

**KAUNO TECHNOLOGIJOS UNIVERSITETAS INFORMATIKOS
FAKULTETAS**

OBJEKTINIS PROGRAMAVIMAS II (P175B123)
Laboratorinio darbo ataskaita

Atliko:

IFF-5/7 gr. Studentė

Viktorija Ražaitė

2016 m. vasario 24 d.

Priėmė:

Jurgis Pralgauskis

KAUNAS 2016

TURINYS

1. Rekursija	5
1.1. Darbo užduotis	5
1.2. Programos tekstas	5
1.3. Pradiniai duomenys ir rezultatai	11
1.4. Grafinės vartotojo sąsajos schema	12
1.5. Grafinės vartotojo sąsajos elementų pakeistos savybės	13
1.6. Klasių diagramos	13
1.7. Programos vartotojo vadovas	14
2. Susietasis sąrašas	15
2.1. Darbo užduotis	15
2.2. Programos tekstas	15
2.3. Pradiniai duomenys ir rezultatai	24
2.4. Grafinės vartotojo sąsajos schema	27
2.5. Grafinės vartotojo sąsajos elementų pakeistos savybės	27
2.6. Klasių diagramos	28
2.7. Programos vartotojo vadovas	28
3. Bendrinis susietasis sąrašas	29
3.1. Darbo užduotis	29
3.2. Programos tekstas	29
3.3. Pradiniai duomenys ir rezultatai	38
3.4. Grafinės vartotojo sąsajos schema	40
3.5. Grafinės vartotojo sąsajos elementų pakeistos savybės Error! Bookmark not defined.
3.6. Klasių diagramos Error! Bookmark not defined.
3.7. Programos vartotojo vadovas	40
4. Bendrinės kolekcijos	43
4.1. Darbo užduotis	43
4.2. Programos tekstas	43
4.3. Pradiniai duomenys ir rezultatai	43
4.4. Grafinės vartotojo sąsajos schema	43
4.5. Grafinės vartotojo sąsajos elementų pakeistos savybės	43
4.6. Klasių diagramos	43
4.7. Programos vartotojo vadovas	43
5. Deklaratyvusis programavimas	44
5.1. Darbo užduotis	44
5.2. Programos tekstas	44

5.3. Pradiniai duomenys ir rezultatai	44
5.4. Grafinės vartotojo sąsajos schema	44
5.5. Grafinės vartotojo sąsajos elementų pakeistos savybės	44
5.6. Klasių diagramos.....	44
5.7. Programos vartotojo vadovas	44

1. Rekursija

1.1. Darbo užduotis

LD_7. Žodžių paieška.

Parašykite programą, kuri perskaitytų iš tekstinio failo "Trecias.txt" tekstą po vieną simbolį ($1 < \text{simbolių kiekis} \leq 2000$) ir jais užpildytų masyvą $A[n,n]$. Į masyvą nerašomi eilutės pabaigos, naujos eilutės ir failo pabaigos simboliai. n parenkamas toks mažiausias, kad tekstas tilptų į kvadratinę matricą. Jei paskutinei eilutei trūksta simbolių, užpildote tarpais. Po to programa iš tekstinio failo "Zodziai.txt", kuriame kiekvienas žodis yra atskiroje eilutėje ir prasideda nuo 1 pozicijos, perskaito eilinį žodį, kurio ilgis $k \leq n/2$. Reikia nustatyti, kiek kartų šis žodis kartojasi lentelėje $A[n, n]$. Paieška turėtų būti atliekama horizontaliai iš kairės į dešinę, vertikaliai iš viršaus žemyn ir pagal dešinę diagonalę žemyn ir į dešinę (ne tik pagrindinė diagonalė). Žodžiai iš eilutės (stulpelio) į eilutę (stulpelį) nekeliami. Žodžiai nepersidengia, bet trumpesnis žodis gali būti ilgesniojo žodžio dalimi.

Ekране atspausdinkite parinktą n reikšmę ir kiekvieno žodžio pasikartojimų skaičių.

Trecias.txt	Matrica									
Berzas,sula;;sula;;klevu saldial lapasula	B	e	r	z	a	s	,	s	u	
a aula, ar suart zemes vaikai du	l	a	;	;	s	u	l	a	;	
Zodziai.txt	;	k	l	e	v	u		s	a	
Sula	l	d	i	a	l		l	a	p	
Alus	a	s	u	l	a			a		
Atsakymas			a	u	l	a	,		a	
n = 9	r			s	u	a	r	t		
sula 3	z	e	m	e	s		v	a	i	
alus 2	k	a	i		d	u				

1.2. Programos tekstas

```
using System;
using System.Collections.Generic;
using System.IO; using
System.Linq; using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class FormL1 : System.Web.UI.Page
{
    public class Simboliai
    {
        public char s { get; set; }

        public Simboliai(char s)
        {
            this.s = s;
        }
    }

    public class Konteinerine
    {
        const int Max = 45; // max stulpeliu ir eiluciu
        private Simboliai[,] Simb;        public
        Konteinerine()
        {
            Simb = new Simboliai[Max, Max];
        }        public void Deti(int i, int j,
        Simboliai a)        {
            Simb[i, j] = a;
        }        public Simboliai
        Imti(int i, int j)
        {
            return Simb[i, j];
        }
    }
}
```

```

    }
}

    const int MaxZodziu = 2000;
const int MaxMatrica = 45;
    private const string f1 = @"F:\2tras\objektinis\L1\Trecias.txt";
private const string f2 = @"F:\2tras\objektinis\L1\Zodziai.txt";    private
const string f3 = @"F:\2tras\objektinis\L1\Rezultatai.txt";
    //Konteinerine simbol = new Konteinerine();//dvimatis
    //string[,] Masyvas = new string[MaxMatrica, MaxMatrica];

protected void Page_Load(object sender, EventArgs e)
{

}

protected void TextBox1_TextChanged(object sender, EventArgs e)
{

}

    /// <summary>
    /// Skaiciuoja, kiek zodziu yra vertikaliai
    /// </summary>
    /// <param name="zodziuC">kiek yra duotuju zodziu</param>
    /// <param name="zodziuMasyvas">duotuju zodziu masyvas</param>
    /// <param name="simboliai">simboliu konteineris</param>
    /// <param name="n">matricos dydis n*n</param>
    /// <param name="ZodziuSkaicius">Rastu zodziu skaicius</param>
    /// <param name="i">parametras, kuris nurodo, kuris zodis bus ieskomas</param>
static void RecursionHORIZONTALA(int zodziuC, string[] zodziuMasyvas,
Konteinerine simboliai, int n, ref int[] ZodziuSkaicius, int i)
    {
        char[] zodRaides =
zodziuMasyvas[i].ToCharArray();        for (int k = 0; k
< n; k++) //stulpeliai ||
        {
            bool tikrinaH = true;
int nuoKurio = 0;

            for (int l = 0; l < n; l++) //eilutes --
            {
                if (zodRaides.Length <= (n - 1)) // patikrina ar tas zodis tilps
iki eilutes pabaigos
                    if ((char.ToLower(zodRaides[0])) ==
char.ToLower(simboliai.Imti(l, k).s))
                    {
                        tikrinaH = true;
                        for (int m1 = 1; m1 < zodziuMasyvas[i].Length; m1++)
                        {
                            if ((m1 + 1 <= n) && (tikrinaH))
                            {
                                if ((char.ToLower(zodRaides[m1])) ==
char.ToLower(simboliai.Imti(l + m1, k).s))
                                    nuoKurio = l + m1;
                                else tikrinaH =
false;
                            }
                        }

                        if (tikrinaH) ZodziuSkaicius[i] += 1;
                    }
            }
        }
        if ((i + 1) < zodziuC)
            RecursionHORIZONTALA(zodziuC, zodziuMasyvas, simboliai, n, ref
ZodziuSkaicius, i + 1);
    }
}

```

```

    }
    /// <summary>
    /// Skaiciuoja, kiek zodziu yra horizontaliai
    /// </summary>
    /// <param name="zodziuC">kiek yra duotuju zodziu</param>
    /// <param name="zodziuMasyvas">duotuju zodziu masyvas</param>
    /// <param name="simboliai">simboliu konteineris</param>
    /// <param name="n">matricos dydis n*n</param>
    /// <param name="ZodziuSkaicius">Rastu zodziu skaicius</param>
    /// <param name="i">parametras, kuris nurodo, kuris zodis bus ieskomas</param>
    static void RecursionVERTIKALE(int zodziuC, string[] zodziuMasyvas,
    Konteinerine simboliai, int n, ref int[] ZodziuSkaicius, int i)
    {
        bool tikrinaV = true;
        int nuoKurioV = 0;
        char[] zodRaides =
        zodziuMasyvas[i].ToCharArray();          for (int l = 0; l
        < n; l++) //stulpeliai ||
        {
            for (int k = 0; k < n; k++) //eilutes --
            {
                if (zodRaides.Length <= (n - k)) // patikrina ar tas zodis tilps
                iki stulpelio pabaigos
                if ((char.ToLower(zodRaides[0])) ==
                char.ToLower(simboliai.Imti(l, k).s))
                {
                    tikrinaV = true;
                    for (int m1 = 1; m1 < zodziuMasyvas[i].Length; m1++)
                    {
                        if ((m1 + k <= n) && (tikrinaV))
                        {
                            if ((char.ToLower(zodRaides[m1])) ==
                            char.ToLower(simboliai.Imti(l, k + m1).s))
                                nuoKurioV = k + m1;
                        }
                    }
                }
            }

            if (tikrinaV) ZodziuSkaicius[i] += 1;
        }
    }

    if ((i + 1) < zodziuC)
        RecursionVERTIKALE(zodziuC, zodziuMasyvas, simboliai, n, ref
        ZodziuSkaicius, i + 1);

}
/// <summary>
/// Skaiciuoja, kiek zodziu yra pagal diagonale
/// </summary>
/// <param name="zodziuC">kiek yra duotuju zodziu</param>
/// <param name="zodziuMasyvas">duotuju zodziu masyvas</param>
/// <param name="simboliai">simboliu konteineris</param>
/// <param name="n">matricos dydis n*n</param>
/// <param name="ZodziuSkaicius">Rastu zodziu skaicius</param>
/// <param name="i">parametras, kuris nurodo, kuris zodis bus ieskomas</param>
    static void RecursionDIAGONALE(int zodziuC, string[] zodziuMasyvas,
    Konteinerine simboliai, int n, ref int[] ZodziuSkaicius, int i)
    {
        bool tikrinaD = true;
        int nuoKurioD = 0;

```

```

        char[] zodRaides =
zodziuMasyvas[i].ToCharArray();          for (int k = 0; k
< n; k++) //stulpeliai ||                {
        for (int l = 0; l < n; l++) //eilutes --
        {
            if ((zodRaides.Length <= (n - 1)) && (zodRaides.Length <= (n -
k))) // patikrina ar tas zodis tilps iki eilutes ir stulpelio pabaigos
            if ((char.ToLower(zodRaides[0])) == char.ToLower(simboliai.Imti(l, k).s))
            {
                tikrinaD = true;
                for (int m1 = 1; m1 < zodziuMasyvas[i].Length; m1++)
                {
                    if ((m1 + 1 <= n) && (tikrinaD) && (m1 + k <= n))
                    {
                        if ((char.ToLower(zodRaides[m1])) ==
char.ToLower(simboliai.Imti(l + m1, k + m1).s))
                        nuoKurioD = l + m1;                      else tikrinaD =
false;
                    }
                }
            }

            if (tikrinaD) ZodziuSkaicius[i] += 1;
        }
    }

    if ((i + 1) < zodziuC)
        RecursionDIAGONALE(zodziuC, zodziuMasyvas, simboliai, n, ref
ZodziuSkaicius, i + 1);

}
/// <summary>
/// Paspaudus mygtuka "start" bus atliekami veiksmai nurodyti siame void'e
/// </summary>
protected void Button1_Click(object sender, EventArgs e)
{
    TextBox7.Text = " tarpas ";
    Konteinerine simboliai = new Konteinerine();
string[] zodziuMasyvas = new string[MaxZodziu];
int zodziuC;          int n;
    Skaityti(simboliai, zodziuMasyvas, out zodziuC, out n);
    //-----surasys, kiek kartu buvo rastas kiekvienas zodis
int[] ZodziuSkaicius = new int[zodziuC];          for (int i = 0;
i < zodziuC; i++)
    {
        ZodziuSkaicius[i] = 0;
    }

    //-----HORIZONTALIAI
int iHoriz = 0;
    RecursionHORIZONTALALE(zodziuC, zodziuMasyvas, simboliai, n, ref
ZodziuSkaicius, iHoriz);

    //-----VERTIKALIAI
int iVertik = 0;
    RecursionVERTIKALE(zodziuC, zodziuMasyvas, simboliai, n, ref
ZodziuSkaicius, iVertik);

    //-----PAGAL DIAGONALE
int iDiagon = 0;
    RecursionDIAGONALE(zodziuC, zodziuMasyvas, simboliai, n, ref
ZodziuSkaicius, iDiagon);

```



```

        string eill = "kvadratine matrica: " + n + " x " + n + "\r\n";
for (int i = 0; i < zodziuC; i++)
{
    eill += "zodis '" + zodziuMasyvas[i] + "' pasikartoja " +
ZodziuSkaicius[i] + " kartus";
eill += "\r\n";
}

using (var writer = File.AppendText(f3))
{
    writer.WriteLine();
    writer.WriteLine("Gauti rezultatai: ");
writer.WriteLine();
    writer.WriteLine(eill);
}

TextBox6.Text = "Rezultatai:";
TextBox3.Text = eill;
}

/// <summary>
/// Skaito duomenis is duom failu
/// </summary>
/// <param name="simboliai">simboliu konteineris</param>
/// <param name="zodziuMasyvas">duotuju zodziu masyvas</param>
/// <param name="n">matricos dydis n*n</param>
/// <param name="zodziuC">kiek yra duotuju zodziu</param>
public void Skaityti(Konteinerine simboliai, string[] zodziuMasyvas, out int
zodziuC, out int nn)
{
    int MaxEil = 50;
string line;
    int
simboliusK = 0;
    string[] eilutes = new string[MaxEil];
    int teksEil = 0; // kiek duotam faile yra eiluciu

    using (StreamReader reader = new StreamReader(f1))
    {
        while ((line = reader.ReadLine()) != null)
        {
            simboliusK += line.Length;
eilutes[teksEil++] = line;
        }
    }

    //-----randa kvadratines matricos n x n
double n = Math.Sqrt(simboliusK);
    n =
Math.Round(n, 1);
    if (n % Math.Round(n, 0) == 0) n = Math.Round(n, 0);
else
    {
        if (n % Math.Round(n, 0) >= 0.5) n = Math.Round(n, 0);
else n = n = Math.Round(n, 0) + 1;
    }

    //-----sureda teksta i viena eilute
nn = Convert.ToInt32(n);
    string failas =
"";

    for (int i = 0; i < teksEil; i++)
    {
        failas = failas + eilutes[i];
    }

    //-----slpit'ina kas simboli ir deda i kvadratine matrica
string[] sym = new string[simboliusK];

    char[] symbol = failas.ToCharArray();

```

```

        int count = 0; char a;
        for (int jj = 0; jj < n; jj++)
        {
            for (int ii = 0; ii < n; ii++)
            {
                if (count < failas.Length)
                {
                    a = symbol[count];
                }
                else
                {
                    a = ' ';
                }
                Simboliai ss = new Simboliai(a);
                simboliai.Deti(ii, jj, ss);
                count++;
            }
        }

        //-----skaitymo antro failo zodzius
        zodziuC = 0;
        string line1; string eilZod = "";
        using (StreamReader reader = new StreamReader(f2))
        {
            while ((line1 = reader.ReadLine()) != null)
            {
                if (line1.Length <= n / 2)
                {
                    zodziuMasyvas[zodziuC++] = line1;
                    eilZod += line1 + "\r\n";
                }
            }
        }
        if (File.Exists(f3))
        {
            File.Delete(f3);
        }
        //-----suraso pradinis duomenis i rezultatu faila
        string eil = "";
        using (var writer = File.AppendText(f3))
        {
            writer.WriteLine("Duotas tekstas: ");
            writer.WriteLine();
            for (int i = 0; i < teksEil; i++)
            {
                writer.WriteLine(eilutes[i]);
            }
            writer.WriteLine("-----");
            writer.WriteLine();
            writer.WriteLine("Zodziu matrica ");
            writer.WriteLine();
            for (int i = 0; i < n; i++)
            {
                for (int j = 0; j < n; j++)
                {
                    writer.Write(" " + simboliai.Imti(j, i).s);
                }
                eil += " " + simboliai.Imti(j, i).s;
                writer.WriteLine();
            }
            eil += "\r\n";
            writer.WriteLine("-----");
            writer.WriteLine();
            writer.WriteLine("Duoti zodziai:");
            writer.WriteLine();
            for (int i = 0; i < zodziuC; i++)
            {
                writer.WriteLine(zodziuMasyvas[i]);
            }
            writer.WriteLine("-----");
            TextBox1.Text = eil;
            TextBox2.Text = "Pradiniai duomenys";
            TextBox5.Text = "Duoti zodziai";
            TextBox4.Text = eilZod;
        }
    }

```


Failas: Trecias.txt

Berzas,sula;;sula;;klevu saldial lapasula
a aula, ar suart zemes vaikai du

Failas: Zodziai.txt

Sula
Alus

Rezultatai

Failas: Rezultatai.txt

Duotas tekstas:

Berzas,sula;;sula;;klevu saldial lapasula
a aula, ar suart zemes vaikai du -----

Zodziu matrica

B e r z a s , s u
l a ; ; s u l a ; ;
k l e v u s a l d
i a l l a p a s u
l a a a u
l a , a r s u
a r t z e m e s
v a i k a i d u

Duoti zodziai:

Sula
Alus

Gauti rezultatai:

kvadratine matrica: 9 x 9 zodis
'Sula' pasikartoja 3 kartus zodis
'Alus' pasikartoja 2 kartus

1.4. Grafinės vartotojo sąsajos schema

START

LD1_7 ŽODŽIŲ PAIEŠKA

Pradiniai duomenys

Duoti zodziai

Rezultatai:

kvadratinė matrica: 9 x 9
zodis 'Sula' pasikartoja 3 kartus
zodis 'Alus' pasikartoja 2 kartus

1.5. Grafinės vartotojo sąsajos elementų pakeistos savybės

Komponentas	ID	text	Back Color	OnClick	TextMode
Button	Button1	START	#ADDFFF	Button1_Click	
TextBox	TextBox1		#FFCCFF		MultiLine
TextBox	TextBox2				
TextBox	TextBox3				
TextBox	TextBox4				
TextBox	TextBox5		#FFCCFF		MultiLine
TextBox	TextBox6				

1.6. Klasijų diagramos

Veiksmai
<pre>+RecursionHORIZONTALA(in zodziuC:integer, in zodziuMasyvas:string[], in simboliai:Konteinerine, in n:integer, inout ZodziuSkaicius:integer[], in i:integer) +RecursionVERTIKALE(in zodziuC:integer, in zodziuMasyvas:string[], in simboliai:Konteinerine, in n:integer, inout ZodziuSkaicius:integer[], in i:integer) +RecursionDIAGONALE(in zodziuC:integer, in zodziuMasyvas:string[], in simboliai:Konteinerine, in n:integer, inout ZodziuSkaicius:integer[], in i:integer) +Page_Load(in sender:object, in e:EventArgs) +Page_Init(in sender:object, in e: EventArgs) +TextBox1_TextChanged(in sender:object, in e: EventArgs) +TextBox2_TextChanged(in sender:object, in e: EventArgs) +Button1_Click(in sender:object, in e: EventArgs)</pre>
Skaityti
<pre>+ Skaityti() {query}</pre>

1.7. Programos vartotojo vadovas

Sukuriamas tekstinis duomenų failas Trecias.txt ir paršomas tekstas. Tekstas surašomas į $n \times n$ matricą, n parenkamos toks mažiausias, kad tekstas tilptų į kvadratinę matricą, jei trūksta simbolių, užpildoma tarpais. Užpildžius duomenų failą, galima įjungti programą. Įjungę programą paspaudžiame mygtuką „START“, kuris į ekraną išveda pradinis duomenis skiltyje „Pradiniai duomenys“ ir išveda rezultatus skiltyje „Rezultatai“. Baigę savo darbą, galime išeiti iš programos spausdami raudoną x mygtuką dešiniam viršutiniame programos kampe. Įvykdžius programą, taip pat sukuriamas tekstinis rezultatų failas Rezultatai.txt.

2. Susietasis sąrašas

2.1. Darbo užduotis

LD_7. Moduliai.

Studentai renka modulius. Už modulius yra atsakingi dėstytojai. Dėstytojas gali būti atsakingas už keletą modulių.

Suraskite, kuris dėstytojas turi daugiausiai pasirinktų modulių.

Nustatykite, ar visų grupių studentai pasirinko šio dėstytojo modulius.

Atspausdinkite grupes, kurių studentai nepasirinko šio dėstytojo modulių.

Duomenys:

- Tekstiniame faile U7a.txt duota informacija apie studentų pasirenkamus modulius: modulio pavadinimas, studento pavardė, vardas, grupė.
- Tekstiniame faile U7b.txt duota informacija apie modulius: modulio pavadinimas, atsakingo dėstytojo pavardė, vardas, kreditų kiekis.

Spausdinamas sąrašas turi būti surikiuotas abėcėlės tvarka.

Sudarykite nurodyto modulio (įvedamas klaviatūra) pasirinkusių studentų sąrašą.

2.2. Programos tekstas

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class L2 : System.Web.UI.Page
{
    public class Studentas
    {
        public string modulis { get; set; }
        public string pavarde { get; set; }
        public string vardas { get; set; }
        public string grupe { get; set; }
        public Studentas(string modulis, string pavarde, string vardas, string
grupe)
        {
            this.modulis = modulis;
            this.pavarde = pavarde;
            this.vardas = vardas;
            this.grupe = grupe;
        }
        static public bool operator >(Studentas pir, Studentas ant)
        {
            int ip = String.Compare(pir.grupe, ant.grupe,
StringComparison.CurrentCulture);
            int ip1 = String.Compare(pir.pavarde, ant.pavarde,
StringComparison.CurrentCulture);
            int ip2 = String.Compare(pir.vardas, ant.vardas,
StringComparison.CurrentCulture);
            return ip > 0 || ip == 0 && ip1 > 0 || ip1 == 0 && ip2 > 0;
        }
        static public bool operator <(Studentas pir, Studentas ant)
        {
            int ip = String.Compare(pir.grupe, ant.grupe,
StringComparison.CurrentCulture);
            int ip1 = String.Compare(pir.pavarde, ant.pavarde,
StringComparison.CurrentCulture);
```

```

        int ip2 = String.Compare(pir.vardas, ant.vardas,
StringComparison.CurrentCulture);
        return ip < 0 || ip == 0 && ip1 < 0 || ip1 == 0 && ip2 < 0;
    }
    public override string ToString()
    {
        string eilute;
        eilute = string.Format("{0,-15} {1,-20} {2, -15} {3,-10}", modulis,
pavarde, vardas, grupe);
        return eilute;
    }
}

public class Destytojas
{
    public string modulis { get; set; }
    public string pavarde { get; set; }
    public string vardas { get; set; }
    public int kreditai { get; set; }
    public Destytojas(string modulis, string pavarde, string vardas, int
kreditai)
    {
        this.modulis = modulis;
        this.pavarde = pavarde;
        this.vardas = vardas;
        this.kreditai = kreditai;
    }
    public string ToString1()
    {
        string eilute;
        eilute = string.Format("{0,-15} {1,-15} {2, -15} {3,-10}", modulis,
pavarde, vardas, kreditai);
        return eilute;
    }
}

public sealed class Mazgas1
{
    public Studentas DuomStud { get; set; }
    public Mazgas1 Kitas { get; set; }
    public Mazgas1(Studentas st, Mazgas1 adr)
    {
        DuomStud = st;
        Kitas = adr;
    }
}

public sealed class Mazgas2
{
    public Destytojas DuomDest { get; set; }
    public Mazgas2 Kitas { get; set; }
    public Mazgas2(Destytojas ds, Mazgas2 adr)
    {
        DuomDest = ds;
        Kitas = adr;
    }
}

public sealed class Sarasas1 //Studentu sarasas
{
    private Mazgas1 pr; //saraso pradzia
    private Mazgas1 pb; //saraso pabaiga
    private Mazgas1 d; //saraso sasajai
    public Sarasas1()
    {
        this.pr = null;
    }
}

```



```

        this.pb = null;
        this.d = null;
    }
    public Studentas ImtiDuomenis()
    {
        return d.DuomStud;
    }
    public Mazgas1 GautiPirma()
    {
        return pr;
    }
    public Mazgas1 GautiPaskutini()
    {
        return pb;
    }
    public void DetiDuomenisT(Studentas naujas)
    {
        var dd = new Mazgas1(naujas, null);
        if (pr != null)
        {
            pb.Kitas = dd;
            pb = dd;
        }
        else
        {
            pr = dd;
            pb = dd;
        }
    }
}

public sealed class Sarasas2 //Destytoju sarasas
{
    private Mazgas2 pr; //saraso pradzia
    private Mazgas2 pb; //saraso pabaiga
    private Mazgas2 d; //saraso sasajai
    public Sarasas2()
    {
        this.pr = null;
        this.pb = null;
        this.d = null;
    }
    public Destytojas ImtiDuomenis()
    {
        return d.DuomDest;
    }
    public Mazgas2 GautiPirma()
    {
        return pr;
    }
    public Mazgas2 GautiPaskutini()
    {
        return pb;
    }
    public void DetiDuomenis2T(Destytojas naujas)
    {
        var dd = new Mazgas2(naujas, null);
        if (pr != null)
        {
            pb.Kitas = dd;
            pb = dd;
        }
        else
        {
            pr = dd;
            pb = dd;
        }
    }
}

```

```

    }
}

protected void Page_Load(object sender, EventArgs e)
{
    Sarasas1 Studentai = new Sarasas1();
    Sarasas2 Destytojai = new Sarasas2();

    protected void Button1_Click(object sender, EventArgs e)
    {
        string CD1 = @"F:\2tras\objektinis\L2\U7a.txt"; //studentu duomenu
failas
        string CD2 = @"F:\2tras\objektinis\L2\U7b.txt"; //destytoju duomenu
failas
        string CR = @"F:\2tras\objektinis\L2\REZ.txt"; //rezultatu duomenu
failas
        if (File.Exists(CR))
            File.Delete(CR);
        SkaitytiStud(CD1, Studentai);
        SkaitytiDest(CD2, Destytojai);
        string destytojasMaxVardas; string destytojasMaxPavarde; //destytojas
turintis daugiausiai moduliu
        DestytojasMaxMod(Studentai, Destytojai, out destytojasMaxVardas, out
destytojasMaxPavarde);
        SpausdintiRezultatus(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde, CR);
        GrupesKurNepasirinko(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde, CR);
    }
}
/// <summary>
/// Nuskaito studentu duomenu faila
/// </summary>
/// <param name="fv">studentu failas</param>
/// <param name="Studentai">Studentu sarasas</param>
static void SkaitytiStud(string fv, Sarasas1 Studentai)
{
    string line;
    using (var reader = new StreamReader(fv))
    {
        while ((line = reader.ReadLine()) != null)
        {
            var v = line.Split(' ');
            var student = new Studentas(v[0], v[1], v[2], v[3]);
            Studentai.DetiDuomenisT(student);
        }
    }
}
/// <summary>
/// Nuskaito destytoju duomenu faila
/// </summary>
/// <param name="fv">destytoju failas</param>
/// <param name="Destytojai">Destytoju sarasas</param>
static void SkaitytiDest(string fv, Sarasas2 Destytojai)
{
    string line;
    using (var reader = new StreamReader(fv))
    {
        while ((line = reader.ReadLine()) != null)
        {
            var v = line.Split(' ');
            var destyt = new Destytojas(v[0], v[1], v[2], int.Parse(v[3]));
            Destytojai.DetiDuomenis2T(destyt);
        }
    }
}

```

```

    }
}
/// <summary>
/// suranda destytoja, turinti daugiausiai modulių
/// </summary>
/// <param name="Studentai">Studentų sąrašas</param>
/// <param name="Destytojai">Destytojų sąrašas</param>
/// <param name="destytojasMaxVardas">daugiausiai modulių turincio destytojo
vardas</param>
/// <param name="destytojasMaxPavarde">daugiausiai modulių turincio destytojo
pavarde</param>
static void DestytojasMaxMod(Sarasas1 Studentai, Sarasas2 Destytojai, out
string destytojasMaxVardas, out string destytojasMaxPavarde)
{
    int max = 0;
    destytojasMaxVardas = null;
    destytojasMaxPavarde = null;
    for (Mazgas2 g = Destytojai.GautiPirma(); g != null; g = g.Kitas)
    {
        int laik = 0;
        for (Mazgas1 d = Studentai.GautiPirma(); d != null; d = d.Kitas)
        {
            if (g.DuomDest.modulis == d.DuomStud.modulis)
            {
                laik += 1;
            }
        }
        if (max < laik)
        {
            destytojasMaxVardas = g.DuomDest.vardas;
            destytojasMaxPavarde = g.DuomDest.pavarde;
            max = laik;
        }
    }
}
/// <summary>
/// Suranda ar visu grupių studentai pasirinko šio destytojo modulius
/// </summary>
/// <param name="Studentai">Studentų sąrašas</param>
/// <param name="Destytojai">Destytojų sąrašas</param>
/// <param name="destytojasMaxVardas">daugiausiai modulių turincio destytojo
vardas</param>
/// <param name="destytojasMaxPavarde">daugiausiai modulių turincio destytojo
pavarde</param>
/// <returns></returns>
static bool ArVisuPasirinko(Sarasas1 Studentai, Sarasas2 Destytojai, string
destytojasMaxVardas, string destytojasMaxPavarde)
{
    for (Mazgas2 g = Destytojai.GautiPirma(); g != null; g = g.Kitas)
    {
        if ((destytojasMaxVardas == g.DuomDest.vardas) &&
(destytojasMaxPavarde == g.DuomDest.pavarde))
        {
            string modulis = g.DuomDest.modulis;
            for (Mazgas1 d = Studentai.GautiPirma(); d != null; d = d.Kitas)
            {
                if (modulis == d.DuomStud.modulis)
                {
                    int visi = 0; int ne = 0;
                    for (Mazgas1 d2 = d; d2 != null; d2 = d2.Kitas)
                    {
                        if (d.DuomStud.grupe == d2.DuomStud.grupe)
                        {
                            visi += 1;
                            if (d2.DuomStud.modulis != modulis)
                                ne += 1;
                        }
                    }
                }
            }
        }
    }
}

```

```

        }
    }
    if (ne == visi)
        return false;
    }
}

return true;
}

/// <summary>
/// suranda grupes, kuriu bent vienas studentas nepasirinko sio destytojo
modulio
/// </summary>
/// <param name="Studentai">Studentu sarasas</param>
/// <param name="Destytojai">Destytoju sarasas</param>
/// <param name="destytojasMaxVardas">daugiausiai modulių turincio destytojo
vardas</param>
/// <param name="destytojasMaxPavarde">daugiausiai modulių turincio destytojo
pavarde</param>
/// <param name="fr">rezultatu failas</param>
public void GrupesKurNepasirinko(Sarasas1 Studentai, Sarasas2 Destytojai,
string destytojasMaxVardas, string destytojasMaxPavarde, string fr)
{
    Sarasas1 StudentaiGR = new Sarasas1();
    StudentaiGR = Studentai;
    //rusiuoja pagal grupes, pavardes ir vardus

    //jei grupes yra nors vienas nepasirinkes modulio, atspauszinti
    //jei visa grupe pasirinkusi ta moduli, pasalinti grupe

    for (Mazgas2 g = Destytojai.GautiPirma(); g != null; g = g.Kitas)
    {
        if ((destytojasMaxVardas == g.DuomDest.vardas) &&
            (destytojasMaxPavarde == g.DuomDest.pavarde))
        {
            string modulis = g.DuomDest.modulis;
            for (Mazgas1 d = Studentai.GautiPirma(); d != null; d = d.Kitas)
            {
                if (modulis == d.DuomStud.modulis)
                {
                    int visi = 0; int ne = 0; string grupe = "";
                    for (Mazgas1 d2 = d; d2 != null; d2 = d2.Kitas)
                    {
                        if (d.DuomStud.grupe == d2.DuomStud.grupe)
                        {
                            visi += 1;
                            if (d2.DuomStud.modulis == modulis)
                            {
                                ne += 1;
                                grupe = d.DuomStud.grupe;
                            }
                        }
                    }
                    if (ne == visi)
                    {
                        SalintiGrupe(StudentaiGR, grupe);
                    }
                }
            }
        }
    }

    Rikiuoti(StudentaiGR);
    SpausdintiGrupes(StudentaiGR, fr);
}

```

```

}
/// <summary>
/// Surikiuoja nauja studentu sarasas pagal grupe, pavarde ir vardą
/// </summary>
/// <param name="StudentaiGR">naujas studentu sarasas</param>
static void Rikiuoti(Sarasas1 StudentaiGR)
{
    for (Mazgas1 d1 = StudentaiGR.GautiPirma(); d1 != null; d1 = d1.Kitas)
    {
        Mazgas1 max = d1;
        for (Mazgas1 d2 = d1; d2 != null; d2 = d2.Kitas)
        {
            if (d2.DuomStud < max.DuomStud)
                max = d2;
            Studentas stud = d1.DuomStud;
            d1.DuomStud = max.DuomStud;
            max.DuomStud = stud;
        }
    }
}
/// <summary>
/// pasalina grupes, kuriu visi studentai pasirinke sio destytojo modulius
/// </summary>
/// <param name="StudentaiGR">Surikiuotu studentu sarasas</param>
/// <param name="grupe">grupe, kuria reikia pasalinti</param>
static void SalintiGrupe(Sarasas1 StudentaiGR, string grupe)
{
    for (Mazgas1 d = StudentaiGR.GautiPirma(); d != null; d = d.Kitas)
    {
        if (d.DuomStud.grupe == grupe)
        {
            if (d.Kitas != null)
            {
                Mazgas1 s = d.Kitas;
                d.DuomStud = s.DuomStud;
                d.Kitas = s.Kitas;
            }
            else
                PasalintiPaskutini(StudentaiGR);
        }
    }
}
/// <summary>
/// pasalina paskutini elemanta
/// </summary>
/// <param name="StudentaiGR">surikiuotu studentu sarasas</param>
static void PasalintiPaskutini(Sarasas1 StudentaiGR)
{
    Mazgas1 k = StudentaiGR.GautiPirma();
    for (Mazgas1 d = StudentaiGR.GautiPirma(); d != null; d = d.Kitas)
    {
        k = d.Kitas;
        if(k.Kitas == null)
        {
            d.Kitas = null;
            StudentaiGR.DetiDuomenisT(d.DuomStud);
        }
    }
}
/// <summary>
/// spausdinti rezultatus
/// </summary>
/// <param name="StudentaiGR">surikiuotu studentu sarasas</param>
/// <param name="fr">rezultatu failas</param>
public void SpausdintiGrupes(Sarasas1 StudentaiGR, string fr)
{

```

```

        string line = "";
        using (var writer = File.AppendText(fr))
        {
            writer.WriteLine();
            writer.WriteLine("Grupes, kuriu studentai nepasirinko sio destytojo
moduliu:");
            Label5.Text = "Grupes, kuriu studentai nepasirinko sio destytojo
moduliu:";
            line += "Modulis          Pavarde          Vardas          Grupe" +
"\r\n" + "\r\n";
            for (Mazgas1 d = StudentaiGR.GautiPirma(); d != null; d= d.Kitas)
            {
                writer.WriteLine(d.DuomStud.ToString());
                line += d.DuomStud.ToString() + "\r\n";
            }
            TextBox3.Text = line;
        }
        protected void Button2_Click(object sender, EventArgs e)
        {
            string CD1 = @"F:\2tras\objektinis\L2\U7a.txt"; //studentu duomenu
failas
            string CR = @"F:\2tras\objektinis\L2\REZ.txt"; //rezultatu duomenu
failas
            SkaitytiStud(CD1, Studentai);
            Label4.Text = "Iveskite modulio pavadinima ir dar karta paspauskite
antraji 'START' mygtuka";
            string modulisVIP = TextBox1.Text;
            Sarasas1 NewStudentai = new Sarasas1();
            SudarytiStudentuSarasa(NewStudentai, modulisVIP, Studentai);
            if(modulisVIP != "")
                SpausdintiRez2(NewStudentai, modulisVIP, CR);
        }
        /// <summary>
        /// sudaromas studentu sarasas pagal ivesta klaviatura moduli
        /// </summary>
        /// <param name="NewStudentai">naujas studentu sarasas</param>
        /// <param name="modulisVIP">Ivestas modulis</param>
        /// <param name="Studentai">studentu sarasas</param>
        public void SudarytiStudentuSarasa(Sarasas1 NewStudentai, string modulisVIP,
Sarasas1 Studentai)
        {
            string eil = "";
            if (modulisVIP != "")
            {
                for (Mazgas1 d = Studentai.GautiPirma(); d != null; d = d.Kitas)
                {
                    if (d.DuomStud.modulis == modulisVIP)
                    {
                        NewStudentai.DetiDuomenisT(d.DuomStud);
                        eil += d.DuomStud + "\r\n";
                    }
                }
            }
            else
                TextBox2.Text = "Neteisingas modulio pavadinimas";
            TextBox2.Text = eil;
        }
        /// <summary>
        /// spausdina gautus rezultatus
        /// </summary>
        /// <param name="Studentai">studentu sarasas</param>
        /// <param name="Destytojai">destytoju sarasas</param>
        /// <param name="destytojasMaxVardas">daugiausiai modulių turincio destytojo
vardas</param>

```

```

    /// <param name="destytojasMaxPavarde">daugiausiai modulių turincio destytojo
pavarde</param>
    /// <param name="fr">rezultatu failas</param>
    public void SpausdintiRezultatus(Sarasas1 Studentai, Sarasas2 Destytojai,
string destytojasMaxVardas, string destytojasMaxPavarde, string fr)
    {
        using (var writer = File.AppendText(fr))
        {
            ///--- pradiniu doumenu spausdinimas
            writer.WriteLine("-----PRADINIAI-DUOMENYS-----");
        };

        writer.WriteLine();
        writer.WriteLine("-----Studentu-duomenu-failas-----");
        for (Mazgas1 d = Studentai.GautiPirma(); d != null; d = d.Kitas)
        {
            writer.WriteLine(d.DuomStud.ToString());
        }
        writer.WriteLine();
        writer.WriteLine("-----Destytoju-duomenu-failas-----");
        for (Mazgas2 d = Destytojai.GautiPirma(); d != null; d = d.Kitas)
        {
            writer.WriteLine(d.DuomDest.ToString1());
        }
        writer.WriteLine("-----");
        writer.WriteLine();
        ///-----
        writer.WriteLine();
        writer.WriteLine("Daugiausiai pasirinktu modulių turi destytojas - {0}
{1}", destytojasMaxVardas, destytojasMaxPavarde);
        Label1.Text = "Daugiausiai pasirinktu modulių turi destytojas - " +
destytojasMaxVardas + " " + destytojasMaxPavarde;
        writer.WriteLine();
        if (ArVisuPasirinko(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde))
        {
            writer.WriteLine("Ne visu grupių studentai pasirinko šio destytojo
modulius.");
            Label2.Text = "Visu grupių studentai pasirinko šio destytojo
modulius.";
        }
        if (!ArVisuPasirinko(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde))
        {
            writer.WriteLine("Ne visu grupių studentai pasirinko šio destytojo
modulius.");
            Label2.Text = "Ne visu grupių studentai pasirinko šio destytojo
modulius.";
        }
    }
}

    /// <summary>
    /// spausdinami rezultatai po įvedamo modulio klaviatura
    /// </summary>
    /// <param name="NewStudentai">naujas studentu sarasas</param>
    /// <param name="modulisVIP">Ivestas modulis</param>
    /// <param name="fr">rezultatu failas</param>
    public void SpausdintiRez2(Sarasas1 NewStudentai, string modulisVIP, string
fr)
    {
        using (var writer = File.AppendText(fr))
        {
            writer.WriteLine();
            writer.WriteLine("-----");
            writer.WriteLine("Ivestas modulio pavadinimas: {0}", modulisVIP);
            writer.WriteLine("Šio modulio studentų sarasas:");
            for (Mazgas1 d = NewStudentai.GautiPirma(); d != null; d = d.Kitas)

```

```

        {
            writer.WriteLine(d.DuomStud.ToString());
        }
    }
}

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="L2.aspx.cs" Inherits="L2"
%>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        #form1 {
            height: 816px;
        }
    </style>
</head>
<body style="background-color:#ADDFFF;">
    <form id="form1" runat="server">
        <div>

            </div>
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="START" BackColor="Blue" BorderColor="White" />
            <br />
            <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
            &nbsp;
            <br />
            <asp:Label ID="Label2" runat="server" Text="Label"></asp:Label>
            &nbsp;
            <br />
            <br />
            <asp:Label ID="Label5" runat="server" Text="Label"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox3" runat="server" Height="231px" Width="556px"
BackColor="#FFCCFF" TextMode="MultiLine"></asp:TextBox>
            <br />
            <br />
            <asp:Button ID="Button2" runat="server" OnClick="Button2_Click"
Text="START" BackColor="Blue" />
            <br />
            <asp:Label ID="Label4" runat="server" Text="Label"></asp:Label>
            &nbsp;<p>
            <asp:TextBox ID="TextBox1" runat="server" Height="30px" Width="187px"
BorderColor="#FFCCFF" TextMode="MultiLine"></asp:TextBox>
            </p>
            <p>
            <asp:TextBox ID="TextBox2" runat="server" Height="144px" Width="563px"
BackColor="#FFCCFF" TextMode="MultiLine"></asp:TextBox>
            </p>
        </form>
    </body>
</html>

```

2.3. Pradiniai duomenys ir rezultatai

Pradiniai duomenys

Failas: U7a.txt

Programavimas Tomaitis Tomas IFF-5/4

Matematika Pjuklaite Ieva IFF-5/2
 Psichologija Puodzius Jonas IFF-5/3
 Programavimas Lygiauskas Vladimiras IFF-5/1
 Programavimas Aiseviciute Aiste IFF-5/4
 Medijos Pukutis Pukis IFF-5/5
 Programavimas Stasys Girnaitis IFF-5/4
 Programavimas Gaile Pagalvyte IFF-5/2
 Medijos Ignas Lovauskas IFF-5/2
 Programavimas Diana Paveikslaitė IFF-5/1
 Matematika Dainius Lentauskas IFF-5/2
 Programavimas Vytenis Deziauskas IFF-5/3
 Psichologija Raigardas Knygius IFF-5/1
 Programavimas Giedrius Palangiauskas IFF-5/4
 Psichologija Audrius Dalgiauskas IFF-5/3
 Programavimas Jomante Deklaityte IFF-5/4
 Matematika Antanas Stiklius IFF-5/3

Failas: U7b.txt

Programavimas Jonas Jonaitis 6
 Medijos Arnas Sofauskas 5
 Informacines Radvile Stalciukaite 7
 Matematika Gintaras Grebliauskas 8
 Skaitmenine Arnas Sofauskas 4
 Psichologija Radvile Stalciukaite 7

Rezultatai

Failas: REZ.txt

-----PRADINIAI-DUOMENYS-----

-----Studentu-duomenu-failas-----

Programavimas	Tomaitis	Tomas	IFF-5/4
Matematika	Pjuklaite	Ieva	IFF-5/2
Psichologija	Puodzius	Jonas	IFF-5/3
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Programavimas	Aiseviciute	Aiste	IFF-5/4
Medijos	Pukutis	Pukis	IFF-5/5
Programavimas	Stasys	Girnaitis	IFF-5/4
Programavimas	Gaile	Pagalvyte	IFF-5/2
Medijos	Ignas	Lovauskas	IFF-5/2
Programavimas	Diana	Paveikslaitė	IFF-5/1
Matematika	Dainius	Lentauskas	IFF-5/2
Programavimas	Vytenis	Deziauskas	IFF-5/3
Psichologija	Raigardas	Knygius	IFF-5/1
Programavimas	Giedrius	Palangiauskas	IFF-5/4
Psichologija	Audrius	Dalgiauskas	IFF-5/3
Programavimas	Jomante	Deklaityte	IFF-5/4
Matematika	Antanas	Stiklius	IFF-5/3

-----Destytoju-duomenu-failas-----

Programavimas	Jonas	Jonaitis	6
Medijos	Arnas	Sofauskas	5
Informacines	Radvile	Stalciukaite	7
Matematika	Gintaras	Grebliauskas	8
Skaitmenine	Arnas	Sofauskas	4
Psichologija	Radvile	Stalciukaite	7

Daugiausiai pasirinktu moduliu turi destytojas - Jonaitis Jonas

Nevisu grupiu studentai pasirinko sio destytojo modulius.

Grupes, kuriu studentai nepasirinko sio destytojo moduliu:

Psichologija	Raigardas	Knygius	IFF-5/1
Programavimas	Diana	Paveikslaitė	IFF-5/1
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Matematika	Dainius	Lentauskas	IFF-5/2
Programavimas	Gaile	Pagalvyte	IFF-5/2
Medijos	Ignas	Lovauskas	IFF-5/2
Matematika	Pjuklaite	Ieva	IFF-5/2
Matematika	Antanas	Stiklius	IFF-5/3
Psichologija	Audrius	Dalgiauskas	IFF-5/3
Programavimas	Vytenis	Deziauskas	IFF-5/3
Psichologija	Puodzius	Jonas	IFF-5/3
Medijos	Pukutis	Pukis	IFF-5/5

Ivestas modulio pavadinimas: Programavimas

Sio modulio studentu sarasas:

Programavimas	Tomaitis	Tomas	IFF-5/4
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Programavimas	Aiseviciute	Aiste	IFF-5/4
Programavimas	Stasys	Girnaitis	IFF-5/4
Programavimas	Gaile	Pagalvyte	IFF-5/2
Programavimas	Diana	Paveikslaitė	IFF-5/1
Programavimas	Vytenis	Deziauskas	IFF-5/3
Programavimas	Giedrius	Palangiauskas	IFF-5/4
Programavimas	Jomante	Deklaityte	IFF-5/4

2.4. Grafinės vartotojo sąsajos schema

START

Daugiausiai pasirinktu moduliui turi destytojas - Jonaitis Jonas

Visu grupių studentai pasirinko šio destytojo modulius.

Grupės, kurių studentai nepasirinko šio destytojo modulių:

Psichologija	Raigardas	Knygius	IFF-5/1
Programavimas	Diana	Paveikslaitė	IFF-5/1
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Matematika	Dainius	Lentauskas	IFF-5/2
Programavimas	Gaile	Pagalvyte	IFF-5/2
Medijos	Ignas	Lovauskas	IFF-5/2
Matematika	Pjuklaite	Ieva	IFF-5/2
Matematika	Antanas	Stiklius	IFF-5/3
Psichologija	Audrius	Dalgiauskas	IFF-5/3
Programavimas	Vytenis	Deziauskas	IFF-5/3
Psichologija	Puodzius	Jonas	IFF-5/3
Medijos	Pukutis	Pukis	IFF-5/5

START

Iveskite modulio pavadinimą ir dar kartą paspauskite antrąjį 'START' mygtuką

Programavimas

Programavimas	Tomaitis	Tomas	IFF-5/4
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Programavimas	Aiseviciute	Aiste	IFF-5/4
Programavimas	Stasys	Girnaitis	IFF-5/4
Programavimas	Gaile	Pagalvyte	IFF-5/2
Programavimas	Diana	Paveikslaitė	IFF-5/1
Programavimas	Vytenis	Deziauskas	IFF-5/3
Programavimas	Giedrius	Palangiauskas	IFF-5/4
Programavimas	Jomante	Deklaityte	IFF-5/4

2.5. Grafinės vartotojo sąsajos elementų pakeistos savybės

Komponentas	ID	text	Back Color	OnClick	TextMode
Button	Button1	START	Blue	Button1_Click	
Label	Label1				
Label	Label2				
Label	Label5				
TextBox	TextBox3		#FFCCFF		MultiLine
Button	Button2	START	Blue	Button2_Click	
Label	Label4				
TextBox	TextBox1		#FFCCFF		MultiLine
TextBox	TextBox2		#FFCCFF		MultiLine

2.6. Klasių diagramos

1.6. Klasių diagramos

Veiksmai
<pre>+Page_Load(in sender:object, in e:EventArgs) +Button1_Click(in sender:object, in e: EventArgs) +DestytojasMaxMod(Studentai:string(), Destytojai:string(), out destytojasMaxVardas string, out destytojasMaxPavarde : string) +ArVisuPasirinko(Studentai:string(), Destytojai:string(), out destytojasMaxVardas string, out destytojasMaxPavarde : string) +GrupėsKurNepasirinko(Studentai:string(), Destytojai:string(), out destytojasMaxVardas string, out destytojasMaxPavarde : string, fr: string) +Rikiuoti(StudentaiGR:string()) +SalintiGrupe(StudentaiGR:string(), grupe:string) +PasalintiPaskutini(StudentaiGR:string()) +SpausdintiGrupės(StudentaiGR:string(), fr:string) +SpausdintiRezultatus(Studentai:string(),Destytojai:string(),destytojasMaxVardas:string, destytojasMaxPavarde:string, fr:string) +Button2_Click(in sender:object, in e: EventArgs) +SudarytiStudentuSarasa(NewStudentai:string(), modulisVIP:string, Studentai:string()) +SpausdintiRez2(NewStudentai:string(), modulisVIP:string, fr:string)</pre>
Skaityti
<pre>+ SkaitytiStud(){query} + SkaitytiDest(){query}</pre>

2.7. Programos vartotojo vadovas

Tekstiniame faile duota informacija apie studentų pasirenkamus modulius. Kitame faile duota informacija apie modulius. Paleidę programą spaudžiame pirmąją „Start“ mygtuką, tuomet ekrane parodoma informacija kuris dėstytojas turi daugiausia pasirinktų modulių, ar visų grupių studentai pasirinko to dėstytojo modulius. Taip pat, atspausdintą lentelę su grupėmis, kurių studentai nepasirinko šio dėstytojo modulių. Toliau spaudžiame antrąją „Start“ mygtuką. Atsiranda užrašas, prašantis įvesti modulio pavadinimą ir dar kartą paspausti antrąją „Start“ mygtuką. Tuomet atsiranda jūsų įvesto modulio studentų sąrašas.

3. Bendrinis susietasis sąrašas

3.1. Darbo užduotis

LD_7. Moduliai.

Studentai renka modulius. Už modulius yra atsakingi dėstytojai. Dėstytojas gali būti atsakingas už keletą modulių.

Suraskite, kuris dėstytojas turi daugiausiai pasirinktų modulių.

Nustatykite, ar visų grupių studentai pasirinko šio dėstytojo modulius.

Atspausdinkite grupes, kurių studentai nepasirinko šio dėstytojo modulių.

Duomenys:

- Tekstiniame faile U7a.txt duota informacija apie studentų pasirenkamus modulius: modulio pavadinimas, studento pavardė, vardas, grupė.
- Tekstiniame faile U7b.txt duota informacija apie modulius: modulio pavadinimas, atsakingo dėstytojo pavardė, vardas, kreditų kiekis.

Spausdinamas sąrašas turi būti surikiuotas abėcėlės tvarka.

Sudarykite nurodyto modulio (įvedamas klaviatūra) pasirinkusių studentų sąrašą.

3.2. Programos tekstas

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class L3 : System.Web.UI.Page
{
    public class Studentas : IComparable<Studentas>, IEquatable<Studentas>
    {
        public string modulis { get; set; }
        public string pavarde { get; set; }
        public string vardas { get; set; }
        public string grupe { get; set; }

        //public Studentas() { }

        public Studentas(string modulis, string pavarde, string vardas, string
grupe)
        {
            this.modulis = modulis;
            this.pavarde = pavarde;
            this.vardas = vardas;
            this.grupe = grupe;
        }
        public int CompareTo(Studentas other)
        {
            if (other == null) return 1;
            if (modulis.CompareTo(other.modulis) != 0)
                return modulis.CompareTo(other.modulis);
            else
                return pavarde.CompareTo(other.pavarde);
        }
        static public bool operator >(Studentas pir, Studentas ant)
        {
```

```

        return pir.CompareTo(ant) == 1;
    }
    static public bool operator <(Studentas pir, Studentas ant)
    {
        return pir.CompareTo(ant) == -1;
    }
    public override string ToString()
    {
        string eilute;
        eilute = string.Format("{0,-15} {1,-20} {2, -15} {3,-10}", modulis,
pavarde, vardas, grupe);
        return eilute;
    }
    public bool Equals(Studentas other)
    {
        throw new NotImplementedException();
    }
}

public class Destytojas : IComparable<Destytojas>, IEquatable<Destytojas>
{
    public string modulis { get; set; }
    public string pavarde { get; set; }
    public string vardas { get; set; }
    public int kreditai { get; set; }
    public Destytojas(string modulis, string pavarde, string vardas, int
kreditai)
    {
        this.modulis = modulis;
        this.pavarde = pavarde;
        this.vardas = vardas;
        this.kreditai = kreditai;
    }
    public string ToString1()
    {
        string eilute;
        eilute = string.Format("{0,-15} {1,-15} {2, -15} {3,-10}", modulis,
pavarde, vardas, kreditai);
        return eilute;
    }
    public int CompareTo(Destytojas other)
    {
        if (other == null) return 1;
        if (modulis.CompareTo(other.modulis) != 0)
            return modulis.CompareTo(other.modulis);
        else
            return pavarde.CompareTo(other.pavarde);
    }
    static public bool operator >(Destytojas pir, Destytojas ant)
    {
        return pir.CompareTo(ant) == 1;
    }
    static public bool operator <(Destytojas pir, Destytojas ant)
    {
        return pir.CompareTo(ant) == -1;
    }
    public bool Equals(Destytojas other)
    {
        throw new NotImplementedException();
    }
}

public sealed class Mazgas<t> where t : IComparable<t>, IEquatable<t>
{
    public t Duom { get; set; }
    public Mazgas<t> Kitas { get; set; }
    public Mazgas(t st, Mazgas<t> adr)

```

```

        {
            Duom = st;
            Kitas = adr;
        }
    }

    public sealed class Sarasas<t> : IEnumerable where t : IComparable<t>,
IEquatable<t>
    {
        private Mazgas<t> pr; //saraso pradzia
        private Mazgas<t> pb; //saraso pabaiga
        private Mazgas<t> d; //saraso sasajai
        public Sarasas()
        {
            this.pr = null;
            this.pb = null;
            this.d = null;
        }
        public Mazgas<t> Pradzia()
        {
            return pr;
        }
        public void Kitas()
        {
            d = d.Kitas;
        }
        public bool Yra()
        {
            return d != null;
        }
        public t ImtiDuomenis()
        {
            return d.Duom;
        }
        public void DetiDuomenisT(t naujas)
        {
            var dd = new Mazgas<t>(naujas, null);
            if (pr != null)
            {
                pb.Kitas = dd;
                pb = dd;
            }
            else
            {
                pr = dd;
                pb = dd;
            }
        }
        public IEnumerator GetEnumerator()
        {
            for (Mazgas<t> dd = pr; dd != null; dd = dd.Kitas)
            {
                yield return dd.Duom;
            }
        }
        //būtina aprašyti, nes IEnumerable<T> paveldi iš IEnumerable
        IEnumerator IEnumerable.GetEnumerator()
        {
            throw new NotImplementedException();
        }
        public void Rikiuoti()
        {
            for (Mazgas<t> d1 = pr; d1 != null; d1 = d1.Kitas)
            {
                Mazgas<t> max = d1;
                for (Mazgas<t> d2 = d1; d2 != null; d2 = d2.Kitas)
                    if (d2.Duom.CompareTo(max.Duom) < 0)

```

```

        max = d2;
        t laik = d1.Duom;
        d1.Duom = max.Duom;
        max.Duom = laik;
    }
}

protected void Page_Load(object sender, EventArgs e)
{ }

protected void Button1_Click(object sender, EventArgs e)
{
    var Studentai = new Sarasas<Studentas>();
    var Destytojai = new Sarasas<Destytojas>();
    string CD1 = @"E:\\2tras\\objektinis\\L2\\U7a.txt"; //studentu duomenu
failas
    string CD2 = @"E:\\2tras\\objektinis\\L2\\U7b.txt"; //destytoju duomenu
failas
    string CR = @"E:\\2tras\\objektinis\\L2\\REZ.txt"; //rezultatu duomenu
failas
    if (File.Exists(CR))
        File.Delete(CR);
    SkaitytiStud(CD1, Studentai);
    SkaitytiDest(CD2, Destytojai);
    string destytojasMaxVardas; string destytojasMaxPavarde; //destytojas
turintis daugiausiai moduliu
    DestytojasMaxMod(Studentai, Destytojai, out destytojasMaxVardas, out
destytojasMaxPavarde);
    SpausdintiRezultatus(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde, CR);
    GrupesKurNepasirinko(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde, CR);
}
/// <summary>
/// Nuskaito studentu duomenu faila
/// </summary>
/// <param name="fv">studentu failas</param>
/// <param name="Studentai">Studentu sarasas</param>
static void SkaitytiStud(string fv, Sarasas<Studentas> Studentai)
{
    string line;
    using (var reader = new StreamReader(fv))
    {
        while ((line = reader.ReadLine()) != null)
        {
            var v = line.Split(' ');
            var student = new Studentas(v[0], v[1], v[2], v[3]);
            Studentai.DetiDuomenisT(student);
        }
    }
}
/// <summary>
/// Nuskaito destytoju duomenu faila
/// </summary>
/// <param name="fv">destytoju failas</param>
/// <param name="Destytojai">Destytoju sarasas</param>
static void SkaitytiDest(string fv, Sarasas<Destytojas> Destytojai)
{
    string line;
    using (var reader = new StreamReader(fv))
    {
        while ((line = reader.ReadLine()) != null)
        {
            var v = line.Split(' ');
            var destyt = new Destytojas(v[0], v[1], v[2], int.Parse(v[3]));

```



```

        Destytojai.DetiDuomenisT(destyt);
    }
}

/// <summary>
/// suranda destytoja, turinti daugiausiai modulių
/// </summary>
/// <param name="Studentai">Studentų sąrašas</param>
/// <param name="Destytojai">Destytojų sąrašas</param>
/// <param name="destytojasMaxVardas">daugiausiai modulių turincio destytojo
vardas</param>
/// <param name="destytojasMaxPavarde">daugiausiai modulių turincio destytojo
pavarde</param>
static void DestytojasMaxMod(Sarasas<Studentas> Studentai, Sarasas<Destytojas>
Destytojai, out string destytojasMaxVardas, out string destytojasMaxPavarde)
{
    int max = 0;
    destytojasMaxVardas = null;
    destytojasMaxPavarde = null;
    foreach (Destytojas dest in Destytojai)
    {
        int laik = 0;
        foreach (Studentas stud in Studentai)
        {
            if (dest.modulis == stud.modulis)
                laik += 1;
        }
        if (max < laik)
        {
            destytojasMaxVardas = dest.vardas;
            destytojasMaxPavarde = dest.pavarde;
            max = laik;
        }
    }
}

/// <summary>
/// Suranda ar visu grupių studentai pasirinko šio destytojo modulius
/// </summary>
/// <param name="Studentai">Studentų sąrašas</param>
/// <param name="Destytojai">Destytojų sąrašas</param>
/// <param name="destytojasMaxVardas">daugiausiai modulių turincio destytojo
vardas</param>
/// <param name="destytojasMaxPavarde">daugiausiai modulių turincio destytojo
pavarde</param>
/// <returns></returns>
static bool ArVisuPasirinko(Sarasas<Studentas> Studentai, Sarasas<Destytojas>
Destytojai, string destytojasMaxVardas, string destytojasMaxPavarde)
{
    foreach (Destytojas dest in Destytojai)
    {
        if ((destytojasMaxVardas == dest.vardas) && (destytojasMaxPavarde ==
dest.pavarde))
        {
            string modulis = dest.modulis;
            foreach (Studentas stud in Studentai)
            {
                if (modulis == stud.modulis)
                {
                    int visi = 0; int ne = 0;
                    foreach (Studentas stud1 in Studentai)
                    {
                        if (stud1.grupe == stud.grupe)
                        {
                            visi += 1;
                            if (stud.modulis != modulis)
                                ne += 1;
                        }
                    }
                }
            }
        }
    }
}

```

```

        }
        }
        if (ne == visi)
            return false;
    }
}

return true;
}

public void SalintiGrupe(Sarasas<Studentas> StudentaiGR, string grupe)
{
    for (Mazgas<Studentas> d = StudentaiGR.Pradzia(); d != null; d = d.Kitas)
    {
        if (d.Duom.grupe == grupe)
        {
            if (d.Kitas != null)
            {
                Mazgas<Studentas> laik = d.Kitas;
                d.Duom = laik.Duom;
                d.Kitas = laik.Kitas;
            }
            else
                PasalintiPaskutini(StudentaiGR);
        }
    }
}

/// <summary>
/// suranda grupes, kuriu bent vienas studentas nepasirinko sio destytojo
modulio
/// </summary>
/// <param name="Studentai">Studentu sarasas</param>
/// <param name="Destytojai">Destytoju sarasas</param>
/// <param name="destytojasMaxVardas">daugiausiai moduliu turincio destytojo
vardas</param>
/// <param name="destytojasMaxPavarde">daugiausiai moduliu turincio destytojo
pavarde</param>
/// <param name="fr">rezultatu failas</param>
public void GrupesKurNepasirinko(Sarasas<Studentas> Studentai,
Sarasas<Destytojas> Destytojai, string destytojasMaxVardas, string
destytojasMaxPavarde, string fr)
{
    var StudentaiGR = new Sarasas<Studentas>();
    StudentaiGR = Studentai;
    //rusiuoja pagal grupes, pavardes ir vardus

    //jei grupes yra nors vienas nepasirinkes modulio, atspauszinti
    //jei visa grupe pasirinkusi ta moduli, pasalinti grupe
    foreach (Destytojas dest in Destytojai)
    {
        if (destytojasMaxVardas == dest.vardas && destytojasMaxPavarde ==
dest.pavarde)
        {
            string modulis = dest.modulis;
            foreach (Studentas stud in Studentai)
            {
                if (modulis == stud.modulis)
                {
                    int visi = 0; int ne = 0; string grupe = "";
                    foreach (Studentas stud1 in Studentai)
                    {
                        if (stud.grupe == stud1.grupe)
                        {
                            visi += 1;
                            if (stud1.modulis == modulis)

```

```

        {
            ne += 1;
            grupe = stud1.grupe;
        }
    }
}
if (ne == visi)
{
    SalintiGrupe(StudentaiGR, grupe);
}
}
}
}
StudentaiGR.Rikiuoti();
SpausdintiGrupės(StudentaiGR, fr);
}
/// <summary>
/// pasalina paskutini elementa
/// </summary>
/// <param name="StudentaiGR">surikiuotu studentu sarasas</param>
static void PasalintiPaskutini(Sarasas<Studentas> StudentaiGR)
{
    Mazgas<Studentas> k = StudentaiGR.Pradzia();
    for (Mazgas<Studentas> d = StudentaiGR.Pradzia(); d != null; d = d.Kitas)
    {
        k = d.Kitas;
        if (k.Kitas == null)
        {
            d.Kitas = null;
            StudentaiGR.DetiDuomenisT(d.Duom);
        }
    }
}
/// <summary>
/// spausdinti rezultatus
/// </summary>
/// <param name="StudentaiGR">surikiuotu studentu sarasas</param>
/// <param name="fr">rezultatu failas</param>
public void SpausdintiGrupės(Sarasas<Studentas> StudentaiGR, string fr)
{
    string line = "";
    using (var writer = File.AppendText(fr))
    {
        writer.WriteLine();
        writer.WriteLine("Grupės, kuriu studentai nepasirinko sio destytojo
moduliu:");
        Label5.Text = "Grupės, kuriu studentai nepasirinko sio destytojo
moduliu:";
        line += "Modulis          Pavarde          Vardas          Grupe" +
"\r\n" + "\r\n";
        for (Mazgas<Studentas> d = StudentaiGR.Pradzia(); d != null; d =
d.Kitas)
        {
            writer.WriteLine(d.Duom.ToString());
            line += d.Duom.ToString() + "\r\n";
        }
        TextBox3.Text = line;
    }
}
protected void Button2_Click(object sender, EventArgs e)
{
    string CD1 = @"E:\\2tras\\objektinis\\L2\\U7a.txt"; //studentu duomenu
failas

```

```

        string CR = @"E:\\2tras\\objektinis\\L2\\REZ.txt"; //rezultatu duomenu
failas
        var Studentai = new Sarasas<Studentas>();
        SkaitytiStud(CD1, Studentai);
        Label4.Text = "Iveskite modulio pavadinima ir dar karta paspauskite
antraji 'START' mygtuka";
        string modulisVIP = TextBox1.Text;
        var NewStudentai = new Sarasas<Studentas>();
        SudarytiStudentuSarasa(NewStudentai, modulisVIP, Studentai);
        if (modulisVIP != "")
            SpausdintiRez2(NewStudentai, modulisVIP, CR);
    }
    /// <summary>
    /// sudaromas studentu sarasas pagal ivesta klaviatura moduli
    /// </summary>
    /// <param name="NewStudentai">naujas studentu sarasas</param>
    /// <param name="modulisVIP">Ivestas modulis</param>
    /// <param name="Studentai">studentu sarasas</param>
    public void SudarytiStudentuSarasa(Sarasas<Studentas> NewStudentai, string
modulisVIP, Sarasas<Studentas> Studentai)
    {
        string eil = "";
        if (modulisVIP != "")
        {
            for (Mazgas<Studentas> d = Studentai.Pradzia(); d != null; d =
d.Kitas)
            {
                if (d.Duom.modulis == modulisVIP)
                {
                    NewStudentai.DetiDuomenisT(d.Duom);
                    eil += d.Duom + "\r\n";
                }
            }
        }
        else
            eil = "Neteisingas modulio pavadinimas";
        TextBox2.Text = eil;
    }
    /// <summary>
    /// spausdina gautus rezultatus
    /// </summary>
    /// <param name="Studentai">studentu sarasas</param>
    /// <param name="Destytojai">destytoju sarasas</param>
    /// <param name="destytojasMaxVardas">daugiausiai moduliui turincio destytojo
vardas</param>
    /// <param name="destytojasMaxPavarde">daugiausiai moduliui turincio destytojo
pavarde</param>
    /// <param name="fr">rezultatu failas</param>
    public void SpausdintiRezultatus(Sarasas<Studentas> Studentai,
Sarasas<Destytojas> Destytojai, string destytojasMaxVardas, string
destytojasMaxPavarde, string fr)
    {
        using (var writer = File.AppendText(fr))
        {
            ///--- pradiniu doumenu spausdinimas
            writer.WriteLine("-----PRADINIAI-DUOMENYS-----");
        }
        writer.WriteLine();
        writer.WriteLine("-----Studentu-duomenu-failas-----");
        for (Mazgas<Studentas> d = Studentai.Pradzia(); d != null; d =
d.Kitas)
        {
            writer.WriteLine(d.Duom.ToString());
        }
        writer.WriteLine();
        writer.WriteLine("-----Destytoju-duomenu-failas-----");
    }

```

```

        for (Mazgas<Destytojas> d = Destytojai.Pradzia(); d != null; d =
d.Kitas)
        {
            writer.WriteLine(d.Duom.ToString1());
        }
        writer.WriteLine("-----");
        writer.WriteLine();
        //-----
        writer.WriteLine();
        writer.WriteLine("Daugiausiai pasirinktu moduliu turi destytojas - {0}
{1}", destytojasMaxVardas, destytojasMaxPavarde);
        Label1.Text = "Daugiausiai pasirinktu moduliu turi destytojas - " +
destytojasMaxVardas + " " + destytojasMaxPavarde;
        writer.WriteLine();
        if (ArVisuPasirinko(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde))
        {
            writer.WriteLine("Ne visu grupiu studentai pasirinko sio destytojo
modulius.");
            Label2.Text = "Visu grupiu studentai pasirinko sio destytojo
modulius.";
        }
        if (!ArVisuPasirinko(Studentai, Destytojai, destytojasMaxVardas,
destytojasMaxPavarde))
        {
            writer.WriteLine("Ne visu grupiu studentai pasirinko sio destytojo
modulius.");
            Label2.Text = "Ne visu grupiu studentai pasirinko sio destytojo
modulius.";
        }
    }
}
/// <summary>
/// spausdinami rezultatai po ivedamo modulio klaviatura
/// </summary>
/// <param name="NewStudentai">naujas studentu sarasas</param>
/// <param name="modulisVIP">Ivestas modulis</param>
/// <param name="fr">rezultatu failas</param>
public void SpausdintiRez2(Sarasas<Studentas> NewStudentai, string modulisVIP,
string fr)
{
    using (var writer = File.AppendText(fr))
    {
        writer.WriteLine();
        writer.WriteLine("-----");
        writer.WriteLine("Ivestas modulio pavadinimas: {0}", modulisVIP);
        writer.WriteLine("Sio modulio studentu sarasas:");
        for (Mazgas<Studentas> d = NewStudentai.Pradzia(); d != null; d =
d.Kitas)
        {
            writer.WriteLine(d.Duom.ToString());
        }
    }
}
}

```

```

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="L3.aspx.cs" Inherits="L3"
%>

```

```

<!DOCTYPE html>

```

```

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>

```

```

<body style="background-color:#A4A4A4">
  <form id="form1" runat="server">
    <div>

      </div>
      <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="Button" BackColor="Blue" BorderColor="White"/>
      <br />
      <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
      <br />
      <asp:Label ID="Label2" runat="server" Text="Label"></asp:Label>
      <br />
      <br />
      <asp:Label ID="Label5" runat="server" Text="Label"></asp:Label>
      <br />
      <asp:TextBox ID="TextBox3" runat="server" Height="254px"
BordeColor="#FFCCFF" TextMode="MultiLine" Width="653px"></asp:TextBox>
      <br />
      <br />
      <asp:Button ID="Button2" runat="server" OnClick="Button2_Click"
Text="Button" BackColor="Blue" BorderColor="White"/>
      <br />
      <asp:Label ID="Label4" runat="server" Text="Label"></asp:Label>
      <br />
      <br />
      <asp:TextBox ID="TextBox1" runat="server" Height="18px"
BordeColor="#58D3F7" TextMode="MultiLine" Width="222px"></asp:TextBox>
      <br />
      <br />
      <asp:TextBox ID="TextBox2" runat="server" Height="175px"
BordeColor="#58D3F7" TextMode="MultiLine" Width="658px"></asp:TextBox>
      <p>
        &nbsp;   </p>
      <p>
        &nbsp;   </p>
      <p>
        &nbsp;   </p>
      <p>
        &nbsp;   </p>
      <br />
      <br />
    </form>
  </body>
</html>

```

3.3. Pradiniai duomenys ir rezultatai

Pradiniai duomenys

Failas: U7a.txt

```

Programavimas Tomaitis Tomas IFF-5/4
Matematika Pjuklaite Ieva IFF-5/2
Psichologija Puodzius Jonas IFF-5/3
Programavimas Lygiauskas Vladimiras IFF-5/1
Programavimas Aiseviciute Aiste IFF-5/4
Medijos Pukutis Pukis IFF-5/5
Programavimas Stasys Girnaitis IFF-5/4
Programavimas Gaile Pagalvyte IFF-5/2
Medijos Ignas Lovauskas IFF-5/2
Programavimas Diana Paveikslaitė IFF-5/1
Matematika Dainius Lentauskas IFF-5/2
Programavimas Vytenis Deziauskas IFF-5/3
Psichologija Raigardas Knygius IFF-5/1

```

Programavimas Giedrius Palangiauskas IFF-5/4
 Psichologija Audrius Dalgiauskas IFF-5/3
 Programavimas Jomante Deklaityte IFF-5/4
 Matematika Antanas Stiklius IFF-5/3

Failas: U7b.txt

Programavimas Jonas Jonaitis 6
 Medijos Arnas Sofauskas 5
 Informacines Radvile Stalciukaite 7
 Matematika Gintaras Grebliauskas 8
 Skaitmenine Arnas Sofauskas 4
 Psichologija Radvile Stalciukaite 7

Rezultatai

Failas: REZ.txt

-----PRADINIAI-DUOMENYS-----

-----Studentu-duomenu-failas-----

Programavimas	Tomaitis	Tomas	IFF-5/4
Matematika	Pjuklaite	Ieva	IFF-5/2
Psichologija	Puodzius	Jonas	IFF-5/3
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Programavimas	Aiseviciute	Aiste	IFF-5/4
Medijos	Pukutis	Pukis	IFF-5/5
Programavimas	Stasys	Girnaitis	IFF-5/4
Programavimas	Gaile	Pagalvyte	IFF-5/2
Medijos	Ignas	Lovauskas	IFF-5/2
Programavimas	Diana	Paveikslaitė	IFF-5/1
Matematika	Dainius	Lentauskas	IFF-5/2
Programavimas	Vytenis	Deziauskas	IFF-5/3
Psichologija	Raigardas	Knygius	IFF-5/1
Programavimas	Giedrius	Palangiauskas	IFF-5/4
Psichologija	Audrius	Dalgiauskas	IFF-5/3
Programavimas	Jomante	Deklaityte	IFF-5/4
Matematika	Antanas	Stiklius	IFF-5/3

-----Destytoju-duomenu-failas-----

Programavimas	Jonas	Jonaitis	6
Medijos	Arnas	Sofauskas	5
Informacines	Radvile	Stalciukaite	7
Matematika	Gintaras	Grebliauskas	8
Skaitmenine	Arnas	Sofauskas	4
Psichologija	Radvile	Stalciukaite	7

Daugiausiai pasirinktu modulių turi dėstytojas - Jonaitis Jonas

Nevisu grupių studentai pasirinko šio dėstytojo modulius.

Grupės, kurių studentai nepasirinko šio dėstytojo modulių:

Psichologija	Raigardas	Knygius	IFF-5/1
Programavimas	Diana	Paveikslaitė	IFF-5/1
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Matematika	Dainius	Lentauskas	IFF-5/2
Programavimas	Gaile	Pagalvyte	IFF-5/2
Medijos	Ignas	Lovauskas	IFF-5/2
Matematika	Pjuklaite	Ieva	IFF-5/2

Matematika	Antanas	Stiklius	IFF-5/3
Psichologija	Audrius	Dalgiauskas	IFF-5/3
Programavimas	Vytenis	Deziauskas	IFF-5/3
Psichologija	Puodzius	Jonas	IFF-5/3
Medijos	Pukutis	Pukis	IFF-5/5

Ivestas modulio pavadinimas: Programavimas

Sio modulio studentu sarasas:

Programavimas	Tomaitis	Tomas	IFF-5/4
Programavimas	Lygiauskas	Vladimiras	IFF-5/1
Programavimas	Aiseviciute	Aiste	IFF-5/4
Programavimas	Stasys	Girnaitis	IFF-5/4
Programavimas	Gaile	Pagalvyte	IFF-5/2
Programavimas	Diana	Paveikslaitė	IFF-5/1
Programavimas	Vytenis	Deziauskas	IFF-5/3
Programavimas	Giedrius	Palangiauskas	IFF-5/4
Programavimas	Jomante	Deklaityte	IFF-5/4

3.4. Grafinės vartotojo sąsajos schema

3.5. Grafinės vartotojo sąsajos elementų pakeistos savybės

Komponentas	ID	text	Back Color	OnClick	TextMode
Button	Button1	START	Blue	Button1_Click	
Label	Label1				
Label	Label2				
Label	Label5				
TextBox	TextBox3		#58D3F7		MultiLine
Button	Button2	START	Blue	Button2_Click	
Label	Label4				
TextBox	TextBox1		#58D3F7		MultiLine
TextBox	TextBox2		#58D3F7		MultiLine

3.6. Klasių diagramos

3.6. Klasių diagramos

Veiksmai
<pre> +Page_Load(in sender:object, in e:EventArgs) +Button1_Click(in sender:object, in e: EventArgs) +DestytojasMaxMod(Studentai:string(), Destytojai:string(), out destytojasMaxVardas string, out destytojasMaxPavarde : string) +ArVisuPasirinko(Studentai:string(), Destytojai:string(), out destytojasMaxVardas string, out destytojasMaxPavarde : string) +GrupėsKurNepasirinko(Studentai:string(), Destytojai:string(), out destytojasMaxVardas string, out destytojasMaxPavarde : string, fr: string) +SalintiGrupe(StudentaiGR:string(), grupe:string) +PasalintiPaskutini(StudentaiGR:string()) +SpausdintiGrupės(StudentaiGR:string(), fr:string) +SpausdintiRezultatus(Studentai:string(),Destytojai:string(),destytojasMaxVardas:string, destytojasMaxPavarde:string, fr:string) +Button2 Click(in sender:object, in e: EventArgs) </pre>


```
+SudarytiStudentuSarasa(NewStudenai:string(), modulisVIP:string, Studentai:string())
+SpausdintiRez2(NewStudentai:string(), modulisVIP:string, fr:string)
```

Skaityti

```
+ SkaitytiStud(){query}
+ SkaitytiDest(){query}
```

3.7. Programos vartotojo vadovas

Tekstiniame faile duota informacija apie studentų pasirenkamus modulius. Kitame faile duota informacija apie modulius. Paleidę programą spaudžiame pirmąją „Start“ mygtuką, tuomet ekrane parodoma informacija kuris dėstytojas turi daugiausia pasirinktų modulių, ar visų grupių studentai pasirinko to dėstytojo modulius. Taip pat, atspausdintą lentelę su grupėmis, kurių studentai nepasirinko šio dėstytojo modulių. Toliau spaudžiame antrąją „Start“ mygtuką. Atsiranda užrašas, prašantis įvesti modulio pavadinimą ir dar kartą paspausti antrąją „Start“ mygtuką. Tuomet atsiranda jūsų įvesto modulio studentų sąrašas.

4. Bendrinės kolekcijos

4.1. Darbo užduotės

4.2. Programos tekstas

4.3. Pradiniai duomenys ir rezultatai

4.4. Grafinės vartotojo sąsajos schema

4.5. Grafinės vartotojo sąsajos elementų pakeistos savybės

4.6. Klasų diagramos

4.7. Programos vartotojo vadovas

5. Deklaratyvusis programavimas

5.1. Darbo užduotis

5.2. Programos tekstas

5.3. Pradiniai duomenys ir rezultatai

5.4. Grafinės vartotojo sąsajos schema

5.5. Grafinės vartotojo sąsajos elementų pakeistos savybės

5.6. Klasių diagramos

5.7. Programos vartotojo vadovas