pradzia == NULL || pabaiga == NULL){

pradzia = new Mazgas;

pabaiga = new Mazgas;

d = pradzia;

Mazgas \*t = new Mazgas;

pradzia->pirmyn = t;

t->pirmyn = pabaiga;

pabaiga->atgal = t;

t->atgal = pradzia;

t->duom = x;

}

else{

Mazgas \*t = new Mazgas;

t->duom = x;

t->pirmyn = pabaiga;

(pabaiga->atgal)->pirmyn = t;

t->atgal = pabaiga->atgal;

pabaiga->atgal = t;

}

}

//--------------------------------------------------------\*/

void Fondas::DetiP(double p, int i)

{

Premija[i] = p;

}

//--------------------------------------------------------

double Fondas::ImtiP(int i)

{

return Premija[i];

}

//--------------------------------------------------------

void Fondas::Paskirstymas() {

double indS = 0;

double pinigai;

for (int i = 0; i < Cpremijos; i++){

indS = 0;

Pradzia();

while (Yra()){

indS+= ImtiDuomenis().ImtiIndelius(i);

Sekantis();

}

Pradzia();

while (Yra()){

pinigai = (ImtiDuomenis().ImtiIndelius(i) / indS) \* Premija[i];

Mokslininkai x;

x = Mokslininkai(ImtiDuomenis());

x.DetiUzdirbius(pinigai, i);

DetiDuomenis(x);

Sekantis();

}

}

Pradzia();

while (Yra()){

Mokslininkai x;

x = Mokslininkai(ImtiDuomenis());

x.ApskaiciuotiS();

DetiDuomenis(x);

Sekantis();

}

}

//-------------------------------------------------------

void Fondas::Rikiavimas()

{

bool yra = true;

Mazgas \*d, \*d1, \*r;

while (yra) {

yra = false;

d = d1 = r = pradzia->pirmyn;

while (d1 != pabaiga) {

if (d1->duom > d->duom) {

yra = true;

if (d == pradzia->pirmyn) { // pirmas elementas su kaimynu

pradzia->pirmyn = (pradzia->pirmyn)->pirmyn;

d->pirmyn = d1->pirmyn;

d1->pirmyn = d;

}

else { // pora tolesnių elementų

d->pirmyn = d1->pirmyn;

d1->pirmyn = d;

r->pirmyn = d1;

}

}

r = d;

d = d1;

d1 = d1->pirmyn;

}

}

}

**Form1.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

#include"Mokslininkai.h"

#include"Fondas.h"

#include"Apie\_Dialogas.h"

const string Cduom = "Duomenys.txt";

const string Crez = "Rezultatai.txt";

const string Crez2= "Rezultatai2.txt";

const string Crez3= "Rezultatai3.txt";

constdouble Cprem = 4;

namespace S2L1 {

usingnamespace System;

usingnamespace System::ComponentModel;

usingnamespace System::Collections;

usingnamespace System::Windows::Forms;

usingnamespace System::Data;

usingnamespace System::Drawing;

void StringTostring(String ^ s, string & os)

{

usingnamespace Runtime::InteropServices;

constchar\* chars = (constchar\*)(Marshal::StringToHGlobalAnsi(s)).ToPointer();

os = chars;

Marshal::FreeHGlobal(IntPtr((void\*)chars));

}

void Duomenys(Fondas \* F, string dfvs)

{

ifstream duom(dfvs.c\_str());

string kod, pav, vard;

double indelis[Mokslininkai::Cpremijos];

double p[Fondas::Cpremijos];

for (int i = 0; i < Fondas::Cpremijos; i++){

duom >> p[i];

F->DetiP(p[i], i);

}

while(!duom.eof()) {

getline (duom, kod, ',');

getline (duom, pav, ',');

getline (duom, vard, ',');

for (int i=0; i<Mokslininkai::Cpremijos; i++)

duom >> indelis[i];

Mokslininkai x;

x = Mokslininkai(kod, pav, vard, indelis);

F->DetiIPabaiga(x);

}

duom.close();

}

//-----------------------------------------------------

// Spausdinami objekto F duomenys

void Isvesti(Fondas \* F)

{

ofstream rez(Crez.c\_str(), ios::app);

for (int i = 0; i < Fondas::Cpremijos; i++)

rez <<' '<< F->ImtiP(i);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(15) <<"Indeliai \n";

rez <<"-----------------------------------------\n";

F->Pradzia();

for( F->Pradzia(); F->Yra(); F->Sekantis())

rez << F->ImtiDuomenis().Spausdinti();

rez << endl <<"--------------------------------------\n\n\n";

rez.close();

}

//-----------------------------------------------------

void Apskaiciavimas(Fondas \* F, Fondas \* R)

{

for (int i = 0; i < Fondas::Cpremijos; i++)

R->DetiP(F->ImtiP(i), i);

F->Pradzia();

while (F->Yra()){

R->DetiIPradzia(F->ImtiDuomenis());

F->Sekantis();

}

R->Paskirstymas();

R->Rikiavimas();

}

//-----------------------------------------------------

void RezultatuIsvedimas(Fondas \* R, string rezFailas)

{

ofstream rez(rezFailas.c\_str(), ios::app);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(20) <<"Indeliai"<<

setw(30) <<"Premijos"<< setw(10) <<"Suma"<< endl;

rez <<"-----------------------------------------------------------------------------------------\n";

R->Pradzia();

while (R->Yra()){

rez << R->ImtiDuomenis().SpausdintiPapildomai();

R->Sekantis();

}

rez << endl;

rez <<"-----------------------------------------------------------------------------------------\n";

rez.close();

}

//----------------------------------------------------

double Vidurkiai(Fondas \* R, Fondas \* U)

{

for (int i = 0; i < Fondas::Cpremijos; i++)

U->DetiP(R->ImtiP(i), i);

R->Pradzia();

double PremijuV = 0;

int n = 0;

while (R->Yra()){

PremijuV += R->ImtiDuomenis().ImtiS();

R->Sekantis();

n++;

}

R->Pradzia();

PremijuV = PremijuV / n;

while(R->Yra()){

if (R->ImtiDuomenis().ImtiS() < PremijuV)

U->DetiIPabaiga(R->ImtiDuomenis());

R->Sekantis();

}

return PremijuV;

}

//-----------------------------------------------------

/// <summary>

/// Summary for Form1

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

publicrefclass Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

info = new Fondas;

infoc = new Fondas;

infoc2 = new Fondas;

dialog = gcnew Apie\_Dialogas;

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

delete info;

delete infoc;

delete infoc2;

delete dialog;

}

private: Fondas \*info, \*infoc, \*infoc2;

private: Apie\_Dialogas ^dialog;

private: System::Windows::Forms::MenuStrip^ menuStrip1;

protected:

private: System::Windows::Forms::ToolStripMenuItem^ nuskaitytiToolStripMenuItem;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::RichTextBox^ richTextBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::RichTextBox^ richTextBox3;

private: System::Windows::Forms::ToolStripMenuItem^ skaičiuotiToolStripMenuItem;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::ToolStripMenuItem^ vidurkisToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ apieToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ nuskaitytiToolStripMenuItem1;

private: System::Windows::Forms::OpenFileDialog^ openFileDialog1;

private: System::Windows::Forms::SaveFileDialog^ saveFileDialog1;

private: System::Windows::Forms::ToolStripMenuItem^ išeitiToolStripMenuItem1;

private: System::Windows::Forms::ToolStripSeparator^ toolStripSeparator1;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container ^components;

#pragmaregion Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());

this->nuskaitytiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->nuskaitytiToolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->toolStripSeparator1 = (gcnew System::Windows::Forms::ToolStripSeparator());

this->išeitiToolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->skaičiuotiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->vidurkisToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->apieToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->richTextBox2 = (gcnew System::Windows::Forms::RichTextBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->richTextBox3 = (gcnew System::Windows::Forms::RichTextBox());

this->label4 = (gcnew System::Windows::Forms::Label());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->openFileDialog1 = (gcnew System::Windows::Forms::OpenFileDialog());

this->saveFileDialog1 = (gcnew System::Windows::Forms::SaveFileDialog());

this->menuStrip1->SuspendLayout();

this->SuspendLayout();

//

// menuStrip1

//

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(4) {this->nuskaitytiToolStripMenuItem,

this->skaičiuotiToolStripMenuItem, this->vidurkisToolStripMenuItem, this->apieToolStripMenuItem});

this->menuStrip1->Location = System::Drawing::Point(0, 0);

this->menuStrip1->Name = L"menuStrip1";

this->menuStrip1->Size = System::Drawing::Size(764, 24);

this->menuStrip1->TabIndex = 0;

this->menuStrip1->Text = L"menuStrip1";

this->menuStrip1->ItemClicked += gcnew System::Windows::Forms::ToolStripItemClickedEventHandler(this, &Form1::menuStrip1\_ItemClicked);

//

// nuskaitytiToolStripMenuItem

//

this->nuskaitytiToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(3) {this->nuskaitytiToolStripMenuItem1,

this->toolStripSeparator1, this->išeitiToolStripMenuItem1});

this->nuskaitytiToolStripMenuItem->Name = L"nuskaitytiToolStripMenuItem";

this->nuskaitytiToolStripMenuItem->Size = System::Drawing::Size(48, 20);

this->nuskaitytiToolStripMenuItem->Text = L"Failas";

this->nuskaitytiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::nuskaitytiToolStripMenuItem\_Click);

//

// nuskaitytiToolStripMenuItem1

//

this->nuskaitytiToolStripMenuItem1->Name = L"nuskaitytiToolStripMenuItem1";

this->nuskaitytiToolStripMenuItem1->Size = System::Drawing::Size(152, 22);

this->nuskaitytiToolStripMenuItem1->Text = L"Nuskaityti...";

this->nuskaitytiToolStripMenuItem1->Click += gcnew System::EventHandler(this, &Form1::nuskaitytiToolStripMenuItem1\_Click);

//

// toolStripSeparator1

//

this->toolStripSeparator1->Name = L"toolStripSeparator1";

this->toolStripSeparator1->Size = System::Drawing::Size(149, 6);

//

// išeitiToolStripMenuItem1

//

this->išeitiToolStripMenuItem1->Name = L"išeitiToolStripMenuItem1";

this->išeitiToolStripMenuItem1->Size = System::Drawing::Size(152, 22);

this->išeitiToolStripMenuItem1->Text = L"Išeiti";

this->išeitiToolStripMenuItem1->Click += gcnew System::EventHandler(this, &Form1::išeitiToolStripMenuItem1\_Click);

//

// skaičiuotiToolStripMenuItem

//

this->skaičiuotiToolStripMenuItem->Enabled = false;

this->skaičiuotiToolStripMenuItem->Name = L"skaičiuotiToolStripMenuItem";

this->skaičiuotiToolStripMenuItem->Size = System::Drawing::Size(70, 20);

this->skaičiuotiToolStripMenuItem->Text = L"Skaičiuoti";

this->skaičiuotiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::skaičiuotiToolStripMenuItem\_Click);

//

// vidurkisToolStripMenuItem

//

this->vidurkisToolStripMenuItem->Enabled = false;

this->vidurkisToolStripMenuItem->Name = L"vidurkisToolStripMenuItem";

this->vidurkisToolStripMenuItem->Size = System::Drawing::Size(61, 20);

this->vidurkisToolStripMenuItem->Text = L"Vidurkis";

this->vidurkisToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::vidurkisToolStripMenuItem\_Click);

//

// apieToolStripMenuItem

//

this->apieToolStripMenuItem->Name = L"apieToolStripMenuItem";

this->apieToolStripMenuItem->Size = System::Drawing::Size(43, 20);

this->apieToolStripMenuItem->Text = L"Apie";

this->apieToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::apieToolStripMenuItem\_Click);

//

// richTextBox1

//

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox1->Location = System::Drawing::Point(0, 59);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->Size = System::Drawing::Size(764, 191);

this->richTextBox1->TabIndex = 1;

this->richTextBox1->Text = L"";

this->richTextBox1->TextChanged += gcnew System::EventHandler(this, &Form1::richTextBox1\_TextChanged);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label1->Location = System::Drawing::Point(12, 33);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(99, 23);

this->label1->TabIndex = 2;

this->label1->Text = L"Duomenys";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label2->Location = System::Drawing::Point(12, 253);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(95, 23);

this->label2->TabIndex = 4;

this->label2->Text = L"Rezultatai";

//

// richTextBox2

//

this->richTextBox2->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox2->Location = System::Drawing::Point(0, 279);

this->richTextBox2->Name = L"richTextBox2";

this->richTextBox2->Size = System::Drawing::Size(764, 196);

this->richTextBox2->TabIndex = 3;

this->richTextBox2->Text = L"";

this->richTextBox2->TextChanged += gcnew System::EventHandler(this, &Form1::richTextBox2\_TextChanged);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label3->Location = System::Drawing::Point(12, 478);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(336, 23);

this->label3->TabIndex = 6;

this->label3->Text = L"Mažai uždirbančių mokslininkų sąrašas";

//

// richTextBox3

//

this->richTextBox3->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox3->Location = System::Drawing::Point(0, 504);

this->richTextBox3->Name = L"richTextBox3";

this->richTextBox3->Size = System::Drawing::Size(764, 138);

this->richTextBox3->TabIndex = 5;

this->richTextBox3->Text = L"";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label4->Location = System::Drawing::Point(491, 478);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(88, 23);

this->label4->TabIndex = 7;

this->label4->Text = L"Vidurkis:";

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(585, 482);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(127, 20);

this->textBox1->TabIndex = 8;

this->textBox1->TextChanged += gcnew System::EventHandler(this, &Form1::textBox1\_TextChanged);

//

// openFileDialog1

//

this->openFileDialog1->FileName = L"openFileDialog1";

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(764, 644);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->richTextBox3);

this->Controls->Add(this->label2);

this->Controls->Add(this->richTextBox2);

this->Controls->Add(this->label1);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->menuStrip1);

this->MainMenuStrip = this->menuStrip1;

this->Name = L"Form1";

this->Text = L"Programa";

this->Load += gcnew System::EventHandler(this, &Form1::Form1\_Load);

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragmaendregion

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void išeitiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void nuskaitytiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void richTextBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void menuStrip1\_ItemClicked(System::Object^ sender, System::Windows::Forms::ToolStripItemClickedEventArgs^ e) {

}

private: System::Void richTextBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void skaičiuotiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Apskaiciavimas(info, infoc);

ofstream rez2(Crez2.c\_str());

rez2.close();

RezultatuIsvedimas(infoc, Crez2);

vidurkisToolStripMenuItem->Enabled = true;

richTextBox2->LoadFile("Rezultatai2.txt", RichTextBoxStreamType::PlainText);

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void vidurkisToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

double vidurkis = Vidurkiai(infoc, infoc2);

ofstream rez3(Crez3.c\_str());

rez3.close();

RezultatuIsvedimas(infoc2, Crez3);

richTextBox3->LoadFile("Rezultatai3.txt", RichTextBoxStreamType::PlainText);

textBox1->Text = ((int)vidurkis).ToString();

}

private: System::Void apieToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

::DialogResult result = dialog->ShowDialog(this);

}

private: System::Void išeitiToolStripMenuItem1\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void nuskaitytiToolStripMenuItem1\_Click(System::Object^ sender, System::EventArgs^ e) {

openFileDialog1->DefaultExt = ".txt";

openFileDialog1->Filter = "TXT failai|\*.txt";

if (openFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK &&

openFileDialog1->FileName->Length > 0) {

String^ dfv;

dfv = openFileDialog1->FileName;

richTextBox1->LoadFile(dfv, RichTextBoxStreamType::PlainText);

string dfvs;

StringTostring(dfv, dfvs);

ofstream rez(Crez.c\_str());

rez.close();

Duomenys(info, dfvs);

Isvesti(info);

richTextBox1->LoadFile("Rezultatai.txt", RichTextBoxStreamType::PlainText);

skaičiuotiToolStripMenuItem->Enabled = true;

}

}

private: System::Void išsaugotiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

// Inicializuoja saveFileDialog vaizdą

saveFileDialog1->DefaultExt = ".txt";

saveFileDialog1->Filter = "TXT failai|\*.txt";

// Patikrina, ar vartotojas pasirinko failą iš SaveFileDialog?

if (saveFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK &&

saveFileDialog1->FileName->Length > 0) {

String^ rfv; // rezultatų failo vardas (String ^)

rfv = saveFileDialog1->FileName;

// Rezultatų failo vardas rfv (Sting ^) konvertuojamas į rfvs (string)

string rfvs; // rezultatų failo vardas (string)

StringTostring(rfv, rfvs);

// Sukuriamas tuščias rezultatų failas

ofstream fr(rfvs.c\_str());

fr.close();

// Atliekami veiksmai su failu rfvs (pvz.: rezultatų spausdinimas)

// ...

}

}

};

}

## Pradiniai duomenys ir rezultatai

**Duomenys.txt:**

5000 2500 8000 1000

38604163754, Hhhhhh, Hhhhhhhhh, 5 9 7 2

37407011234, Gggggg, Ggggggggg, 2 1 0 3

39012128243, Ffffff, Fffffffff, 6 7 1 9

36401288430, Eeeeee, Eeeeeeeee, 1 2 3 4

37702131348, Dddddd, Ddddddddd, 9 4 3 6

37103044686, Cccccc, Ccccccccc, 4 5 4 6

33008141568, Aaaaaa, Aaaaaaaaa, 7 2 2 3

39011128434, Bbbbbb, Bbbbbbbbb, 7 6 1 9

**Rezultatai.txt:**

5000 2500 8000 1000

Kodas Pavarde Vardas Indeliai

-----------------------------------------

38604163754 HhhhhhHhhhhhhhh 5 9 7 2

37407011234 GgggggGgggggggg 2 1 0 3

39012128243 FfffffFffffffff 6 7 1 9

36401288430 EeeeeeEeeeeeeee 1 2 3 4

37702131348 DdddddDdddddddd 9 4 3 6

37103044686 CcccccCcccccccc 4 5 4 6

33008141568 AaaaaaAaaaaaaaa 7 2 2 3

39011128434 BbbbbbBbbbbbbbb 7 6 1 9

--------------------------------------

**Rezultatai2.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

37103044686 CcccccCcccccccc 4 5 4 6 487 347 1523 142 2501

37702131348 DdddddDdddddddd 9 4 3 6 1097 277 1142 142 2661

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

38604163754 HhhhhhHhhhhhhhh 5 9 7 2 609 625 2666 47 3949

-----------------------------------------------------------------------------------------

**Rezultatai3.txt:**

**Kodas Pavarde Vardas Indeliai Premijos Suma**

**-----------------------------------------------------------------------------------------**

**33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825**

**39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865**

**36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498**

**39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813**

**37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384**

**-----------------------------------------------------------------------------------------**

# Objektų sąrašas (L4)

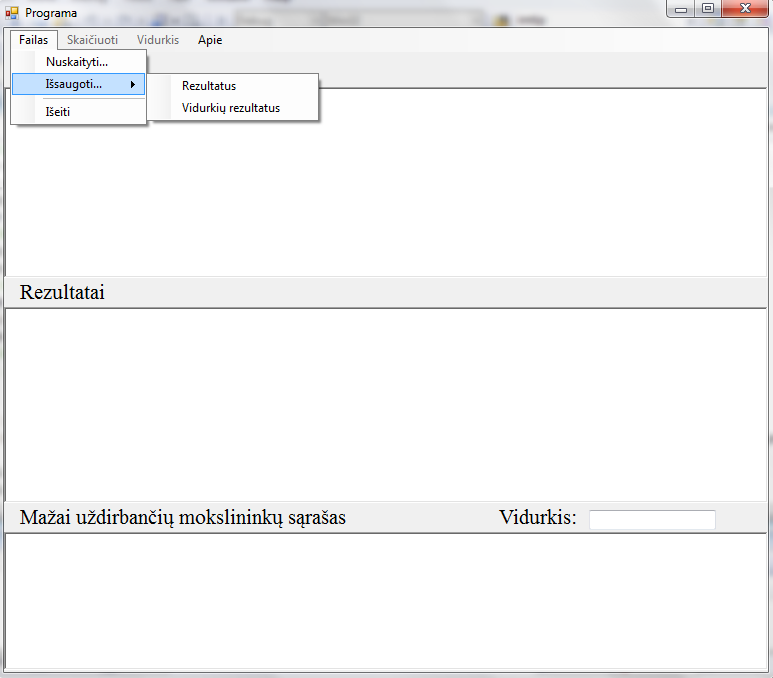
## Darbo užduotis

39. Moksliniai darbuotojai atliko darbus 4 skirtingose temose. Visos temos gavo premijas. Duomenų faile tokia informacija:

Pirmoje eilutėje – premijų dydžiai.

Tolesnėse eilutėse – darbuotojų indėliai, kurie išreikšti naudingumo koeficientu, į eilinę temą. Informacija apie darbuotoją užima vieną eilutę. Remiantis darbuotojų indėliais, suskaičiuoti kiekvienam darbuotojui priklausančios premijos dydį pagal kiekvieną temą atskirai ir bendrą premijų sumą. Suformuoti sąrašą darbuotojų, kurie uždirbo mažiau už vidurkį. Rikiuoti(pavardė, vardas).

## Grafinės vartotojo sąsajos schema



## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Meniu juosta  (ToolStripMenu) | Rodo, kokias funkcijas atlieka programa | Ja pasinaudojus galima nuskaityti duomenis, juos apsikačiuoti, išvesti vidurkius, baigti darbą su programa |
| Tekstinių failų laukeliai (richtextbox1, richtextbox2, richtextbox3) | Pavaizduoja atitinkamų tekstinių failų turinį | Parodo nuskaitytų pradinių duomenų failą, rezultatų failą su skaičiavimais ir rikiavimu. |
| Vidurkio reikšmės laukelis (textbox1) | Pavaizduoja, koks buvo vidurkis, pagal kurį buvo atrinktas mažai uždirbančių mokslininkų sąrašas | Padeda vartotojui lengviau įvertinti duomenis |
| Laukelių pavadinimai (label1, label2, label3, label4) | Skirti pavaizduoti antraštėms ar tekstui programos lauke | Skirta vartotojui atskirti duomenų, rezultatų failą, taip pat atskirti atrinktą rezultatų failą bei nurodant textbox1 reikšmę programai. |
| Dialogo langas  (Apie\_Dialogas) | Skirtas pateikti svarbią informaciją vartotojui | Skirtas bendrai informacijai apie programos pavadinimą, kūrėją ir datą |

## Programos vartotojo vadovas

1. Pasirinkite failą, kurį norite nuskaityti, paspaudę mygtuką Failas->Nuskaityti...

2. Atsilaisvinus mygtukui „Skaičiuoti“, spauskite jį, norėdami apskaičiuoti kiekvieno mokslininko uždirbtą premiją bei visas premijas kartu.

3. Spauskite mygtuką „Vidurkis“, jei norite apskaičiuoti premijų vidurkį

3-iajame teksto lauke bus pavaizduotas sąrašas mokslininkų, kurių premijų suma nesiekė vidurkio.

4. Mygtukas „Apie“ pateikia bendrąją informaciją apie programą.

5. Spauskite Failas->Išsaugoti->Rezultatus norėdami išsaugoti rezultatų .txt failą. Spauskite   
Failas->Išsaugoti->Vidurkių\_Rezultatus norėdami išsaugoti vidurkių rezultatų .txt failą.

5. Norėdami baigti darbą, spauskite „Išeiti“.

## Programos tekstas

**Mokslininkai.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Mokslininkai

{

public:

staticconstint Cpremijos=4;

private:

string kodas, vardas, pavarde;

double ind[Cpremijos], pin[Cpremijos] , S;

public:

Mokslininkai() {}

Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]);

string Spausdinti();

string SpausdintiPapildomai();

void ApskaiciuotiS();

void DetiUzdirbius(double a, int b);

double ImtiS();

double ImtiIndelius(int i);

booloperator> (constMokslininkai & C);

};

**Mokslininkai.cpp:**

#include"StdAfx.h"

#include"Mokslininkai.h"

usingnamespace std;

Mokslininkai::Mokslininkai(string kodasR, string vardasR, string pavardeR, double indR[]) :

kodas(kodasR), vardas(vardasR), pavarde(pavardeR), S(0)

{

for (int i = 0; i < Cpremijos; i++){

ind[i] = indR[i];

pin[i] = 0;

}

}

string Mokslininkai::Spausdinti()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

return ss.str();

}

string Mokslininkai::SpausdintiPapildomai()

{

stringstream ss;

ss << left << setw(15) << kodas << setw(15) << vardas << setw(15) << pavarde;

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

ss << right << setw(4) << ind[i];

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

ss <<' '<< right << setw(6) << ((int)pin[i]);

ss << right << setw(10) << ((int)S);

return ss.str();

}

//----------------------------------

double Mokslininkai::ImtiS() { return S; }

//----------------------------------

double Mokslininkai::ImtiIndelius(int i)

{

return ind[i];

}

//----------------------------------

void Mokslininkai::ApskaiciuotiS(){

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

S+= pin[i];

}

//----------------------------------

void Mokslininkai::DetiUzdirbius(double a, int b)

{

pin[b] = a;

}

//------------------------------------

bool Mokslininkai::operator> (const Mokslininkai & C)

{

return pavarde < C.pavarde || ((pavarde == C.pavarde) && (vardas < C.vardas));

}

**Elementas.h:**

#pragmaonce

#include"Mokslininkai.h"

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

class Elementas

{

private:

Mokslininkai duom;

Elementas \*pirmyn;

Elementas \*atgal;

public:

Elementas();

~Elementas();

void DetiDuom(Mokslininkai x);

void DetiPirmyn(Elementas \* p);

void DetiAtgal(Elementas \*a);

Mokslininkai ImtiDuom();

Elementas \* ImtiPirmyn();

Elementas \* ImtiAtgal();

};

**Elementas.cpp:**

#include"StdAfx.h"

#include"Elementas.h"

usingnamespace std;

Elementas::Elementas() : duom(Mokslininkai()), pirmyn(NULL), atgal(NULL) {}

Elementas::~Elementas() {}

void Elementas::DetiDuom(Mokslininkai x) { duom = x; }

void Elementas::DetiPirmyn(Elementas \* p) { pirmyn = p; }

void Elementas::DetiAtgal(Elementas \*a) { atgal = a; }

Mokslininkai Elementas::ImtiDuom() {return duom; }

Elementas \* Elementas::ImtiPirmyn() {return pirmyn; }

Elementas \* Elementas::ImtiAtgal() {return atgal; }

**Fondas.h:**

#pragmaonce

#include"Elementas.h"

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

usingnamespace std;

class Fondas

{

public:

staticconstint Cpremijos=4;

private:

double Premija[Cpremijos];

Elementas \*pradzia;

Elementas \*pabaiga;

Elementas \*d;

public:

Fondas();

~Fondas();

void Pradzia();

void Pabaiga();

void Sekantis();

void Ankstesnis();

bool Yra();

Mokslininkai ImtiDuomenis();

void DetiDuomenis(Mokslininkai x);

void DetiIPradzia(Mokslininkai x);

void DetiIPabaiga(Mokslininkai x);

void DetiP(double p, int i);

double ImtiP (int i);

void Paskirstymas();

void Rikiavimas();

};

**Fondas.cpp:**

#include"StdAfx.h"

#include"Fondas.h"

usingnamespace std;

Fondas::Fondas() : pradzia(NULL), pabaiga(NULL), d(NULL) {}

Fondas::~Fondas()

{

Elementas \*pap;

while (pradzia != pabaiga){

pap = pradzia;

pradzia = pradzia->ImtiPirmyn();

delete pap;

}

d = NULL;

}

//-----------------Baziniai saraso veiksmai---------------------

void Fondas::Pradzia() { d = pradzia->ImtiPirmyn(); }

void Fondas::Pabaiga() { d = pabaiga->ImtiAtgal(); }

void Fondas::Sekantis() {d = d->ImtiPirmyn();}

void Fondas::Ankstesnis() {d = d->ImtiAtgal();}

bool Fondas::Yra() {return d != pabaiga; }

Mokslininkai Fondas::ImtiDuomenis() {return d->ImtiDuom(); }

void Fondas::DetiDuomenis(Mokslininkai x) { d->DetiDuom(x);}

//-----------------Naujas elementas dedamas i pradzia----------

void Fondas::DetiIPradzia(Mokslininkai x)

{

if (pradzia == NULL || pabaiga == NULL){

pradzia = new Elementas;

pabaiga = new Elementas;

d = pradzia;

Elementas \*t = new Elementas;

pradzia->DetiPirmyn(t);

t->DetiPirmyn(pabaiga);

pabaiga->DetiAtgal(t);

t->DetiAtgal(pradzia);

t->DetiDuom(x);

}

else{

Elementas \*t = new Elementas;

t->DetiDuom(x);

t->DetiAtgal(pradzia);

(pradzia->ImtiPirmyn())->DetiAtgal(t);

t->DetiPirmyn(pradzia->ImtiPirmyn());

pradzia->DetiPirmyn(t);

}

}

//----------------Naujas elementas dedamas i pabaiga-----------

void Fondas::DetiIPabaiga(Mokslininkai x)

{

if (pradzia == NULL || pabaiga == NULL){

pradzia = new Elementas;

pabaiga = new Elementas;

d = pradzia;

Elementas \*t = new Elementas;

pradzia->DetiPirmyn(t);

t->DetiPirmyn(pabaiga);

pabaiga->DetiAtgal(t);

t->DetiAtgal(pradzia);

t->DetiDuom(x);

}

else{

Elementas \*t = new Elementas;

t->DetiDuom(x);

t->DetiPirmyn(pabaiga);

(pabaiga->ImtiAtgal())->DetiPirmyn(t);

t->DetiAtgal(pabaiga->ImtiAtgal());

pabaiga->DetiAtgal(t);

}

}

//--------------------------------------------------------\*/

void Fondas::DetiP(double p, int i)

{

Premija[i] = p;

}

//--------------------------------------------------------

double Fondas::ImtiP(int i)

{

return Premija[i];

}

//--------------------------------------------------------

void Fondas::Paskirstymas() {

double indS = 0;

double pinigai;

for (int i = 0; i < Cpremijos; i++){

indS = 0;

Pradzia();

while (Yra()){

indS+= ImtiDuomenis().ImtiIndelius(i);

Sekantis();

}

Pradzia();

while (Yra()){

pinigai = (ImtiDuomenis().ImtiIndelius(i) / indS) \* Premija[i];

Mokslininkai x;

x = Mokslininkai(ImtiDuomenis());

x.DetiUzdirbius(pinigai, i);

DetiDuomenis(x);

Sekantis();

}

}

Pradzia();

while (Yra()){

Mokslininkai x;

x = Mokslininkai(ImtiDuomenis());

x.ApskaiciuotiS();

DetiDuomenis(x);

Sekantis();

}

}

//-------------------------------------------------------

void Fondas::Rikiavimas()

{

bool yra = true;

Elementas \*d, \*d1, \*r;

while (yra) {

yra = false;

d = d1 = r = pradzia->ImtiPirmyn();

while (d1 != pabaiga) {

if (d1->ImtiDuom() > d->ImtiDuom()) {

yra = true;

if (d == pradzia->ImtiPirmyn()) { // pirmas elementas su kaimynu

pradzia->DetiPirmyn((pradzia->ImtiPirmyn())->ImtiPirmyn());

d->DetiPirmyn(d1->ImtiPirmyn());

d1->DetiPirmyn(d);

}

else { // pora tolesnių elementų

d->DetiPirmyn(d1->ImtiPirmyn());

d1->DetiPirmyn(d);

r->DetiPirmyn(d1);

}

}

r = d;

d = d1;

d1 = d1->ImtiPirmyn();

}

}

}

**Form1.h:**

#pragmaonce

#include<string>

#include<iostream>

#include<fstream>

#include<sstream>

#include<iomanip>

#include"Mokslininkai.h"

#include"Fondas.h"

#include"Apie\_Dialogas.h"

const string Cduom = "Duomenys.txt";

const string Crez = "Rezultatai.txt";

const string Crez2= "Rezultatai2.txt";

const string Crez3= "Rezultatai3.txt";

constdouble Cprem = 4;

namespace S2L1 {

usingnamespace System;

usingnamespace System::ComponentModel;

usingnamespace System::Collections;

usingnamespace System::Windows::Forms;

usingnamespace System::Data;

usingnamespace System::Drawing;

void StringTostring(String ^ s, string & os)

{

usingnamespace Runtime::InteropServices;

constchar\* chars = (constchar\*)(Marshal::StringToHGlobalAnsi(s)).ToPointer();

os = chars;

Marshal::FreeHGlobal(IntPtr((void\*)chars));

}

void Duomenys(Fondas \* F, string dfvs)

{

ifstream duom(dfvs.c\_str());

string kod, pav, vard;

double indelis[Mokslininkai::Cpremijos];

double p[Fondas::Cpremijos];

for (int i = 0; i < Fondas::Cpremijos; i++){

duom >> p[i];

F->DetiP(p[i], i);

}

while(!duom.eof()) {

getline (duom, kod, ',');

getline (duom, pav, ',');

getline (duom, vard, ',');

for (int i=0; i<Mokslininkai::Cpremijos; i++)

duom >> indelis[i];

Mokslininkai x;

x = Mokslininkai(kod, pav, vard, indelis);

F->DetiIPabaiga(x);

}

duom.close();

}

//-----------------------------------------------------

// Spausdinami objekto F duomenys

void Isvesti(Fondas \* F)

{

ofstream rez(Crez.c\_str());

for (int i = 0; i < Fondas::Cpremijos; i++)

rez <<' '<< F->ImtiP(i);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(15) <<"Indeliai \n";

rez <<"-----------------------------------------\n";

F->Pradzia();

for( F->Pradzia(); F->Yra(); F->Sekantis())

rez << F->ImtiDuomenis().Spausdinti();

rez << endl <<"--------------------------------------\n\n\n";

rez.close();

}

//-----------------------------------------------------

void Apskaiciavimas(Fondas \* F, Fondas \* R)

{

for (int i = 0; i < Fondas::Cpremijos; i++)

R->DetiP(F->ImtiP(i), i);

F->Pradzia();

while (F->Yra()){

R->DetiIPradzia(F->ImtiDuomenis());

F->Sekantis();

}

R->Paskirstymas();

R->Rikiavimas();

}

//-----------------------------------------------------

void RezultatuIsvedimas(Fondas \* R, string rezFailas)

{

ofstream rez(rezFailas.c\_str(), ios::app);

rez << endl << left << setw(15) <<"Kodas"<< setw(15) <<"Pavarde"<< setw(15) <<"Vardas"<< setw(20) <<"Indeliai"<<

setw(30) <<"Premijos"<< setw(10) <<"Suma"<< endl;

rez <<"-----------------------------------------------------------------------------------------\n";

R->Pradzia();

while (R->Yra()){

rez << R->ImtiDuomenis().SpausdintiPapildomai();

R->Sekantis();

}

rez << endl;

rez <<"-----------------------------------------------------------------------------------------\n";

rez.close();

}

//----------------------------------------------------

double Vidurkiai(Fondas \* R, Fondas \* U)

{

for (int i = 0; i < Fondas::Cpremijos; i++)

U->DetiP(R->ImtiP(i), i);

R->Pradzia();

double PremijuV = 0;

int n = 0;

while (R->Yra()){

PremijuV += R->ImtiDuomenis().ImtiS();

R->Sekantis();

n++;

}

R->Pradzia();

PremijuV = PremijuV / n;

while(R->Yra()){

if (R->ImtiDuomenis().ImtiS() < PremijuV)

U->DetiIPabaiga(R->ImtiDuomenis());

R->Sekantis();

}

return PremijuV;

}

void NuskaitymoVeiksmai(ToolStripMenuItem^ skaičiuotiToolStripMenuItem, RichTextBox^ richTextBox1, Fondas \* info, string p)

{

Duomenys(info, p);

Isvesti(info);

richTextBox1->LoadFile("Rezultatai.txt", RichTextBoxStreamType::PlainText);

skaičiuotiToolStripMenuItem->Enabled = true;

}

//-----------------------------------------------------

/// <summary>

/// Summary for Form1

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

publicrefclass Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

info = new Fondas;

infoc = new Fondas;

infoc2 = new Fondas;

dialog = gcnew Apie\_Dialogas;

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

delete info;

delete infoc;

delete infoc2;

delete dialog;

}

private: Fondas \*info, \*infoc, \*infoc2;

private: Apie\_Dialogas ^dialog;

private: System::Windows::Forms::MenuStrip^ menuStrip1;

protected:

private: System::Windows::Forms::ToolStripMenuItem^ nuskaitytiToolStripMenuItem;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::RichTextBox^ richTextBox2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::RichTextBox^ richTextBox3;

private: System::Windows::Forms::ToolStripMenuItem^ skaičiuotiToolStripMenuItem;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::ToolStripMenuItem^ vidurkisToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ apieToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ nuskaitytiToolStripMenuItem1;

private: System::Windows::Forms::OpenFileDialog^ openFileDialog1;

private: System::Windows::Forms::SaveFileDialog^ saveFileDialog1;

private: System::Windows::Forms::ToolStripMenuItem^ išeitiToolStripMenuItem1;

private: System::Windows::Forms::ToolStripSeparator^ toolStripSeparator1;

private: System::Windows::Forms::ToolStripMenuItem^ išsaugotiToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ rezultatusToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ vidurkiųRezultatusToolStripMenuItem;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container ^components;

#pragmaregion Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());

this->nuskaitytiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->nuskaitytiToolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->išsaugotiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->rezultatusToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->vidurkiųRezultatusToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->toolStripSeparator1 = (gcnew System::Windows::Forms::ToolStripSeparator());

this->išeitiToolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->skaičiuotiToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->vidurkisToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->apieToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->richTextBox2 = (gcnew System::Windows::Forms::RichTextBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->richTextBox3 = (gcnew System::Windows::Forms::RichTextBox());

this->label4 = (gcnew System::Windows::Forms::Label());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->openFileDialog1 = (gcnew System::Windows::Forms::OpenFileDialog());

this->saveFileDialog1 = (gcnew System::Windows::Forms::SaveFileDialog());

this->menuStrip1->SuspendLayout();

this->SuspendLayout();

//

// menuStrip1

//

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(4) {this->nuskaitytiToolStripMenuItem,

this->skaičiuotiToolStripMenuItem, this->vidurkisToolStripMenuItem, this->apieToolStripMenuItem});

this->menuStrip1->Location = System::Drawing::Point(0, 0);

this->menuStrip1->Name = L"menuStrip1";

this->menuStrip1->Size = System::Drawing::Size(764, 24);

this->menuStrip1->TabIndex = 0;

this->menuStrip1->Text = L"menuStrip1";

this->menuStrip1->ItemClicked += gcnew System::Windows::Forms::ToolStripItemClickedEventHandler(this, &Form1::menuStrip1\_ItemClicked);

//

// nuskaitytiToolStripMenuItem

//

this->nuskaitytiToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(4) {this->nuskaitytiToolStripMenuItem1,

this->išsaugotiToolStripMenuItem, this->toolStripSeparator1, this->išeitiToolStripMenuItem1});

this->nuskaitytiToolStripMenuItem->Name = L"nuskaitytiToolStripMenuItem";

this->nuskaitytiToolStripMenuItem->Size = System::Drawing::Size(48, 20);

this->nuskaitytiToolStripMenuItem->Text = L"Failas";

this->nuskaitytiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::nuskaitytiToolStripMenuItem\_Click);

//

// nuskaitytiToolStripMenuItem1

//

this->nuskaitytiToolStripMenuItem1->Name = L"nuskaitytiToolStripMenuItem1";

this->nuskaitytiToolStripMenuItem1->Size = System::Drawing::Size(152, 22);

this->nuskaitytiToolStripMenuItem1->Text = L"Nuskaityti...";

this->nuskaitytiToolStripMenuItem1->Click += gcnew System::EventHandler(this, &Form1::nuskaitytiToolStripMenuItem1\_Click);

//

// išsaugotiToolStripMenuItem

//

this->išsaugotiToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(2) {this->rezultatusToolStripMenuItem,

this->vidurkiųRezultatusToolStripMenuItem});

this->išsaugotiToolStripMenuItem->Name = L"išsaugotiToolStripMenuItem";

this->išsaugotiToolStripMenuItem->Size = System::Drawing::Size(152, 22);

this->išsaugotiToolStripMenuItem->Text = L"Išsaugoti...";

//

// rezultatusToolStripMenuItem

//

this->rezultatusToolStripMenuItem->Name = L"rezultatusToolStripMenuItem";

this->rezultatusToolStripMenuItem->Size = System::Drawing::Size(172, 22);

this->rezultatusToolStripMenuItem->Text = L"Rezultatus";

this->rezultatusToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::rezultatusToolStripMenuItem\_Click);

//

// vidurkiųRezultatusToolStripMenuItem

//

this->vidurkiųRezultatusToolStripMenuItem->Name = L"vidurkiųRezultatusToolStripMenuItem";

this->vidurkiųRezultatusToolStripMenuItem->Size = System::Drawing::Size(172, 22);

this->vidurkiųRezultatusToolStripMenuItem->Text = L"Vidurkių rezultatus";

this->vidurkiųRezultatusToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::vidurkiųRezultatusToolStripMenuItem\_Click);

//

// toolStripSeparator1

//

this->toolStripSeparator1->Name = L"toolStripSeparator1";

this->toolStripSeparator1->Size = System::Drawing::Size(149, 6);

//

// išeitiToolStripMenuItem1

//

this->išeitiToolStripMenuItem1->Name = L"išeitiToolStripMenuItem1";

this->išeitiToolStripMenuItem1->Size = System::Drawing::Size(152, 22);

this->išeitiToolStripMenuItem1->Text = L"Išeiti";

this->išeitiToolStripMenuItem1->Click += gcnew System::EventHandler(this, &Form1::išeitiToolStripMenuItem1\_Click);

//

// skaičiuotiToolStripMenuItem

//

this->skaičiuotiToolStripMenuItem->Enabled = false;

this->skaičiuotiToolStripMenuItem->Name = L"skaičiuotiToolStripMenuItem";

this->skaičiuotiToolStripMenuItem->Size = System::Drawing::Size(70, 20);

this->skaičiuotiToolStripMenuItem->Text = L"Skaičiuoti";

this->skaičiuotiToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::skaičiuotiToolStripMenuItem\_Click);

//

// vidurkisToolStripMenuItem

//

this->vidurkisToolStripMenuItem->Enabled = false;

this->vidurkisToolStripMenuItem->Name = L"vidurkisToolStripMenuItem";

this->vidurkisToolStripMenuItem->Size = System::Drawing::Size(61, 20);

this->vidurkisToolStripMenuItem->Text = L"Vidurkis";

this->vidurkisToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::vidurkisToolStripMenuItem\_Click);

//

// apieToolStripMenuItem

//

this->apieToolStripMenuItem->Name = L"apieToolStripMenuItem";

this->apieToolStripMenuItem->Size = System::Drawing::Size(43, 20);

this->apieToolStripMenuItem->Text = L"Apie";

this->apieToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::apieToolStripMenuItem\_Click);

//

// richTextBox1

//

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox1->Location = System::Drawing::Point(0, 59);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->Size = System::Drawing::Size(764, 191);

this->richTextBox1->TabIndex = 1;

this->richTextBox1->Text = L"";

this->richTextBox1->TextChanged += gcnew System::EventHandler(this, &Form1::richTextBox1\_TextChanged);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label1->Location = System::Drawing::Point(12, 33);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(99, 23);

this->label1->TabIndex = 2;

this->label1->Text = L"Duomenys";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label2->Location = System::Drawing::Point(12, 253);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(95, 23);

this->label2->TabIndex = 4;

this->label2->Text = L"Rezultatai";

//

// richTextBox2

//

this->richTextBox2->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox2->Location = System::Drawing::Point(0, 279);

this->richTextBox2->Name = L"richTextBox2";

this->richTextBox2->Size = System::Drawing::Size(764, 196);

this->richTextBox2->TabIndex = 3;

this->richTextBox2->Text = L"";

this->richTextBox2->TextChanged += gcnew System::EventHandler(this, &Form1::richTextBox2\_TextChanged);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label3->Location = System::Drawing::Point(12, 478);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(336, 23);

this->label3->TabIndex = 6;

this->label3->Text = L"Mažai uždirbančių mokslininkų sąrašas";

//

// richTextBox3

//

this->richTextBox3->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox3->Location = System::Drawing::Point(0, 504);

this->richTextBox3->Name = L"richTextBox3";

this->richTextBox3->Size = System::Drawing::Size(764, 138);

this->richTextBox3->TabIndex = 5;

this->richTextBox3->Text = L"";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Times New Roman", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->label4->Location = System::Drawing::Point(491, 478);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(88, 23);

this->label4->TabIndex = 7;

this->label4->Text = L"Vidurkis:";

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(585, 482);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(127, 20);

this->textBox1->TabIndex = 8;

this->textBox1->TextChanged += gcnew System::EventHandler(this, &Form1::textBox1\_TextChanged);

//

// openFileDialog1

//

this->openFileDialog1->FileName = L"openFileDialog1";

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(764, 644);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->richTextBox3);

this->Controls->Add(this->label2);

this->Controls->Add(this->richTextBox2);

this->Controls->Add(this->label1);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->menuStrip1);

this->MainMenuStrip = this->menuStrip1;

this->Name = L"Form1";

this->Text = L"Programa";

this->Load += gcnew System::EventHandler(this, &Form1::Form1\_Load);

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragmaendregion

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void išeitiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void nuskaitytiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void richTextBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void menuStrip1\_ItemClicked(System::Object^ sender, System::Windows::Forms::ToolStripItemClickedEventArgs^ e) {

}

private: System::Void richTextBox2\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void skaičiuotiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

Apskaiciavimas(info, infoc);

ofstream rez2(Crez2.c\_str());

rez2.close();

RezultatuIsvedimas(infoc, Crez2);

vidurkisToolStripMenuItem->Enabled = true;

richTextBox2->LoadFile("Rezultatai2.txt", RichTextBoxStreamType::PlainText);

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void vidurkisToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

double vidurkis = Vidurkiai(infoc, infoc2);

ofstream rez3(Crez3.c\_str());

rez3.close();

RezultatuIsvedimas(infoc2, Crez3);

richTextBox3->LoadFile("Rezultatai3.txt", RichTextBoxStreamType::PlainText);

textBox1->Text = ((int)vidurkis).ToString();

}

private: System::Void apieToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

::DialogResult result = dialog->ShowDialog(this);

}

private: System::Void išeitiToolStripMenuItem1\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void nuskaitytiToolStripMenuItem1\_Click(System::Object^ sender, System::EventArgs^ e) {

openFileDialog1->DefaultExt = ".txt";

openFileDialog1->Filter = "TXT failai|\*.txt";

if (openFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK &&

openFileDialog1->FileName->Length > 0) {

String^ dfv;

dfv = openFileDialog1->FileName;

richTextBox1->LoadFile(dfv, RichTextBoxStreamType::PlainText);

string dfvs;

StringTostring(dfv, dfvs);

NuskaitymoVeiksmai(skaičiuotiToolStripMenuItem, richTextBox1, info, dfvs);

}

}

private: System::Void išsaugotiToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {}

private: System::Void rezultatusToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

// Inicializuoja saveFileDialog vaizdą

saveFileDialog1->DefaultExt = ".txt";

saveFileDialog1->Filter = "TXT failai|\*.txt";

// Patikrina, ar vartotojas pasirinko failą iš SaveFileDialog?

if (saveFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK &&

saveFileDialog1->FileName->Length > 0) {

String^ rfv; // rezultatų failo vardas (String ^)

rfv = saveFileDialog1->FileName;

// Rezultatų failo vardas rfv (Sting ^) konvertuojamas į rfvs (string)

string rfvs; // rezultatų failo vardas (string)

StringTostring(rfv, rfvs);

// Sukuriamas tuščias rezultatų failas

ofstream fr(rfvs.c\_str());

fr.close();

RezultatuIsvedimas(infoc, rfvs);

}

}

private: System::Void vidurkiųRezultatusToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e) {

// Inicializuoja saveFileDialog vaizdą

saveFileDialog1->DefaultExt = ".txt";

saveFileDialog1->Filter = "TXT failai|\*.txt";

// Patikrina, ar vartotojas pasirinko failą iš SaveFileDialog?

if (saveFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK &&

saveFileDialog1->FileName->Length > 0) {

String^ rfv; // rezultatų failo vardas (String ^)

rfv = saveFileDialog1->FileName;

// Rezultatų failo vardas rfv (Sting ^) konvertuojamas į rfvs (string)

string rfvs; // rezultatų failo vardas (string)

StringTostring(rfv, rfvs);

// Sukuriamas tuščias rezultatų failas

ofstream fr(rfvs.c\_str());

fr.close();

RezultatuIsvedimas(infoc2, rfvs);

}

}

};

}

## Pradiniai duomenys ir rezultatai

**Duomenys.txt:**

5000 2500 8000 1000

38604163754, Hhhhhh, Hhhhhhhhh, 5 9 7 2

37407011234, Gggggg, Ggggggggg, 2 1 0 3

39012128243, Ffffff, Fffffffff, 6 7 1 9

36401288430, Eeeeee, Eeeeeeeee, 1 2 3 4

37702131348, Dddddd, Ddddddddd, 9 4 3 6

37103044686, Cccccc, Ccccccccc, 4 5 4 6

33008141568, Aaaaaa, Aaaaaaaaa, 7 2 2 3

39011128434, Bbbbbb, Bbbbbbbbb, 7 6 1 9

**Rezultatai.txt:**

5000 2500 8000 1000

Kodas Pavarde Vardas Indeliai

-----------------------------------------

38604163754 HhhhhhHhhhhhhhh 5 9 7 2

37407011234 GgggggGgggggggg 2 1 0 3

39012128243 FfffffFffffffff 6 7 1 9

36401288430 EeeeeeEeeeeeeee 1 2 3 4

37702131348 DdddddDdddddddd 9 4 3 6

37103044686 CcccccCcccccccc 4 5 4 6

33008141568 AaaaaaAaaaaaaaa 7 2 2 3

39011128434 BbbbbbBbbbbbbbb 7 6 1 9

--------------------------------------

**Rezultatai2.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

37103044686 CcccccCcccccccc 4 5 4 6 487 347 1523 142 2501

37702131348 DdddddDdddddddd 9 4 3 6 1097 277 1142 142 2661

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

38604163754 HhhhhhHhhhhhhhh 5 9 7 2 609 625 2666 47 3949

-----------------------------------------------------------------------------------------

**Rezultatai3.txt:**

Kodas Pavarde Vardas Indeliai Premijos Suma

-----------------------------------------------------------------------------------------

33008141568 AaaaaaAaaaaaaaa 7 2 2 3 853 138 761 71 1825

39011128434 BbbbbbBbbbbbbbb 7 6 1 9 853 416 380 214 1865

36401288430 EeeeeeEeeeeeeee 1 2 3 4 121 138 1142 95 1498

39012128243 FfffffFffffffff 6 7 1 9 731 486 380 214 1813

37407011234 GgggggGgggggggg 2 1 0 3 243 69 0 71 384

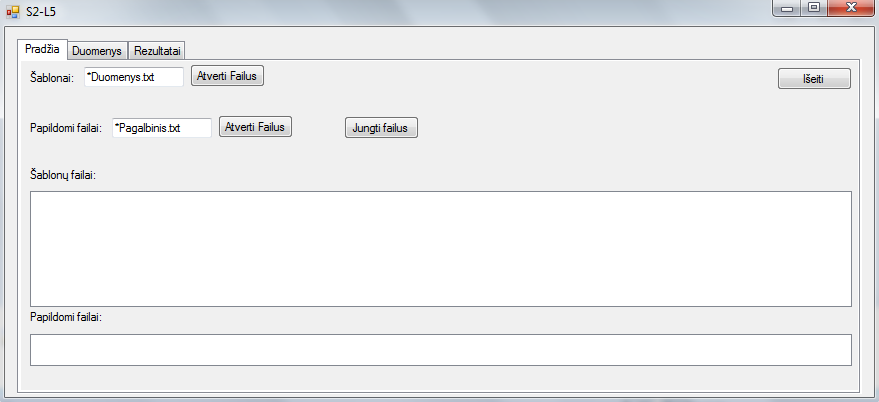
-----------------------------------------------------------------------------------------

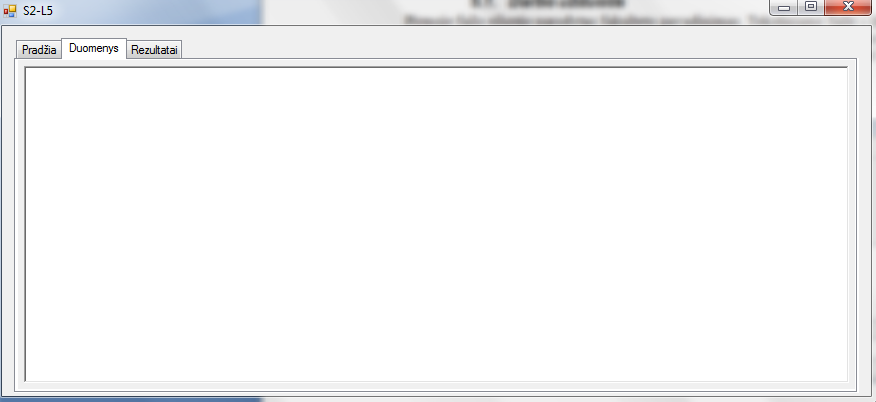
# Šakotas sąrašas (L5)

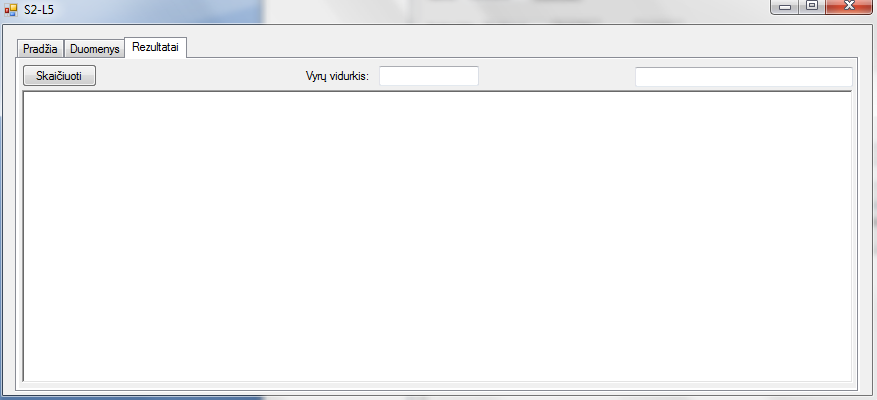
## Darbo užduotis

Pirmoje failo eilutėje nurodytas fakulteto pavadinimas. Tekstiniame faile – žiemos sesijos pažymių sąrašas. Eilutėje apie studentą yra tokie duomenys: pavardė, vardas, grupė, gimimo metai, lytis, pažymiai. Nustatyti, vaikinų ar merginų yra aukštesnis vidutinis mokymosi vidurkis. Suformuoti bendrą sąrašą vaikinų, kurių vidurkis aukštesnis už vidutinį (fakultetas, pavardė)

## Grafinės vartotojo sąsajos schema







## Sąsajoje panaudotų komponentų keičiamos savybės

|  |  |  |
| --- | --- | --- |
| Komponentas | Savybė | Reikšmė |
| Form1 | Text  Size | S2-L5  885;408 |
| tabControl1 | TabPages(Collection) | Pradžia ; Duomenys ; Rezultatai |
| Label3 | Text | Šablonai: |
| Label5 | Text | Papildomi Failai: |
| textBox1 | Text  Lines | \*Duomenys.txt  String[]Array |
| textBox2 | Text  Lines | \*Pagalbinis.txt  String[]Array |
| Button1, Button3 | Text | Atverti Failus |
| Button4 | Text | Jungti Failus |
| Button2 | Text | Išeiti |
| Label4 | Text | Šablonų failai: |
| Label8 | Text | Papildomi failai: |
| Listbox1, Listbox2 | Label  HorizontalScrollBar  ItemHeight  FormattingEnabled | Courier new; 8,25pt  True  14  True |
| RichTextBox1, RichTextBox2 | Lines  Font | String[]Array  Courier new; 8,25pt |
| Button5 | Text | Skaičiuoti |
| Label9 | Text | Vyrų Vidurkis: |
| textBox4 | Lines | String[]Array |
| TextBox3 | Lines | String[]Array |

## Programos vartotojo vadovas

1. Paspauskite „Atverti Failus“ ties šablonų skiltimi, norėdami pasirinkti katalogą, kuriame yra įrašyti visi pirminiai duomenys. Šalia esančiame teksto langelyje nurodykite failų bendrą vardo šaknį.
2. Paspauskite „Atverti Failus“ ties „Papildomi failai“ skiltimi, norėdami pasirinkti katalogą, kuriame yra įrašyti visi papildomi duomenys. Šalia esančiame teksto langelyje nurodykite failų bendrą vardo šaknį.
3. Spauskite „Jungti failus“, norėdami sujungti pirminius ir papildomus duomenis. Informacija bus išvesta skiltyje „Duomenys“.
4. Skiltyje „Rezultatai“ spauskite mygtuką „Skaičiuoti“, norėdami apskaičiuoti, kas turi geresnius vidurkius – merginos ar vaikinai. Šis rezultatas bus pavaizduotas teksto langelyje, esančiame skilties viršuje. Šalia jo bus išvestas langelyje vyrų vidurkių bendras vidurkis, pagal kurį buvo surinktas naujas sąrašas. Šis sąrašas bus pavaizduotas esamos skilties lange.
5. Spauskite mygtuką „Išeiti“ skiltyje „Pradžia“ norėdami baigti darbą su programa. Visi rezultatai bus išsaugomi viename faile „Rezultatai.txt“.

## Programos tekstas

**Studentas.h:**

#pragma once

#include <string>

#include <fstream>

#include <sstream>

#include <iomanip>

using namespace std;

class Studentas

{

private:

string pavarde, vardas, grupe, metai, lytis, fakultetas;

int Npaz, paz[5];

double vidurkis;

public:

Studentas();

Studentas(string a, string b, string c, string d, string e, int f, int h[], string i);

~Studentas();

void VidurkioApskaiciavimas();

string SpausdintiEilute();

string ImtiVarda();

string ImtiPavarde();

string ImtiGrupe();

string ImtiMetus();

string ImtiLyti();

string ImtiFakulteta();

bool operator < (const Studentas & R);

double ImtiVidurki();

int ImtiPazymi (int i);

};

**Studentas.cpp:**

#include "StdAfx.h"

#include "Studentas.h"

//--------------------------------------------------------------------------------------

Studentas::Studentas(): pavarde(""), vardas(""), grupe(""), metai(""), lytis(""), Npaz(0), vidurkis(0), fakultetas("")

{

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

paz[i] = 0;

}

//--------------------------------------------------------------------------------------

Studentas::Studentas(string a, string b, string c, string d, string e, int f, int h[], string i):

pavarde(a), vardas(b), grupe(c), metai(d), lytis(e), Npaz(f), fakultetas(i)

{

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

paz[i] = h[i];

vidurkis = 0;

}

//--------------------------------------------------------------------------------------

Studentas::~Studentas() {}

//--------------------------------------------------------------------------------------

void Studentas::VidurkioApskaiciavimas()

{

double s = 0;

ofstream rez("Rezultatai3.txt", ios::app);

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

{

rez << paz[i] << endl;

s += paz[i];

}

vidurkis = s / Npaz;

rez.close();

}

//--------------------------------------------------------------------------------------

string Studentas::SpausdintiEilute()

{

stringstream ss;

ss << left << setw(15) << pavarde << left << setw(15) << vardas << setw(10) << grupe << setw(12) << metai << setw(10) << lytis;

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

ss << right << setw(3) << paz[i];

ss<< ' ';

if (vidurkis != 0.0)

ss << right << setw(7) << vidurkis;

return ss.str();

}

//--------------------------------------------------------------------------------------

bool Studentas::operator < (const Studentas & R)

{

return fakultetas < R.fakultetas || (fakultetas == R.fakultetas && pavarde < R.pavarde);

}

//--------------------------------------------------------------------------------------

double Studentas::ImtiVidurki()

{

return vidurkis;

}

//--------------------------------------------------------------------------------------

string Studentas::ImtiVarda()

{

return vardas;

}

string Studentas::ImtiPavarde()

{

return pavarde;

}

//--------------------------------------------------------------------------------------

string Studentas::ImtiGrupe()

{

return grupe;

}

string Studentas::ImtiMetus()

{

return metai;

}

string Studentas::ImtiLyti()

{

return lytis;

}

string Studentas::ImtiFakulteta()

{

return fakultetas;

}

//------------------------------------------

int Studentas::ImtiPazymi (int i)

{

return paz[i];

}

**Elementas.h:**

#pragma once

#include "Studentas.h"

using namespace std;

class Elementas

{

private:

Studentas duom;

Elementas \* kitas;

public:

Elementas();

~Elementas();

void DetiStudenta(Studentas x);

Studentas ImtiStudenta();

Elementas \* ImtiPirmyn();

void DetiPirmyn(Elementas \* p);

};

**Elementas.cpp:**

#include "StdAfx.h"

#include "Elementas.h"

Elementas::Elementas(): duom(Studentas()), kitas(NULL) {}

Elementas::~Elementas() {}

//-----------------------------------------

void Elementas::DetiStudenta(Studentas x)

{

duom = x;

}

//-----------------------------------------

Studentas Elementas::ImtiStudenta()

{

return duom;

}

//-----------------------------------------

Elementas \* Elementas::ImtiPirmyn()

{

return kitas;

}

//-----------------------------------------

void Elementas::DetiPirmyn(Elementas \* p)

{

kitas = p;

}

//-----------------------------------------

**Fakultetas.h:**

#pragma once

#include "Elementas.h"

#include "SarasuGrupe.h"

using namespace std;

class Fakultetas

{

private:

Fakultetas \* kitas;

string info;

Elementas \*p;

Elementas \*d;

public:

Fakultetas();

~Fakultetas();

string ImtiInfo();

void DetiInfo(string x);

void DetiDuomenis(Studentas x);

void SpausdintiFakulteta(string Crez);

bool YraElementas();

void FakultetoPradzia();

void SekantisElementas();

Fakultetas \* ImtiPirmyn();

void DetiIPabaiga(Studentas x);

void Rikiavimas();

void Papildymas(SarasuGrupe \* x);

void Naikinti();

void DetiPirmyn(Fakultetas \* r);

void Lytis(double & a, int & b, double & c, int & e);

void Isarasa(SarasuGrupe \*S, double v);

};

**Fakultetas.cpp:**

#include "StdAfx.h"

#include "Fakultetas.h"

Fakultetas::Fakultetas() : kitas(NULL), info(""), p(NULL) {}

Fakultetas::~Fakultetas() { Naikinti();}

//----------------------------------------------------------

void Fakultetas::DetiInfo(string x)

{

info = x;

}

//----------------------------------------------------------

void Fakultetas::SpausdintiFakulteta(string Crez)

{

FakultetoPradzia();

ofstream rez(Crez.c\_str(), ios::app);

rez <<"------------------------------------------------------------------------" << endl;

rez << info << endl;

while (YraElementas()){

rez << d->ImtiStudenta().SpausdintiEilute() << endl;

d = d->ImtiPirmyn();

}

rez <<"------------------------------------------------------------------------" << endl;

rez.close();

}

//----------------------------------------------------------

bool Fakultetas::YraElementas()

{

return (d != NULL);

}

//----------------------------------------------------------

void Fakultetas::FakultetoPradzia()

{

d = p;

}

//----------------------------------------------------------

void Fakultetas::SekantisElementas()

{

d = d->ImtiPirmyn();

}

//----------------------------------------------------------

void Fakultetas::DetiIPabaiga(Studentas x)

{

FakultetoPradzia();

Elementas \*t = new Elementas;

t->DetiStudenta(x);

t->DetiPirmyn(NULL);

if (p == NULL)

p = t;

else

{

Elementas \*v = new Elementas;

for (v = p; v->ImtiPirmyn() != NULL; v = v->ImtiPirmyn());

v->DetiPirmyn(t);

}

}

//----------------------------------------------------------

void Fakultetas::Rikiavimas()

{

bool yra = true;

Elementas \*d, \*d1, \*r;

while (yra) {

yra = false;

d = d1 = r = p;

while (d1 != NULL) {

if (d1->ImtiStudenta() < d->ImtiStudenta()) {

yra = true;

if (d == p) { // pirmas elementas su kaimynu

p = p->ImtiPirmyn();

d->DetiPirmyn(d1->ImtiPirmyn());

d1->DetiPirmyn(d);

}

else { // pora tolesnių elementų

d->DetiPirmyn(d1->ImtiPirmyn());

d1->DetiPirmyn(d);

r->DetiPirmyn(d1);

}

}

r = d;

d = d1;

d1 = d1->ImtiPirmyn();

}

}

}

//----------------------------------------------------------

void Fakultetas::DetiDuomenis(Studentas x)

{

d->DetiStudenta(x);

}

//----------------------------------------------------------

void Fakultetas::Papildymas(SarasuGrupe \* x)

{

x->Pradzia();

while (x->YraSarasas())

{

string vardas = x->ImtiStudenta().ImtiVarda();

string pavarde = x->ImtiStudenta().ImtiPavarde();

FakultetoPradzia();

while (YraElementas())

{

string vardas2 = d->ImtiStudenta().ImtiVarda();

string pavarde2 = d->ImtiStudenta().ImtiPavarde();

if (vardas == vardas2 && pavarde == pavarde2)

{

int paz[5];

int N = 0;

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

paz[j] = x->ImtiStudenta().ImtiPazymi(j);

Studentas S = d->ImtiStudenta();

Studentas T = Studentas(S.ImtiPavarde(), S.ImtiVarda(), S.ImtiGrupe(), S.ImtiMetus(), S.ImtiLyti(), 5, paz, S.ImtiFakulteta());

T.VidurkioApskaiciavimas();

d->DetiStudenta(T);

}

SekantisElementas();

}

x->Kitas();

}

}

//----------------------------------------------------------

string Fakultetas::ImtiInfo()

{

return info;

}

//----------------------------------------------------------

void Fakultetas::Naikinti()

{

while (p != NULL){

d = p;

p = p->ImtiPirmyn();

delete d;

}

}

//----------------------------------------------------------

Fakultetas \*Fakultetas::ImtiPirmyn()

{

return kitas;

}

//----------------------------------------------------------

void Fakultetas::DetiPirmyn(Fakultetas \* r)

{

kitas = r;

}

//----------------------------------------------------------

void Fakultetas::Lytis(double & a, int & b, double & c, int & e)

{

a = 0.0; //vyru vidurkiu suma

b = 0; //vary vidurkiu kiekis

c = 0.0; //moteru vidurkiu suma

e = 0; //moteru vidurkiu kiekis

d = p;

while (YraElementas())

{

Studentas S = d->ImtiStudenta();

if (S.ImtiLyti() == "vyras")

{

a+= S.ImtiVidurki();

b++;

}

else

if (S.ImtiLyti() == "moteris")

{

c+= S.ImtiVidurki();

e++;

}

SekantisElementas();

}

}

//--------------------------------------------------

void Fakultetas::Isarasa(SarasuGrupe \*S, double v)

{

FakultetoPradzia();

while (YraElementas())

{

if (d->ImtiStudenta().ImtiLyti() == "vyras")

if (d->ImtiStudenta().ImtiVidurki() >= v)

S->DetiIPabaiga(d->ImtiStudenta());

SekantisElementas();

}

}

**Medis.h:**

#pragma once

#include "Fakultetas.h"

#include "SarasuGrupe.h"

//#include <strsafe.h>

//#include <vcclr.h>

using namespace std;

class Medis

{

private:

Fakultetas \*p;

Fakultetas \*d;

public:

Medis();

~Medis();

void Pradzia();

void Kitas();

bool Yra();

void DetiFakulteta(Fakultetas \* F);

void Isvesti(string Crez);

void NaikintiMedi();

void DuomenuPapildymas(SarasuGrupe \* x);

string SkaiciuotiLyti();

double VyruVidurkis();

void VaikinuAtrinkimas(SarasuGrupe \*S, double v);

};

**Medis.cpp:**

#include "StdAfx.h"

#include "Medis.h"

Medis::Medis() : p(NULL), d(NULL) {}

Medis::~Medis()

{

NaikintiMedi();

}

//----------------------------------------------

void Medis::Pradzia()

{

d = p;

}

//----------------------------------------------

void Medis::Kitas()

{

d = d->ImtiPirmyn();

}

//----------------------------------------------

bool Medis::Yra()

{

return (d != NULL);

}

//----------------------------------------------

void Medis::DetiFakulteta(Fakultetas \* F)

{

Pradzia();

Fakultetas \* t = new Fakultetas;

t = F;

t->DetiPirmyn(NULL);

if (p == NULL)

p = t;

else {

Fakultetas \*v = new Fakultetas;

for (v = p; v->ImtiPirmyn() != NULL; v = v->ImtiPirmyn());

v->DetiPirmyn(t);

}

}

//-----------------------------------------------------

void Medis::Isvesti(string Crez)

{

ofstream rez(Crez.c\_str());

Pradzia();

rez << "/------------------------------------------------------------------------------------------------/" << endl;

rez << left << setw(15) << "Pavardė" << left << setw(15) << "Vardas" << setw(10) << "Grupė" << setw(12) << "Metai" << setw(10) << "Lytis" << setw(16) << "Pažymiai" << setw(9) << "Vidurkis" << endl;

rez << "/------------------------------------------------------------------------------------------------/" << endl;

rez.close();

while (Yra())

{

d->SpausdintiFakulteta(Crez);

Kitas();

}

rez.close();

}

//----------------------------------------------

void Medis::NaikintiMedi()

{

while (p != NULL){

d = p;

p = p->ImtiPirmyn();

delete d;

}

}

//----------------------------------------------

void Medis::DuomenuPapildymas(SarasuGrupe \* x)

{

Pradzia();

while (Yra()){

d->Papildymas(x);

Kitas();

}

}

//----------------------------------------------

string Medis::SkaiciuotiLyti()

{

double Vyrai = 0;

double Moterys = 0;

int VyraiN = 0;

int MoterysN = 0;

d = p;

while (Yra())

{

double a = 0.0;

int b = 0;

double c = 0.0;

int e = 0;

d->Lytis(a, b, c, e);

Vyrai += a; VyraiN += b; Moterys += c; MoterysN += e;

Kitas();

}

if (Vyrai / VyraiN > Moterys / MoterysN)

return "Vyrų vidurkis yra aukštesnis";

else

if (Vyrai / VyraiN < Moterys / MoterysN)

return "Moterų vidurkis yra aukštesnis";

else

return "Vyrų ir Moterų vidurkiai yra vienodi";

}

//-----------------------------------------------------

double Medis::VyruVidurkis()

{

double Vyrai = 0;

int VyraiN = 0;

d = p;

while (Yra())

{

double a = 0.0;

int b = 0;

double c = 0.0;

int e = 0;

d->Lytis(a, b, c, e);

Vyrai += a; VyraiN += b;

Kitas();

}

return Vyrai / VyraiN;

}

//-------------------------------------------------------

void Medis::VaikinuAtrinkimas(SarasuGrupe \*S, double v)

{

Pradzia();

while (Yra())

{

d->Isarasa(S, v);

Kitas();

}

}

**SarasuGupe.h:**

#pragma once

#include "Elementas.h"

using namespace std;

class SarasuGrupe

{

private:

Elementas \*p;

Elementas \*d;

public:

SarasuGrupe();

~SarasuGrupe();

void Pradzia();

void Kitas();

void DetiIPabaiga(Studentas x);

Studentas ImtiStudenta();

void Naikinti();

bool YraSarasas();

void Rikiavimas();

void Isvedimas(string Crez);

};

**SarasuGrupe.cpp:**

#include "StdAfx.h"

#include "SarasuGrupe.h"

SarasuGrupe::SarasuGrupe(): p(NULL), d(NULL) {}

SarasuGrupe::~SarasuGrupe() {Naikinti();}

//----------------------------------------------------------

void SarasuGrupe::DetiIPabaiga(Studentas x)

{

Pradzia();

Elementas \*t = new Elementas;

t->DetiStudenta(x);

t->DetiPirmyn(NULL);

if (p == NULL)

p = t;

else

{

Elementas \*v = new Elementas;

for (v = p; v->ImtiPirmyn() != NULL; v = v->ImtiPirmyn());

v->DetiPirmyn(t);

}

}

//---------------------------------------------------------

void SarasuGrupe::Pradzia()

{

d = p;

}

//---------------------------------------------------------

void SarasuGrupe::Kitas()

{

d = d->ImtiPirmyn();

}

//---------------------------------------------------------

Studentas SarasuGrupe::ImtiStudenta()

{

return d->ImtiStudenta();

}

//---------------------------------------------------------

bool SarasuGrupe::YraSarasas()

{

return (d != NULL);

}

//---------------------------------------------------------

void SarasuGrupe::Naikinti()

{

while (p != NULL){

d = p;

p = p->ImtiPirmyn();

delete d;

}

}

//---------------------------------------------------------

void SarasuGrupe::Rikiavimas()

{

bool yra = true;

Elementas \*d, \*d1, \*r;

while (yra) {

yra = false;

d = d1 = r = p;

while (d1 != NULL) {

if (d1->ImtiStudenta() < d->ImtiStudenta()) {

yra = true;

if (d == p) { // pirmas elementas su kaimynu

p = p->ImtiPirmyn();

d->DetiPirmyn(d1->ImtiPirmyn());

d1->DetiPirmyn(d);

}

else { // pora tolesnių elementų

d->DetiPirmyn(d1->ImtiPirmyn());

d1->DetiPirmyn(d);

r->DetiPirmyn(d1);

}

}

r = d;

d = d1;

d1 = d1->ImtiPirmyn();

}

}

}

//-------------------------------------------------------------

void SarasuGrupe::Isvedimas(string Crez)

{

ofstream rez(Crez.c\_str(), ios::app);

Pradzia();

rez << "/------------------------------------------------------------------------------------------------/" << endl;

rez << left << setw(15) << "Pavardė" << left << setw(15) << "Vardas" << setw(10) << "Grupė" << setw(12) << "Metai" << setw(10) << "Lytis" << setw(16) << "Pažymiai" << setw(9) << "Vidurkis" << right << setw(14) << "Fakultetas" << endl;

rez << "/------------------------------------------------------------------------------------------------/" << endl;

while (YraSarasas())

{

rez << d->ImtiStudenta().SpausdintiEilute() << " " << d->ImtiStudenta().ImtiFakulteta() << endl;

Kitas();

}

rez << "/------------------------------------------------------------------------------------------------/" << endl;

rez.close();

}

**Form1.h:**

#pragma once

#include <windows.h> // Čia yra failų išrinkimui reikalingi tipai

#include <strsafe.h> // TCHAR[] funkcijos

#include <vcclr.h>

#include "Medis.h"

const string Crez = "Rezultatai.txt";

const string Crez2 = "Rezultatai2.txt";

namespace S2L5 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

void StringTostring(String ^ s, string & os)

{

using namespace Runtime::InteropServices;

const char\* chars = (const char\*)(Marshal::StringToHGlobalAnsi(s)).ToPointer();

os = chars;

Marshal::FreeHGlobal(IntPtr((void\*)chars));

}

void Atrinkimas(Medis \* info, SarasuGrupe \* galutinis, RichTextBox^ richTextBox2, TextBox^ textBox4)

{

double vidurkis = info->VyruVidurkis();

stringstream ss;

ss << right << setw(4) << vidurkis;

String ^sak = gcnew String((ss.str()).c\_str()); // string to String

textBox4->Text = sak;

info->VaikinuAtrinkimas(galutinis, vidurkis);

ofstream rez(Crez2.c\_str());

rez.close();

galutinis->Rikiavimas();

galutinis->Isvedimas(Crez2);

galutinis->Isvedimas(Crez);

richTextBox2->LoadFile("Rezultatai2.txt", RichTextBoxStreamType::PlainText);

}

//----------------------------------------------------

void PapildomasNuskaitymas(TCHAR Cduom[MAX\_PATH], SarasuGrupe \* papild)

{

string pavarde, vardas;

int u[5];

ifstream duom(Cduom);

while (!duom.eof())

{

getline (duom, pavarde, ',');

getline (duom, vardas, ',');

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

duom >> u[i];

duom.ignore();

papild->DetiIPabaiga(Studentas(pavarde, vardas, "", "", "", 5, u, ""));

}

duom.close();

}

//----------------------------------------------------

void Nuskaitymas(TCHAR Cduom[MAX\_PATH], Medis \* info)

{

string fakultetas, pavarde, vardas, grupe, metai, lytis;

int u[1];

u[0] = 0;

ifstream duom(Cduom);

Fakultetas \* F = new Fakultetas;

getline (duom, fakultetas);

F->DetiInfo(fakultetas);

while (!duom.eof())

{

getline (duom, pavarde, ',');

getline (duom, vardas, ',');

duom >> grupe >> metai >> lytis;

duom.ignore();

F->DetiIPabaiga(Studentas(pavarde, vardas, grupe, metai, lytis, 0, u, fakultetas));

}

info->DetiFakulteta(F);

duom.close();

}

//----------------------------------------------------

/// <summary>

/// Summary for Form1

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

public ref class Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

info = new Medis;

papild = new SarasuGrupe;

galutinis = new SarasuGrupe;

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

delete info;

delete papild;

delete galutinis;

}

private: Medis \*info;

private: SarasuGrupe \* papild;

private: SarasuGrupe \* galutinis;

private: System::Windows::Forms::Label^ label1;

protected:

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::ListBox^ listBox1;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::FolderBrowserDialog^ folderBrowserDialog1;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Label^ label7;

private: System::Windows::Forms::ListBox^ listBox2;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label8;

private: System::Windows::Forms::Button^ button4;

private: System::Windows::Forms::TabControl^ tabControl1;

private: System::Windows::Forms::TabPage^ tabPage1;

private: System::Windows::Forms::TabPage^ tabPage2;

private: System::Windows::Forms::TabPage^ tabPage3;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Button^ button5;

private: System::Windows::Forms::RichTextBox^ richTextBox2;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::Label^ label9;

private: System::Windows::Forms::TextBox^ textBox4;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->label1 = (gcnew System::Windows::Forms::Label());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->button1 = (gcnew System::Windows::Forms::Button());

this->label3 = (gcnew System::Windows::Forms::Label());

this->listBox1 = (gcnew System::Windows::Forms::ListBox());

this->button2 = (gcnew System::Windows::Forms::Button());

this->folderBrowserDialog1 = (gcnew System::Windows::Forms::FolderBrowserDialog());

this->label5 = (gcnew System::Windows::Forms::Label());

this->button3 = (gcnew System::Windows::Forms::Button());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->label7 = (gcnew System::Windows::Forms::Label());

this->listBox2 = (gcnew System::Windows::Forms::ListBox());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label8 = (gcnew System::Windows::Forms::Label());

this->button4 = (gcnew System::Windows::Forms::Button());

this->tabControl1 = (gcnew System::Windows::Forms::TabControl());

this->tabPage1 = (gcnew System::Windows::Forms::TabPage());

this->tabPage2 = (gcnew System::Windows::Forms::TabPage());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->tabPage3 = (gcnew System::Windows::Forms::TabPage());

this->label9 = (gcnew System::Windows::Forms::Label());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->button5 = (gcnew System::Windows::Forms::Button());

this->richTextBox2 = (gcnew System::Windows::Forms::RichTextBox());

this->tabControl1->SuspendLayout();

this->tabPage1->SuspendLayout();

this->tabPage2->SuspendLayout();

this->tabPage3->SuspendLayout();

this->SuspendLayout();

//

// label1

//

this->label1->Location = System::Drawing::Point(14, 62);

this->label1->MaximumSize = System::Drawing::Size(400, 0);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(400, 0);

this->label1->TabIndex = 0;

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(63, 6);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(100, 20);

this->textBox1->TabIndex = 2;

this->textBox1->Text = L"\*Duomenys.txt";

this->textBox1->TextChanged += gcnew System::EventHandler(this, &Form1::textBox1\_TextChanged);

//

// button1

//

this->button1->Location = System::Drawing::Point(169, 3);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(75, 23);

this->button1->TabIndex = 3;

this->button1->Text = L"Atverti Failus";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &Form1::button1\_Click);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(6, 10);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(51, 13);

this->label3->TabIndex = 4;

this->label3->Text = L"Šablonai:";

//

// listBox1

//

this->listBox1->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->listBox1->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->listBox1->FormattingEnabled = true;

this->listBox1->HorizontalScrollbar = true;

this->listBox1->ItemHeight = 14;

this->listBox1->Location = System::Drawing::Point(9, 130);

this->listBox1->Name = L"listBox1";

this->listBox1->Size = System::Drawing::Size(822, 116);

this->listBox1->TabIndex = 6;

//

// button2

//

this->button2->Location = System::Drawing::Point(756, 6);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(75, 23);

this->button2->TabIndex = 7;

this->button2->Text = L"Išeiti";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &Form1::button2\_Click);

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(6, 60);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(79, 13);

this->label5->TabIndex = 10;

this->label5->Text = L"Papildomi failai:";

this->label5->Click += gcnew System::EventHandler(this, &Form1::label5\_Click);

//

// button3

//

this->button3->Location = System::Drawing::Point(197, 54);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(75, 23);

this->button3->TabIndex = 9;

this->button3->Text = L"Atverti Failus";

this->button3->UseVisualStyleBackColor = true;

this->button3->Click += gcnew System::EventHandler(this, &Form1::button3\_Click);

//

// textBox2

//

this->textBox2->Location = System::Drawing::Point(91, 57);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(100, 20);

this->textBox2->TabIndex = 8;

this->textBox2->Text = L"\*Pagalbinis.txt";

//

// label7

//

this->label7->Location = System::Drawing::Point(14, 118);

this->label7->MaximumSize = System::Drawing::Size(400, 0);

this->label7->Name = L"label7";

this->label7->Size = System::Drawing::Size(400, 0);

this->label7->TabIndex = 11;

//

// listBox2

//

this->listBox2->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->listBox2->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->listBox2->FormattingEnabled = true;

this->listBox2->HorizontalScrollbar = true;

this->listBox2->ItemHeight = 14;

this->listBox2->Location = System::Drawing::Point(9, 273);

this->listBox2->Name = L"listBox2";

this->listBox2->Size = System::Drawing::Size(822, 32);

this->listBox2->TabIndex = 13;

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(6, 107);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(73, 13);

this->label4->TabIndex = 5;

this->label4->Text = L"Šablonų failai:";

this->label4->Click += gcnew System::EventHandler(this, &Form1::label4\_Click);

//

// label8

//

this->label8->AutoSize = true;

this->label8->Location = System::Drawing::Point(6, 249);

this->label8->Name = L"label8";

this->label8->Size = System::Drawing::Size(79, 13);

this->label8->TabIndex = 14;

this->label8->Text = L"Papildomi failai:";

//

// button4

//

this->button4->Location = System::Drawing::Point(323, 55);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(75, 23);

this->button4->TabIndex = 15;

this->button4->Text = L"Jungti failus";

this->button4->UseVisualStyleBackColor = true;

this->button4->Click += gcnew System::EventHandler(this, &Form1::button4\_Click);

//

// tabControl1

//

this->tabControl1->Controls->Add(this->tabPage1);

this->tabControl1->Controls->Add(this->tabPage2);

this->tabControl1->Controls->Add(this->tabPage3);

this->tabControl1->Location = System::Drawing::Point(12, 12);

this->tabControl1->Name = L"tabControl1";

this->tabControl1->SelectedIndex = 0;

this->tabControl1->Size = System::Drawing::Size(845, 355);

this->tabControl1->TabIndex = 16;

//

// tabPage1

//

this->tabPage1->Controls->Add(this->label3);

this->tabPage1->Controls->Add(this->listBox2);

this->tabPage1->Controls->Add(this->label8);

this->tabPage1->Controls->Add(this->button4);

this->tabPage1->Controls->Add(this->textBox1);

this->tabPage1->Controls->Add(this->listBox1);

this->tabPage1->Controls->Add(this->button1);

this->tabPage1->Controls->Add(this->label4);

this->tabPage1->Controls->Add(this->button2);

this->tabPage1->Controls->Add(this->button3);

this->tabPage1->Controls->Add(this->label5);

this->tabPage1->Controls->Add(this->textBox2);

this->tabPage1->Location = System::Drawing::Point(4, 22);

this->tabPage1->Name = L"tabPage1";

this->tabPage1->Padding = System::Windows::Forms::Padding(3);

this->tabPage1->Size = System::Drawing::Size(837, 329);

this->tabPage1->TabIndex = 0;

this->tabPage1->Text = L"Pradžia";

this->tabPage1->Click += gcnew System::EventHandler(this, &Form1::tabPage1\_Click);

//

// tabPage2

//

this->tabPage2->Controls->Add(this->richTextBox1);

this->tabPage2->Location = System::Drawing::Point(4, 22);

this->tabPage2->Name = L"tabPage2";

this->tabPage2->Padding = System::Windows::Forms::Padding(3);

this->tabPage2->Size = System::Drawing::Size(837, 329);

this->tabPage2->TabIndex = 1;

this->tabPage2->Text = L"Duomenys";

//

// richTextBox1

//

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox1->Location = System::Drawing::Point(6, 6);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->Size = System::Drawing::Size(825, 317);

this->richTextBox1->TabIndex = 0;

this->richTextBox1->Text = L"";

//

// tabPage3

//

this->tabPage3->Controls->Add(this->label9);

this->tabPage3->Controls->Add(this->textBox4);

this->tabPage3->Controls->Add(this->textBox3);

this->tabPage3->Controls->Add(this->button5);

this->tabPage3->Controls->Add(this->richTextBox2);

this->tabPage3->Location = System::Drawing::Point(4, 22);

this->tabPage3->Name = L"tabPage3";

this->tabPage3->Size = System::Drawing::Size(837, 329);

this->tabPage3->TabIndex = 2;

this->tabPage3->Text = L"Rezultatai";

//

// label9

//

this->label9->AutoSize = true;

this->label9->Location = System::Drawing::Point(284, 10);

this->label9->Name = L"label9";

this->label9->Size = System::Drawing::Size(70, 13);

this->label9->TabIndex = 4;

this->label9->Text = L"Vyrų vidurkis:";

//

// textBox4

//

this->textBox4->Location = System::Drawing::Point(360, 7);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(100, 20);

this->textBox4->TabIndex = 3;

//

// textBox3

//

this->textBox3->Location = System::Drawing::Point(616, 8);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(218, 20);

this->textBox3->TabIndex = 2;

this->textBox3->TextChanged += gcnew System::EventHandler(this, &Form1::textBox3\_TextChanged);

//

// button5

//

this->button5->Location = System::Drawing::Point(3, 5);

this->button5->Name = L"button5";

this->button5->Size = System::Drawing::Size(75, 23);

this->button5->TabIndex = 1;

this->button5->Text = L"Skaičiuoti";

this->button5->UseVisualStyleBackColor = true;

this->button5->Click += gcnew System::EventHandler(this, &Form1::button5\_Click);

//

// richTextBox2

//

this->richTextBox2->Font = (gcnew System::Drawing::Font(L"Courier New", 8.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(186)));

this->richTextBox2->Location = System::Drawing::Point(3, 31);

this->richTextBox2->Name = L"richTextBox2";

this->richTextBox2->Size = System::Drawing::Size(831, 293);

this->richTextBox2->TabIndex = 0;

this->richTextBox2->Text = L"";

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(869, 370);

this->Controls->Add(this->tabControl1);

this->Controls->Add(this->label7);

this->Controls->Add(this->label1);

this->Name = L"Form1";

this->Text = L"S2-L5";

this->Load += gcnew System::EventHandler(this, &Form1::Form1\_Load);

this->tabControl1->ResumeLayout(false);

this->tabPage1->ResumeLayout(false);

this->tabPage1->PerformLayout();

this->tabPage2->ResumeLayout(false);

this->tabPage3->ResumeLayout(false);

this->tabPage3->PerformLayout();

this->ResumeLayout(false);

}

#pragma endregion

private: System::Void Form1\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void label4\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e) {

Close();

}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox1->Text == L"") {

MessageBox::Show (L"Įrašykite šabloną", L"Neįvesta reikšmė", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (folderBrowserDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK && folderBrowserDialog1->SelectedPath ->Length > 0) {

listBox1->Items->Clear(); // Išvalome listBox1 komponentą

WIN32\_FIND\_DATA ffd; // Paieškai reikalingi ...

HANDLE hFind = INVALID\_HANDLE\_VALUE; // ... tarnybiniai kintamieji

// Funkcijai PtrToStringChars() reikalingas #include <vcclr.h>

pin\_ptr<const wchar\_t> pletinys = PtrToStringChars(textBox1->Text);

TCHAR katalogas[MAX\_PATH]; // Saugosime kelią iki duomenų failų

TCHAR paieskosSablonas[MAX\_PATH]; // Saugosime paieškos šabloną

TCHAR pilnasVardas[MAX\_PATH]; // Saugosime konkretaus duomenų failo pilną (su keliu) vardą

// Sudarome kelią į katalogą, kuriame sudėti duomenų failai

pin\_ptr<const wchar\_t> pasirinktasKatalogas = PtrToStringChars(folderBrowserDialog1->SelectedPath);

StringCchCopy(katalogas, MAX\_PATH, pasirinktasKatalogas);

StringCchCat(katalogas, MAX\_PATH, TEXT("\\"));

label1->Text = gcnew String(katalogas);

// Sudarome paieškos šabloną

StringCchCopy(paieskosSablonas, MAX\_PATH, katalogas);

StringCchCat(paieskosSablonas, MAX\_PATH, pletinys);

// Atliekame paiešką

hFind = FindFirstFile(paieskosSablonas, &ffd);

if (hFind == INVALID\_HANDLE\_VALUE) { // Tikriname, gal nėra nei vieno šabloną atitinkančio failo

listBox1->Items->Add(gcnew String(L"Šabloną atitinkančių failų nerasta."));

return;

}

do {

// Sudarome pilną (su keliu) rasto duomenų failo vardą

StringCchCopy(pilnasVardas, MAX\_PATH, katalogas);

StringCchCat(pilnasVardas, MAX\_PATH, ffd.cFileName);

// naudojame failo vardą, kuris saugomas kintamajame pilnasVardas,

// pavyzdžiui, ifstream fd(pilnasVardas);

Nuskaitymas(pilnasVardas, info);

// šioje programoje tiesiog išvedame listBox1 komponente

listBox1->Items->Add(String::Format(L"{0}", gcnew String(pilnasVardas)));

} while (FindNextFile(hFind, &ffd) != 0);

FindClose(hFind);

}

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void label5\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox2->Text == L"") {

MessageBox::Show (L"Įrašykite papildomą failą", L"Neįvesta reikšmė", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

if (folderBrowserDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK && folderBrowserDialog1->SelectedPath ->Length > 0) {

listBox2->Items->Clear(); // Išvalome listBox1 komponentą

WIN32\_FIND\_DATA ffd; // Paieškai reikalingi ...

HANDLE hFind = INVALID\_HANDLE\_VALUE; // ... tarnybiniai kintamieji

// Funkcijai PtrToStringChars() reikalingas #include <vcclr.h>

pin\_ptr<const wchar\_t> pletinys = PtrToStringChars(textBox2->Text);

TCHAR katalogas[MAX\_PATH]; // Saugosime kelią iki duomenų failų

TCHAR paieskosSablonas[MAX\_PATH]; // Saugosime paieškos šabloną

TCHAR pilnasVardas[MAX\_PATH]; // Saugosime konkretaus duomenų failo pilną (su keliu) vardą

// Sudarome kelią į katalogą, kuriame sudėti duomenų failai

pin\_ptr<const wchar\_t> pasirinktasKatalogas = PtrToStringChars(folderBrowserDialog1->SelectedPath);

StringCchCopy(katalogas, MAX\_PATH, pasirinktasKatalogas);

StringCchCat(katalogas, MAX\_PATH, TEXT("\\"));

label7->Text = gcnew String(katalogas);

// Sudarome paieškos šabloną

StringCchCopy(paieskosSablonas, MAX\_PATH, katalogas);

StringCchCat(paieskosSablonas, MAX\_PATH, pletinys);

// Atliekame paiešką

hFind = FindFirstFile(paieskosSablonas, &ffd);

if (hFind == INVALID\_HANDLE\_VALUE) { // Tikriname, gal nėra nei vieno šabloną atitinkančio failo

listBox2->Items->Add(gcnew String(L"Papildomą failą atitinkančių failų nerasta."));

return;

}

do {

// Sudarome pilną (su keliu) rasto duomenų failo vardą

StringCchCopy(pilnasVardas, MAX\_PATH, katalogas);

StringCchCat(pilnasVardas, MAX\_PATH, ffd.cFileName);

// naudojame failo vardą, kuris saugomas kintamajame pilnasVardas,

// pavyzdžiui, ifstream fd(pilnasVardas);

PapildomasNuskaitymas(pilnasVardas, papild);

// šioje programoje tiesiog išvedame listBox1 komponente

listBox2->Items->Add(String::Format(L"{0}", gcnew String(pilnasVardas)));

} while (FindNextFile(hFind, &ffd) != 0);

FindClose(hFind);

}

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e) {

info->DuomenuPapildymas(papild);

info->Isvesti(Crez);

richTextBox1->LoadFile("Rezultatai.txt", RichTextBoxStreamType::PlainText);

}

private: System::Void tabPage1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void textBox3\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void button5\_Click(System::Object^ sender, System::EventArgs^ e) {

String ^sak = gcnew String(info->SkaiciuotiLyti().c\_str()); // string to String

textBox3->Text = sak;

Atrinkimas(info, galutinis, richTextBox2, textBox4);

}

private: System::Void button6\_Click(System::Object^ sender, System::EventArgs^ e) {

}

};

}

## Pradiniai duomenys ir rezultatai

**1 Duomenys.txt:**

Informatika

Aaaaa, Aaaaa, IF-1/1 1990-09-09 vyras

Bbbbbb, Bbbb, IF-2/1 1991-08-07 vyras

Cccc, CCCccc, IF-3/1 1992-06-13 vyras

Ddddddddd, D, IF-4/1 1993-04-28 vyras

Eeeee, eeEeeee, IF-5/1 1994-12-01 vyras

**2 Duomenys.txt:**

Elektros valdymas

Uuuuu, Uuuuu, S-0/1 1990-02-09 vyras

Rrrrr, Rrrr, S-2/4 1998-12-07 moteris

Kkkk, Kkkkkkk, S-3/9 1991-12-13 vyras

Sssss, Sss, S-4/3 1988-03-28 moteris

Tttttt, TT, S-5/4 1981-04-01 moteris

**3 Duomenys.txt:**

Pedagogika

Abc, Abc, P-1/1 1988-09-11 moteris

Bcd, Bcd, P-2/1 1913-08-12 moteris

Cde, Cde, P-3/1 1997-06-13 moteris

Def, Def, P-4/1 1991-04-08 moteris

Efg, Efg, P-5/1 1990-12-21 moteris

**4 Duomenys.txt:**

Vadyba

Cda, Cda, V-1/1 1990-01-01 vyras

asfaf, asgag, V-2/2 1991-02-02 moteris

Neturi, Prasmes, V-3/3 1992-03-03 vyras

Wwwww, Wwwwww, V-4/4 1993-04-04 moteris

Test, Test, V-5/5 1994-05-05 vyras

**5 Duomenys.txt:**

Dizaino Technologijos

A a a a, A a a a, DT-9/1 1991-01-01 moteris

B b b b, B b b b, DT-8/2 1992-02-02 vyras

C c c c, C c c c, DT-7/3 1993-03-03 moteris

D d d d, D d d d, DT-6/4 1994-04-04 vyras

E e e e, E e e e, DT-5/5 1995-05-05 moteris

**Pagalbinis.txt:**

Aaaaa, Aaaaa, 1 2 3 4 5

Bbbbbb, Bbbb, 2 3 4 5 6

Cccc, CCCccc, 3 4 5 8 7

Ddddddddd, D, 4 5 6 7 8

Eeeee, eeEeeee, 5 6 7 8 9

Uuuuu, Uuuuu, 6 7 8 9 10

Rrrrr, Rrrr, 7 8 7 10 1

Kkkk, Kkkkkkk, 8 9 10 1 2

Sssss, Sss, 9 10 1 2 3

Tttttt, TT, 10 1 2 3 4

Abc, Abc, 1 2 3 4 5

Bcd, Bcd, 2 3 4 10 6

Cde, Cde, 3 4 5 6 7

Def, Def, 4 5 6 7 8

Efg, Efg, 5 6 7 8 9

Cda, Cda, 6 7 8 10 10

asfaf, asgag, 7 8 9 10 1

Neturi, Prasmes, 8 9 10 1 2

Wwwww, Wwwwww, 9 10 1 2 3

Test, Test, 10 1 2 3 4

A a a a, A a a a, 1 2 3 4 5

B b b b, B b b b, 2 10 4 5 6

C c c c, C c c c, 3 4 5 6 7

D d d d, D d d d, 4 4 6 7 8

E e e e, E e e e, 5 8 7 8 9

**Rezultatai.txt:**

/------------------------------------------------------------------------------------------------/

Pavardė Vardas Grupė Metai Lytis Pažymiai Vidurkis

/------------------------------------------------------------------------------------------------/

------------------------------------------------------------------------

Informatika

Aaaaa Aaaaa IF-1/1 1990-09-09 vyras 1 2 3 4 5 3

Bbbbbb Bbbb IF-2/1 1991-08-07 vyras 2 3 4 5 6 4

Cccc CCCccc IF-3/1 1992-06-13 vyras 3 4 5 8 7 5.4

Ddddddddd D IF-4/1 1993-04-28 vyras 4 5 6 7 8 6

Eeeee eeEeeee IF-5/1 1994-12-01 vyras 5 6 7 8 9 7

------------------------------------------------------------------------

------------------------------------------------------------------------

Elektros valdymas

Uuuuu Uuuuu S-0/1 1990-02-09 vyras 6 7 8 9 10 8

Rrrrr Rrrr S-2/4 1998-12-07 moteris 7 8 7 10 1 6.6

Kkkk Kkkkkkk S-3/9 1991-12-13 vyras 8 9 10 1 2 6

Sssss Sss S-4/3 1988-03-28 moteris 9 10 1 2 3 5

Tttttt TT S-5/4 1981-04-01 moteris 10 1 2 3 4 4

------------------------------------------------------------------------

------------------------------------------------------------------------

Pedagogika

Abc Abc P-1/1 1988-09-11 moteris 1 2 3 4 5 3

Bcd Bcd P-2/1 1913-08-12 moteris 2 3 4 10 6 5

Cde Cde P-3/1 1997-06-13 moteris 3 4 5 6 7 5

Def Def P-4/1 1991-04-08 moteris 4 5 6 7 8 6

Efg Efg P-5/1 1990-12-21 moteris 5 6 7 8 9 7

------------------------------------------------------------------------

------------------------------------------------------------------------

Vadyba

Cda Cda V-1/1 1990-01-01 vyras 6 7 8 10 10 8.2

asfaf asgag V-2/2 1991-02-02 moteris 7 8 9 10 1 7

Neturi Prasmes V-3/3 1992-03-03 vyras 8 9 10 1 2 6

Wwwww Wwwwww V-4/4 1993-04-04 moteris 9 10 1 2 3 5

Test Test V-5/5 1994-05-05 vyras 10 1 2 3 4 4

------------------------------------------------------------------------

------------------------------------------------------------------------

Dizaino Technologijos

A a a a A a a a DT-9/1 1991-01-01 moteris 1 2 3 4 5 3

B b b b B b b b DT-8/2 1992-02-02 vyras 2 10 4 5 6 5.4

C c c c C c c c DT-7/3 1993-03-03 moteris 3 4 5 6 7 5

D d d d D d d d DT-6/4 1994-04-04 vyras 4 4 6 7 8 5.8

E e e e E e e e DT-5/5 1995-05-05 moteris 5 8 7 8 9 7.4

------------------------------------------------------------------------

/------------------------------------------------------------------------------------------------/

Pavardė Vardas Grupė Metai Lytis Pažymiai Vidurkis Fakultetas

/------------------------------------------------------------------------------------------------/

D d d d D d d d DT-6/4 1994-04-04 vyras 4 4 6 7 8 5.8 Dizaino Technologijos

Kkkk Kkkkkkk S-3/9 1991-12-13 vyras 8 9 10 1 2 6 Elektros valdymas

Uuuuu Uuuuu S-0/1 1990-02-09 vyras 6 7 8 9 10 8 Elektros valdymas

Ddddddddd D IF-4/1 1993-04-28 vyras 4 5 6 7 8 6 Informatika

Eeeee eeEeeee IF-5/1 1994-12-01 vyras 5 6 7 8 9 7 Informatika

Cda Cda V-1/1 1990-01-01 vyras 6 7 8 10 10 8.2 Vadyba

Neturi Prasmes V-3/3 1992-03-03 vyras 8 9 10 1 2 6 Vadyba