KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Objektinis programavimas II (P175B123)

Darbų aplankas

Atliko:

IFF-8/11 gr. studentas

Arnas Švenčionis

2019 m. vasario 20 d.

Priėmė:

Doc. Romas Marcinkevičius

KAUNAS 2019

TURINYS

Contents

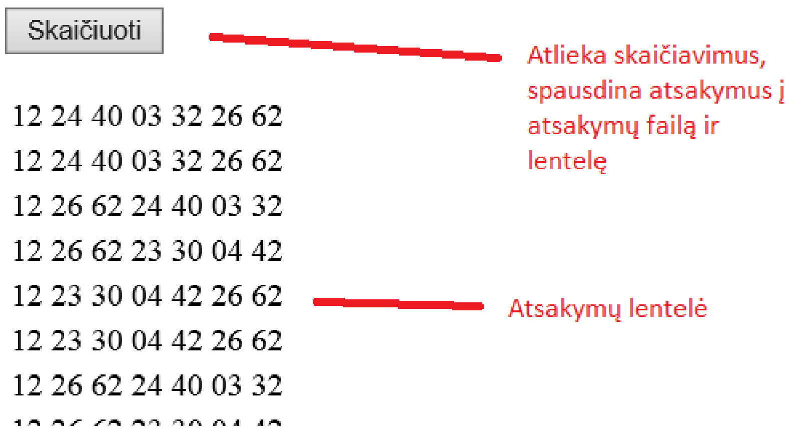
1. Rekursija (L1)

1.1. Darbo užduotis

LD_3. Domino.

Imami 7 vieno domino rinkinio kauliukai. Vieną domino kauliuką sudaro dvi dalys, kurių kiekvienoje arba nieko nėra (baltas), arba juodi taškai, kurių yra nuo 1 iki 6. Kauliuką galima nusakyti kaip dviženklį skaičių, kurio pirmas skaitmuo nurodo pirmos dalies taškų skaičių, o antrasis – antros. Jeigu dalis tuščia, tai rašomas skaitmuo 0 (nulis). Parašykite programą, kuri sudarytų iš šių 7 kauliukų visas galimas grandines, kai jungiami kauliukai galais su vienodu taškų skaičiumi. Gali būti, kad tokios grandinės visai nėra. Sudarant grandines, kauliukas gali būti apsukamas, t.y. kauliukas 35 gali būti padėtas, kaip 53. Kauliukų duomenys įvedami iš tekstinio failo'Kur3.txt'. Čia vienoje eilutėje yra parašyti 7 (septyni) dviženkliai skaičiai. Rezultatus surašyti į tekstinį failą eilutėmis po vieną grandinę. Grandinę sudaro 7 kauliukai, tarp kiekvieno kauliuko (dviženklio skaičiaus) paliekamas vieno tarpo ženklas.

1.2. Grafinės vartotojo sąsajos schema


Skaičiuoti
Atlieka skaičiavimus, spausdina atsakymus į atsakymų failą ir lentelę
Atsakymų lentelė

```
12 24 40 03 32 26 62
12 24 40 03 32 26 62
12 26 62 24 40 03 32
12 26 62 23 30 04 42
12 23 30 04 42 26 62
12 23 30 04 42 26 62
12 26 62 24 40 03 32
```

1.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Button1	Paspaudžiamas	Atlieka sprendimus, surašo atsakymus
Table1	Saugo duomenis	Rodo atsakymus

1.4. Klasių diagrama

Konteineris
- kauliukai : string[] - Count : integer

+Konteineris()
+Konteineris(in dydis : integer)
+PridetiKauliuka(in kauliukas : string)
+NustatytiKauliuka(in kauiukas : string, in ind : integer)
+GautiKauliuka{query}
+ApverstiKauliuka : string{query}
+Konteineris(in a : Konteineris, in index : integer)

Forma1.aspx.cs
max : integer
#Page_Load {query} #Button1_Click {query} -Skaityti : Konteineris() -Galimybes(in kauliukai : Konteineris, in rez : string) -PrintTable{query} -PrintToFile{query} -Tikrinimas{query}

1.5. Programos vartotojo vadovas

Paspaudus mygtuką atliekami skaičiavimai – ieškomos visi įmanomi kauliukų išdėstymo variantai.

Variantai spausdinami lentele po mygtuku. Esant kauliukų trūkumui arba pertekliui, vartotojas informuojamas.

Jei nėra įmanomų kauliukų išdėstymo variantų, vartotojui pranešama.

1.6. Programos tekstas

Konteineris.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace _1Laboras
{
    public class Konteineris
    {
        private string[] kauliukai;
        public int Count { get; private set; }

        /// <summary>
        /// Sukuria nauja kauliuku konteineri 7 dydzio
        /// </summary>
        public Konteineris ()
        {
            kauliukai = new string[7];
            Count = 0;
        }

        /// <summary>
        /// sukuria nauja konteineri, dydi leidziama nustatyti
        /// </summary>
        /// <param name="dydis">konteinerio dydis</param>
        public Konteineris (int dydis)
        {
            kauliukai = new string[dydis];
                Count = 0;
        }

        /// <summary>
        /// Prideti kauliuka i konteineri
        /// </summary>
        /// <param name="kauliukas">kauliukas</param>
        public void PridetiKauliuka(string kauliukas)
        {
            kauliukai[Count++] = kauliukas;
        }

        /// <summary>
        /// Nustatyti kauliuko duomenis
        /// </summary>
        /// <param name="kauliukas">naujas kauliukas</param>
        /// <param name="ind">naujo kauliuko vieta konteineryje</param>
        public void NustatytiKauliuka (string kauliukas, int ind)
        {
            kauliukai[ind] = kauliukas;
        }

        /// <summary>
        /// Paima kauliuka is konteinerio
        /// </summary>
        /// <param name="ind">norimo kauliuko vieta konteineryje</param>
        /// <returns>kauliuka</returns>
        public string GautiKauliuka (int ind)
        {
            return kauliukai[ind];
        }

        public string ApverstiKauliuka (int index)
        {
            return kauliukai[index][1] + kauliukai[index][0].ToString();
        }

        /// <summary>
        /// Sukuria nauja konteineri be nurodyto kauliuko
        /// </summary>
        /// <param name="a">paduotas konteineris</param>
        /// <param name="index">nenorimas kauliukas</param>
        public Konteineris(Konteineris a, int index)
        {
            int aa = 0;
                kauliukai = new string[a.Count - 1];
                for (int i = 0; i < a.Count; i++)
                    if (i != index) kauliukai[aa++] = a.GautiKauliuka(i);
            Count = a.Count-1;
        }
    }
}
```

Forma1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Forma1.aspx.cs"
Inherits="_1Laboras.Forma1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="Skaičiuoti" />
            <br />
            <br />
            <asp:Table ID="Table1" runat="server" Width="379px" style="margin-
bottom: 0px">
            </asp:Table>
        </div>
    </form>
</body>
</html>
```

Forma1.aspx.cs

```
using System;
using System.IO;
using System.Web.UI.WebControls;

namespace _1Laboras
{
    public partial class Forma1 : System.Web.UI.Page
    {
        const int max = 7;

        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            File.Delete(Server.MapPath("Ats3.txt"));
            Konteineris Kauliukai = Skaityti();
            Galimybes(Kauliukai, "");
            Tikrinimas();
        }

        /// <summary>
        /// Nuskaito duomenis is tekstinio failo
        /// </summary>
        /// <returns>sukurta duomenu konteineri</returns>
        Konteineris Skaityti()
        {
            Konteineris Kauliukai = new Konteineris();
            string line = File.ReadAllText(Server.MapPath("App_Data/Kur3.txt"));
            string[] values = line.Split(' ');
            if(values.Length != max)
            {
                PrintTable("Netinkamas kauliukų skaičius");
                PrintToFile("Netinkamas kauliukų skaičius");
                return null;
            }
            for (int i = 0; i < max; i++)
                Kauliukai.PridetiKauliuka(values[i]);
            return Kauliukai;
        }

        /// <summary>
        /// Skaiciuoja visas imanomas kauliuku sustatymo galimybes
        /// </summary>
        /// <param name="kauliukai">kauliuku rinkinys</param>
        /// <param name="rez">rezultato eilute</param>
        private void Galimybes (Konteineris kauliukai, string rez)
        {
            if (kauliukai == null)
                return;
            if (rez == "")
            {
                for (int i = 0; i < kauliukai.Count; i++)
                {
                    Konteineris naujiKauliukai = new Konteineris(kauliukai, i);
                    rez = " " + kauliukai.GautiKauliuka(i);
                    Galimybes(naujiKauliukai, rez);

                    rez = " " + kauliukai.ApverstiKauliuka(i);
                    Galimybes(naujiKauliukai, rez);
                }
            }
            else if(kauliukai.Count > 0)
            {
                for (int i = 0; i < kauliukai.Count; i++)
                {
                    if(kauliukai.GautiKauliuka(i)[0] == rez[rez.Length - 1])
                    {
                        Konteineris naujiKauliukai = new Konteineris(kauliukai,
                        i);
                        rez += " " + kauliukai.GautiKauliuka(i);
                        Galimybes(naujiKauliukai, rez);
                        rez = rez.Remove(rez.Length - 3, 3);
                    }
                    if(kauliukai.ApverstiKauliuka(i)[0] == rez[rez.Length - 1])
                    {
                        Konteineris naujiKauliukai = new Konteineris(kauliukai,
                        i);
                        rez += " " + kauliukai.ApverstiKauliuka(i);
                        Galimybes(naujiKauliukai, rez);
                        rez = rez.Remove(rez.Length - 3, 3);
                    }
                }
            }
            else
            {
                PrintTable(rez);
                PrintToFile(rez);
            }
        }

        /// <summary>
        /// Atspausdina gautus atsakymus lentele
        /// </summary>
        /// <param name="rez">vienas atsakymas</param>
        private void PrintTable (string rez)
        {
            TableCell cell = new TableCell();
            cell.Text = rez;

            TableRow row = new TableRow();
            row.Cells.Add(cell);

            Table1.Rows.Add(row);
        }

        /// <summary>
        /// Spausdina gautus atsakymus i txt faila
        /// </summary>
        /// <param name="rez">vienas atsakymas</param>
        private void PrintToFile (string rez)
        {
            using (StreamWriter writer = new
            StreamWriter(Server.MapPath(@"Ats3.txt"),
            true))
            {
                writer.WriteLine("{0}", rez);
            }
        }

        /// <summary>
        /// Skaiciavimu gale patikrina kiek gauta atsakymu
        /// Jei atsakymu negauta, pranesa
        /// </summary>
```

```
        private void Tikrinimas()
        {
            if (Table1.Rows.Count == 0)
            {
                PrintTable("Nėra galimų grandinių.");
                PrintToFile("Nėra galimų grandinių.");
            }
        }
    }
}
```

1.7. *Pradiniai duomenys ir rezultatai*

Pirmas bandymas:

Kur3.txt

```
13 01 02 24 14 12 25
```

Ats3.txt

```
 31 10 02 24 41 12 25
 31 10 02 21 14 42 25
 31 14 42 20 01 12 25
 31 14 42 21 10 02 25
 31 12 20 01 14 42 25
 31 12 24 41 10 02 25
 52 20 01 14 42 21 13
 52 20 01 12 24 41 13
 52 24 41 10 02 21 13
 52 24 41 12 20 01 13
 52 21 10 02 24 41 13
 52 21 14 42 20 01 13
```

Antras bandymas:

Kur3.txt

```
13 26 54 15 65 21 42
```

Ats3.txt

```
Nėra galimų grandinių.
```

Trečias bandymas:

Kur3.txt

```
13 26 54 15 65 21
```

Ats3.txt

```
Netinkamas kauliukų skaičius
```

Ketvirtas bandymas:

Kur3.txt

```
12 24 26 40 03 32 62
```

Ats3.txt

```
 12 24 40 03 32 26 62
 12 24 40 03 32 26 62
 12 26 62 24 40 03 32
```

```
12 26 62 23 30 04 42
12 23 30 04 42 26 62
12 23 30 04 42 26 62
12 26 62 24 40 03 32
12 26 62 23 30 04 42
24 40 03 32 26 62 21
24 40 03 32 26 62 21
26 62 24 40 03 32 21
26 62 23 30 04 42 21
23 30 04 42 26 62 21
23 30 04 42 26 62 21
26 62 24 40 03 32 21
26 62 23 30 04 42 21
```

1.8. Dėstytojo pastabos

Testo rezultatas : 1;

2. Dinaminis atminties valdymas (L2)

2.1. Darbo užduotis

LD_3. Leidiniai.

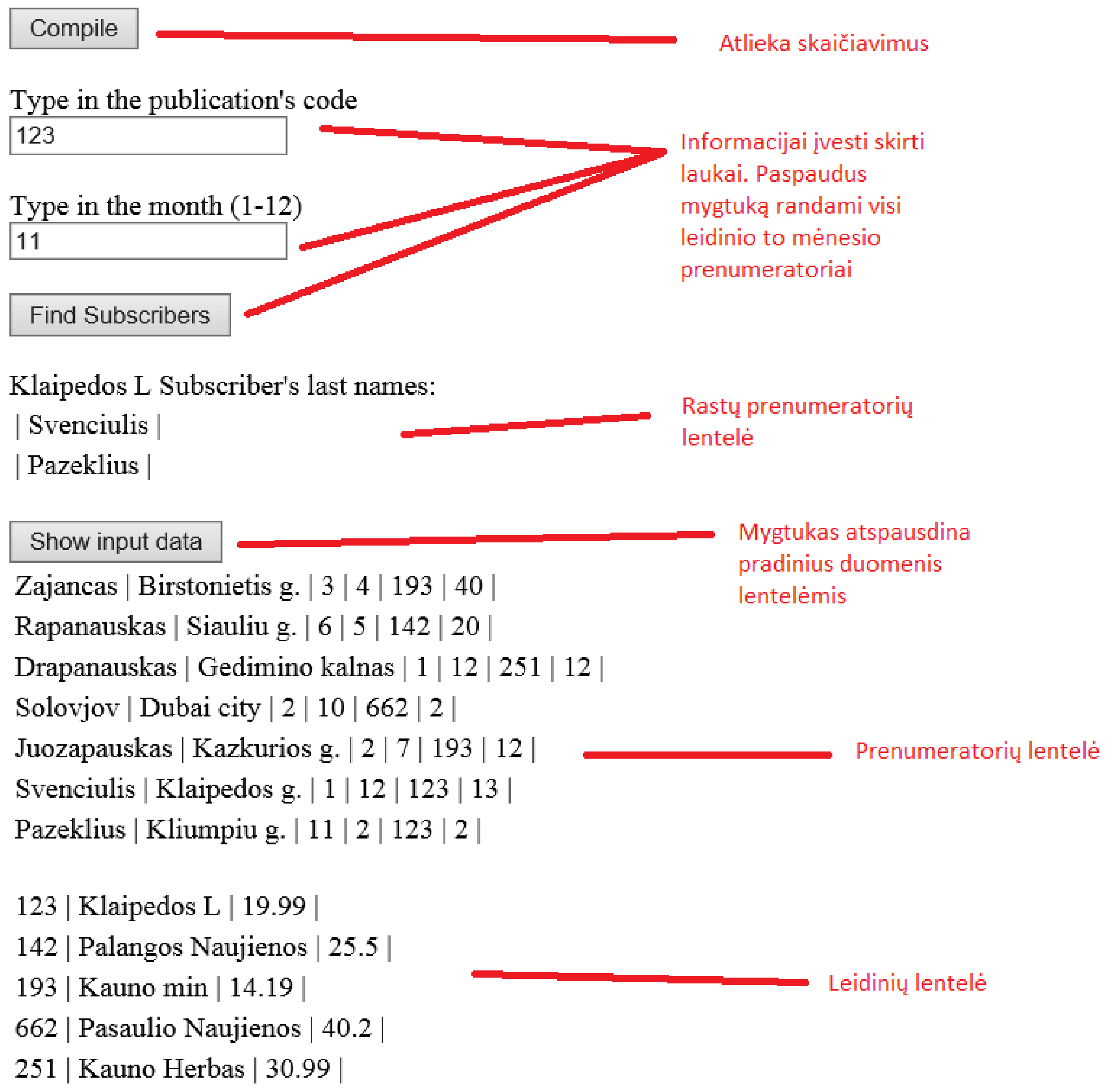
Žmonės užsisako spaudą. Užsakymas vyksta metų ribose. Leidiniai gauna dėl to pajamas. Nustatykite kiekvienam mėnesiui, kurio leidinio pajamos yra didžiausios. Nustatykite bendrąsias leidinių pajamas. Sudarykite sąrašą leidinių, kurių pajamos mažesnės už vidutines. Duomenys:

Tekstiniame faile U3a.txt yra tokia informacija apie leidinius: leidinio kodas, leidinio pavadinimas, vieno mėnesio leidinio kaina.

Tekstiniame faile U3b.txt yra informacija apie prenumeratorius: prenumeratoriaus pavardė, adresas, laikotarpio pradžia (sveikasis skaičius 1..12), laikotarpio ilgis, leidinio kodas, leidinių kiekis.

Spausdinamas sąrašas turi būti surikiuotas pagal vieno mėnesio leidinio kainą mažėjimo ir leidinio pavadinimą abėcėlės tvarka. Sudarykite nurodyto leidinio (įvedamas klaviatūra) nurodyto mėnesio ((įvedamas klaviatūra) prenumeratorių sąrašą.

2.2. Grafinės vartotojo sąsajos schema


Compile
Atlieka skaičiavimus
Informacijai įvesti skirti laukai. Paspaudus mygtuką randami visi leidinio to mėnesio prenumeratoriai
Find Subscribers
Rastų prenumeratorių lentelė
Show input data
Mygtukas atspausdina pradinius duomenis lentelėmis
Prenumeratorių lentelė
Leidinių lentelė

Type in the publication's code

123

Type in the month (1-12)

11

```
Klaipedos L Subscriber's last names:
| Svenciulis |
| Pazeklius |
```

```
Zajancas | Birstonietis g. | 3 | 4 | 193 | 40 |
Rapanauskas | Siauliu g. | 6 | 5 | 142 | 20 |
Drapanauskas | Gedimino kalnas | 1 | 12 | 251 | 12 |
Solovjov | Dubai city | 2 | 10 | 662 | 2 |
Juozapauskas | Kazkurios g. | 2 | 7 | 193 | 12 |
Svenciulis | Klaipedos g. | 1 | 12 | 123 | 13 |
Pazeklius | Kliumpiu g. | 11 | 2 | 123 | 2 |

123 | Klaipedos L | 19.99 |
142 | Palangos Naujienos | 25.5 |
193 | Kauno min | 14.19 |
662 | Pasaulio Naujienos | 40.2 |
251 | Kauno Herbas | 30.99 |
```

2.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Button1	Clickable	Atlieka skaičiavimus
Button2	Clickable	Atspausdina prenumeratorių sarašą
Button3	Clickable	Atspausdina pradinius duomenis lentele
Label1	Rodyti tekstą	Yra virš TextBox1, paaiškina ką rašyti
Label2	Rodyti tekstą	Yra virš TextBox2, paaiškina ką rašyti
Label3	Rodyti tekstą	Yra virš Table1, paaiškina kas vaizduojama
SubTLabel	Rodyti tekstą	Pradinių prenumeratorių duomenų lentelė
SubPLabel	Rodyti tekstą	Pradinių leidinių duomenų lentelė
Label4	Rodyti tekstą	Praneša vartotoją neradus duomenų
TextBox1	Įvesti tekstą	Leidinio kodo įvedimui
TextBox2	Įvesti tekstą	Mėnesio įvedimui
Table1	Rodytiduomenis lentele	Spausdinamas leidinio prenumeratorių sąrašas
RequiredFieldValidator 1	Apsauga	Tikrina ar TextBox1 yra tuščias
RequiredFieldValidator 2	Apsauga	Tikrina ar TextBox2 yra tuščias
ValidationSummary1	Praneša dėl apsaugos	Parodo jei bent vienas laukas yra tuščias
TableSubs	Rodytiduomenis lentele	Pradinių duomenų prenumeratoriu lentelė
TablePubs	Rodytiduomenis lentele	Pradinių duomenų leidinių lentelė

2.4. Klasių diagrama

Publication.cs
+Code : string +Name : string +Price: double

+Publication(in code: string, in name: string, in price: double)
+<(in l: Publication, in r: Publication): boolen
+>(in l: Publication, in r: Publication): boolen
+ToString() : string {query}
+Header() : string {query}

Publist.cs
first : Knot last : Knot current : Knot

-Knot : sealed class
+PubList()
+AddToEnd(in pub : Publication)
+First()
+Next()
+End() : boolen

+PublicationData() : Publication
+Empty() : boolen
+SetIncomeToZero()
+Sorting()

-Knot

+publication : Publication
+next : Knot

+Knot(in input : Publication, in adr : Knot)

Subscriber.cs

+LastName : string
+Adress: string
+SubscribtionStart: integer
+SubscribtionDuration: integer
+SubscribtionCode: string
+SubscribtionAmount: integer

+Subscriber(in lastname: string in adress: string, in subscribtionStart: integer, in subscribtionDuration: integer, subscribtionCode: string, in subscribtionAmount: integer)
+ToString() : string
+Header() : string

SubList.cs

first : Knot
last : Knot
current : Knot

-Knot : sealed class
+SubList()
+AddToEnd(in pub : Subscriber)
+First()
+Next()
+End() : boolen
+SubscriberData() : Subscriber
+Empty() : boolen

-Knot

+subscriber : Subscriber
+next : Knot

+Knot(in input: Subscriber, in adr: Knot)

Forma.aspx.cs

#Page_Load(in sender: object, in e : EventArgs)
#Button1_Click(in sender : object, in e : EventArgs)
-SubscriberInfo(in list: SubList) : SubList
-PublicationInfo(in list: PubList): PubList
-MostIncomeByMonth(in P: PubList, in S: SubList)
-HighestIncome(in list PubList) : Publication
- PrintBestMonthly(in pub: Publication, in month: integer)
-PublicationIncome(in Subs: Sublist, in Pubs: Publist)
-AllIncome(in Publications: Publist)
-LowIncomePublications(in All: PubList) : PubList
-Average(in All: PubList) : double
#Button2_Click(in sender: object, in e : EventArgs)
-FindPublicationWithCode(in list : PubList) : Publication
-FindSubscribers(in Subs: SubList, in Code : string, in month: integer): SubList
-PrintSubsribersToTable(in list: Sublist, in pubName: string)
-PrintData(in file: string, in list: SubList)
-PrintData(in file: string, in header: string, in list: PubList)
-PrintData(in file: string, in AllIncome: double)
-PrintSubsToTable(in subs: SubList)
-PrintPublicationToPubTable(in pubs: PubList)
-PrintInputData(in subs: SubList, in pubs: PubList, in answerFile: string)
#Button3_Click(in sender: object, in e : EventArgs)
-Checking(in Publications: PubList, in Subscribers: SubList)

2.5. Programos vartotojo vadovas

Paspaudus mygtuką Compile, programa atlieka skaičiavimus, atsakymus surašo į atsakymų faila.

Įrašius leidinio kodą ir norimą mėnesį į teksto laukus ir paspaudus mygtuką “Find Subscribers”, programa randa ir atspausdina to leidinio ir mėnesio prenumeratorių sarašą.

Paspaudus mygtuką “Show input data”, programa parodo pradinius duomenis lentelėmis.

2.6. Programos tekstas

```
public class PubList
    {
        Knot first { get; set; }
        Knot last { get; set; }
        Knot current { get; set; }
        private sealed class Knot
        {
            public Publication publication { get; set; }
            public Knot next { get; set; }

            public Knot(Publication input, Knot adr)
            {
                publication = input;
                next = adr;
            }
        }
        /// <summary>
        /// sets the first and last pointers to null
        /// </summary>
        public PubList()
        {
            first = last = null;
        }
```

```
/// <summary>
/// adds publication to the end of the list
/// </summary>
/// <param name="pub">Publication</param>
public void AddToEnd(Publication pub)
{
    Knot temp = new Knot(pub, null);
    if (first == null)
    {
        first = last = temp;
    }
    else
    {
        last.next = temp;
        last = temp;
    }
}
/// <summary>
/// sets the current pointer to the first
/// </summary>
public void First()
{
    current = first;
}
/// <summary>
/// sets the current pointer to the next publication
/// </summary>
public void Next()
{
    current = current.next;
}
/// <summary>
/// checks if the current pointer is the last one
/// </summary>
/// <returns>true or false</returns>
public bool End()
{
    return current == null;
}
/// <summary>
/// gets the current publication's data
/// </summary>
/// <returns>pblication's data</returns>
public Publication PublicationData()
{
    return current.publication;
}
/// <summary>
/// checks if the list is empty
/// </summary>
/// <returns>true or false</returns>
public bool Empty()
{
    return first == null;
}
/// <summary>
/// sets all publication's incomes to zero
/// </summary>
public void SetIncomeToZero()
{
    for (Knot i = first; i != null; i = i.next)
    {
        i.publication.Income = 0;
    }
}
/// <summary>
```

```
        /// sors the list
        /// </summary>
        public void Sorting()
        {
            bool bc = true;
            Knot d0, d1, r1;
            while (bc)
            {
                bc = false;
                d0 = d1 = r1 = first;
                while (d1 != null)
                {
                    if(d0.publication > d1.publication)
                    {
                        bc = true;
                        if(d0 == first)
                        {
                            first = first.next;
                            d0.next = d1.next;
                            d1.next = d0;
                        }
                        else
                        {
                            d0.next = d1.next;
                            d1.next = d0;
                            r1.next = d1;
                        }
                    }
                    r1 = d0;
                    d0 = d1;
                    d1 = d1.next;
                }
            }
        }
    }
public class Publication
    {
        public string Code { get; set; }
        public string Name { get; set; }
        public double Price { get; set; }

        public double Income { get; set; }
        /// <summary>
        /// creates a new publication object
        /// </summary>
        /// <param name="code"></param>
        /// <param name="name"></param>
        /// <param name="price"></param>
        public Publication(string code, string name ,double price)
        {
            Code = code;
            Name = name;
            Price = price;

            Income = 0;
        }
        /// <summary>
        /// Operator. Compares by purice and name
        /// </summary>
        /// <param name="l">one pubication</param>
        /// <param name="r">other publication</param>
        /// <returns>true or false</returns>
        static public bool operator < (Publication l, Publication r)
        {
            if (l.Price.CompareTo(r.Price) == 0)
            {
                return (l.Name.CompareTo(r.Name) < 0);
            }
            else return (l.Price.CompareTo(r.Price) < 0);
        }
        /// <summary>
        /// > operator. Compares by price and name
        /// </summary>
        /// <param name="l">one publication</param>
        /// <param name="r">other publication</param>
        /// <returns>true or false</returns>
        static public bool operator > (Publication l, Publication r)
        {
            if (l.Price.CompareTo(r.Price) == 0)
            {
                return (l.Name.CompareTo(r.Name) > 0);
            }
            else return (l.Price.CompareTo(r.Price) > 0);
        }
        /// <summary>
        /// prinst all publication's data to one formated string
        /// </summary>
        /// <returns>formated publication's information</returns>
        public override string ToString()
        {
            return String.Format("{0, 10} | {1, -20} | {2, 5} |", Code, Name, Price);
        }
        /// <summary>
        /// prints formated header
        /// </summary>
        /// <returns>formated header string</returns>
        public string Header()
        {
            return String.Format("{0, -10} | {1, -20} | {2, -5} |","Code", "Name", "Price");
        }
    }
    public class SubList
    {
        Knot first { get; set; }
        Knot last { get; set; }
        Knot current { get; set; }
        private sealed class Knot
        {
            public Subscriber subscriber { get; set; }
            public Knot next { get; set; }

            public Knot(Subscriber input, Knot adr)
            {
                subscriber = input;
                next = adr;
            }
        }
        /// <summary>
        /// sets the first and last knots to null
        /// </summary>
        public SubList()
        {
            first = last = null;
        }
        /// <summary>
        /// Adds a subscriber to the end of the list
        /// </summary>
        /// <param name="sub">subscriber data</param>
        public void AddToEnd(Subscriber sub)
        {
            Knot temp = new Knot(sub, null);
            if (first == null)
            {
                first = last = temp;
            }
            else
            {
                last.next = temp;
                last = temp;
            }
        }
        /// <summary>
        /// sets the current pointer to the first
        /// </summary>
        public void First()
        {
            current = first;
        }
        /// <summary>
        /// sets the current pointer to the next one
        /// </summary>
        public void Next()
        {
            current = current.next;
        }
        /// <summary>
        /// checks of the current pointer is the last one
        /// </summary>
        /// <returns>true or false</returns>
        public bool End()
        {
            return current == null;
        }
        /// <summary>
        /// gets the current pointer's subscriber's data
        /// </summary>
        /// <returns></returns>
        public Subscriber SubscriberData()
        {
            return current.subscriber;
        }
        /// <summary>
        /// checks if the list is empty
        /// </summary>
        /// <returns>true or false</returns>
        public bool Empty()
        {
            return first == null;
        }
    }
public class Subscriber
    {
        public string LastName { get; set; }
        public string Adress { get; set; }
        public int SubscribtionStart { get; set; }
        public int SubscribtionDuration { get; set; }
        public string SubscribtionCode { get; set; }
        public int SubscribtionAmount { get; set; }
        /// <summary>
        /// creats a new subscriber object
        /// </summary>
        /// <param name="lastname">subscriber's last name</param>
        /// <param name="adress">subscriber's adress</param>
        /// <param name="subscribtionstart">subscriber's subscribtion start</param>
        /// <param name="subscribtionduration">subscriber's subscribtion duration</param>
        /// <param name="subscribtioncode">subscriber's subscribtion code</param>
        /// <param name="subscribtionamount">subscriber's subscribtion amount</param>
        public Subscriber (string lastname, string adress, int subscribtionstart, int
              subscribtionduration, string subscribtioncode, int subscribtionamount)
        {
            LastName = lastname;
            Adress = adress;
            SubscribtionStart = subscribtionstart;
            SubscribtionDuration = subscribtionduration;
            SubscribtionCode = subscribtioncode;
            SubscribtionAmount = subscribtionamount;
        }
        /// <summary>
        /// prints subscriber's information to one formated string
        /// </summary>
        /// <returns>formated information</returns>
        public override string ToString()
        {
            return String.Format("{0, -20} | {1, -20} | {2, -20} | {3, -25} | {4, -25} | {5, -
              20} |", LastName, Adress, SubscribtionStart, SubscribtionDuration,
              SubscribtionCode, SubscribtionAmount);
        }
        /// <summary>
        /// prints the header of a table for a table of subscribers
        /// </summary>
        /// <returns>formated header</returns>
        public string Header()
        {
            return String.Format("{0, -20} | {1, -20} | {2, 20} | {3, 25} | {4, 25} | {5, 20}
|", "LastName", "Adress", "Subscribtion Start", "Subscribtion Duration", "Subscribtion Code",
"Subscribtion Amount");
        }
    }


<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Forma.aspx.cs"
Inherits="_2Lab.Forma" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="Compile" />
            <asp:Label ID="Label4" runat="server" ForeColor="Red"></asp:Label>
            <br />
            <br />
            <asp:Label ID="Label1" runat="server" Text="Type in the publication's
              code"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
              ControlToValidate="TextBox1" ErrorMessage="Code required" ForeColor="Red"
              ValidationGroup="Val1">*</asp:RequiredFieldValidator>
            <br />
            <br />
            <asp:Label ID="Label2" runat="server" Text="Type in the month (1-12)"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
              ControlToValidate="TextBox2" ErrorMessage="Month required" ForeColor="Red"
              ValidationGroup="Val1">*</asp:RequiredFieldValidator>
            <br />
            <br />
            <asp:Button ID="Button2" runat="server" OnClick="Button2_Click" Text="Find
              Subscribers" ValidationGroup="Val1" />
            <br />
            <br />
            <asp:Label ID="Label3" runat="server" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table1" runat="server" GridLines="Both">
            </asp:Table>
            <asp:ValidationSummary ID="ValidationSummary1" runat="server" ForeColor="Red"
              ValidationGroup="Val1" />
            <br />
            <asp:Button ID="Button3" runat="server" OnClick="Button3_Click" Text="Show input
              data" />
            <br />
            <asp:Label ID="SubTLabel" runat="server" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="TableSubs" runat="server" Visible="False" GridLines="Both">
            </asp:Table>
            <br />
            <asp:Label ID="PubTLabel" runat="server" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="TablePubs" runat="server" Visible="False" GridLines="Both">
            </asp:Table>
            <br />
        </div>
    </form>
</body>
</html>

public partial class Forma : System.Web.UI.Page
    {

        protected void Page_Load(object sender, EventArgs e)
        {
            SubList Subscribers = new SubList();
            PubList Publications = new PubList();

            Publications = PublicationInfo(Publications);
            Subscribers = SubscriberInfo(Subscribers);
            PrintSPublicationToPubTable(Publications);
            PrintSubsToSubTable(Subscribers);

            Checking(Publications, Subscribers);
        }
        /// <summary>
        /// Compiles the program
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        protected void Button1_Click(object sender, EventArgs e)
        {
            const string answerFile = "Answer.txt";

            File.Delete(Server.MapPath(answerFile));

            SubList Subscribers = new SubList();
            PubList Publications = new PubList();

            Publications = PublicationInfo(Publications);
            Subscribers = SubscriberInfo(Subscribers);

            MostIncomeByMonth(Publications, Subscribers);

            PublicationIncome(Subscribers, Publications);
            double AllPubIncome = AllIncome(Publications);
            PrintData(answerFile, AllPubIncome);
            PubList LowIncomePubs = LowIncomePublications(Publications);
            LowIncomePubs.Sorting();
            PrintDataToFile(answerFile, "Below average income publications:", LowIncomePubs);

            PrintInputData(Subscribers, Publications, answerFile);
        }

        #region ReadData
        /// <summary>
        /// Reads subscriber information from file
        /// and adds to the end of the list
        /// </summary>
        /// <param name="list">the list starting list</param>
        /// <returns>a list of subscribers with their information</returns>
        private SubList SubscriberInfo(SubList list)
        {
            using (StreamReader sr = new StreamReader(Server.MapPath("App_Data/U3b.txt")))
            {
                string line;
                while ((line = sr.ReadLine()) != null)
                {
                    string[] values = line.Split(';');
                    string lastName = values[0];
                    string city = values[1];
                    int beginning = int.Parse(values[2]);
                    int duration = int.Parse(values[3]);
                    string code = values[4];
                    int amount = int.Parse(values[5]);
                    Subscriber sub = new Subscriber(lastName, city, beginning, duration, code,
                     amount);
                    list.AddToEnd(sub);
                }
            }
            return list;
        }
        /// <summary>
        /// Reads publication informations from file
        /// and adds to the end of the given list
        /// </summary>
        /// <param name="list">a list that needs to be filled</param>
        /// <returns>updated list</returns>
        private PubList PublicationInfo(PubList list)
        {
            using (StreamReader sr = new StreamReader(Server.MapPath("App_Data/U3a.txt")))
            {
                string line;
                while ((line = sr.ReadLine()) != null)
                {
                    string[] values = line.Split(';');
                    string code = values[0];
                    string name = values[1];
                    double price = double.Parse(values[2]);
                    Publication publication = new Publication(code, name, price);
                    list.AddToEnd(publication);
                }
            }
            return list;
        }
        #endregion

        #region Income by month
        /// <summary>
        /// Goes month by month, publication by publication
        /// and searches for it's subscribers and calculates
        /// the month's income. makes a list of all publications
```

```
/// and their incomes each month
/// </summary>
/// <param name="P">list of publications</param>
/// <param name="S">list of subscribers</param>
private void MostIncomeByMonth(PubList P, SubList S)
{
    for (int month = 1; month <= 12; month++)
    {
        PubList monthly = new PubList();
        P.SetIncomeToZero();
        for (P.First(); !P.End(); P.Next()) //eina per leidinius
        {

            for (S.First(); !S.End(); S.Next()) //eina per prenumeratorius
                if (P.PublicationData().Code == S.SubscriberData().SubscribtionCode)
                    if ((month >= S.SubscriberData().SubscribtionStart) && (month <=
                    S.SubscriberData().SubscribtionStart +
                    S.SubscriberData().SubscribtionDuration - 1))
                        P.PublicationData().Income +=
                    S.SubscriberData().SubscribtionAmount * P.PublicationData().Price;
            monthly.AddToEnd(P.PublicationData());
        }
        Publication maxPub = HighestIncome(monthly);
        PrintBestMonthly(maxPub, month);
    }
}
/// <summary>
/// Compares the publications
/// finds the publication with the most income
/// that month
/// </summary>
/// <param name="list">list of publications</param>
/// <returns>the publication with the highest income</returns>
private Publication HighestIncome(PubList list)
{
    double max = 0;
    Publication maxPub = null;
    for (list.First(); !list.End(); list.Next())
        if (list.PublicationData().Income > max)
        {
            max = list.PublicationData().Income;
            maxPub = list.PublicationData();
        }
    return maxPub;
}
/// <summary>
/// Adds the publication with the highest income each month
/// to file
/// </summary>
/// <param name="pub"></param>
/// <param name="month"></param>
private void PrintBestMonthly(Publication pub, int month)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath("Answer.txt"), true))
    {
        if (month == 1)
            sw.WriteLine("Highest income every month:");
        if (pub == null)
            sw.WriteLine("{0, -2}. |{1, 20}|", month, "Nera");
        else
            sw.WriteLine("{0, -2}. |{1, 20}|", month, pub.Name);
    }
}

#endregion
```

```
#region All Publication Income
/// <summary>
/// goes through all publications, finds their subscribers
///  and calculates the publication's info
/// </summary>
/// <param name="Subs">list of subscribers</param>
/// <param name="Publications">list of publications</param>
private void PublicationIncome(SubList Subs, PubList Publications)
{
    Publications.SetIncomeToZero();
    for (Publications.First(); !Publications.End(); Publications.Next())
        for (Subs.First(); !Subs.End(); Subs.Next())
            if (Subs.SubscriberData().SubscribtionCode ==
             Publications.PublicationData().Code)
                Publications.PublicationData().Income +=
                    Publications.PublicationData().Price *
                       Subs.SubscriberData().SubscribtionAmount *
                          Subs.SubscriberData().SubscribtionDuration;
}
/// <summary>
/// sums up all of the publication's income
/// </summary>
/// <param name="Publications">list of publications</param>
/// <returns>the sum of all publication's income</returns>
private double AllIncome(PubList Publications)
{
    double sum = 0;
    for (Publications.First(); !Publications.End(); Publications.Next())
        sum += Publications.PublicationData().Income;
    return sum;
}
#endregion

#region Low Income Publications
/// <summary>
/// goes through all publications and adds
/// the publications with below average income to a new list
/// </summary>
/// <param name="All">list of publications</param>
/// <returns>a list of publications with below average income</returns>
private PubList LowIncomePublications(PubList All)
{
    double average = Average(All);
    PubList LowIncomePubs = new PubList();
    for (All.First(); !All.End(); All.Next())
        if (All.PublicationData().Income < average)
            LowIncomePubs.AddToEnd(All.PublicationData());
    return LowIncomePubs;
}

/// <summary>
/// finds the average income of all publications
/// </summary>
/// <param name="All">list of publications</param>
/// <returns>the average income</returns>
private double Average(PubList All)
{
    double sum = 0;
    int k = 0;
    for (All.First(); !All.End(); All.Next())
    {
        sum += All.PublicationData().Income;
        k++;
    }
    return sum / k;
}
```

```
#endregion

#region Selected Publication and month
/// <summary>
/// the button used to find the subscribers of a selected
/// publication the selected month
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
protected void Button2_Click(object sender, EventArgs e)
{
    PubList Pubs = new PubList();
    Pubs = PublicationInfo(Pubs);

    SubList Subs = new SubList();
    Subs = SubscriberInfo(Subs);

    string SelectedCode = TextBox1.Text;
    int SelectedMonth = int.Parse(TextBox2.Text);

    Publication foundPubl = FindPublicationWithCode(Pubs, SelectedCode);
    SubList PubSubs = FindSubscribers(Subs, SelectedCode, SelectedMonth);
    PrintSubscribersToTable(PubSubs, foundPubl);
}
/// <summary>
/// using the selected code, finds the publication
/// </summary>
/// <param name="list">list of publications</param>
/// <param name="code">the selected code</param>
/// <returns>the publication that was found using the code</returns>
private Publication FindPublicationWithCode(PubList list, string code)
{
    for (list.First(); !list.End(); list.Next())
        if (code == list.PublicationData().Code)
            return list.PublicationData();
    return null;
}
/// <summary>
/// finds the subscribers for the selected publication the selected month
/// </summary>
/// <param name="Subs">list o subscribers</param>
/// <param name="Code">the selected code</param>
/// <param name="month">the selected month</param>
/// <returns>a list of subscribers of the selected publication
/// the selected month</returns>
private SubList FindSubscribers(SubList Subs, string Code, int month)
{
    SubList PubSubs = new SubList();
    for (Subs.First(); !Subs.End(); Subs.Next())
        if (Subs.SubscriberData().SubscribtionCode == Code)
            if (month >= Subs.SubscriberData().SubscribtionStart && month <=
             Subs.SubscriberData().SubscribtionStart +
             Subs.SubscriberData().SubscribtionDuration - 1)
                PubSubs.AddToEnd(Subs.SubscriberData());
    return PubSubs;
}
/// <summary>
/// prints the found subscribers to a table
/// </summary>
/// <param name="list">list of subscribers</param>
/// <param name="pubName">the name of the publication</param>
private void PrintSubscribersToTable(SubList list, Publication pub)
{
    if (!list.Empty() && pub != null)
        for (list.First(); !list.End(); list.Next())
        {
            Label3.Visible = true;
            Label3.Text = pub.Name + " Subscriber's last names:";
            TableCell cell = new TableCell();
            string tempstring = String.Format("{0}", list.SubscriberData().LastName);
            cell.Text = tempstring;

            TableRow row = new TableRow();
            row.Cells.Add(cell);

            Table1.Rows.Add(row);
        }
    else
    {
        Label3.Visible = true;
        Label3.Text = "The publication does not have any subscribers the selected
             month.";
    }
}

#endregion

#region PrintResults
/// <summary>
/// prints subscribers to a file
/// </summary>
/// <param name="file">file name</param>
/// <param name="list">list of subscribers</param>
private void PrintDataToFile(string file, SubList list)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(@file), true))
    {
        if(list.Empty())
        {
            sw.WriteLine("No subscribers found");
            sw.WriteLine();
        }
        else
        {
            list.First();
            string line = new string('-', list.SubscriberData().ToString().Length);
            sw.WriteLine(list.SubscriberData().Header());
            sw.WriteLine(line);
            while (!list.End())
            {
                sw.WriteLine(list.SubscriberData().ToString());
                sw.WriteLine(line);
                list.Next();
            }
            sw.WriteLine();
        }
    }
}
/// <summary>
/// prints the given list to file
/// </summary>
/// <param name="file">file path</param>
/// <param name="header">header of the table</param>
/// <param name="list">list of publiactions</param>
private void PrintDataToFile(string file, string header, PubList list)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(@file), true))
    {
        if (list.Empty())
        {
            sw.WriteLine("No publications found");
            sw.WriteLine();
        }
        else
        {
            list.First();
            string line = new string('-', list.PublicationData().ToString().Length);

            sw.WriteLine(header);
            sw.WriteLine(line);
            for (list.First(); !list.End(); list.Next())
            {
                sw.WriteLine(list.PublicationData().ToString());
                sw.WriteLine(line);
            }
            sw.WriteLine();
        }
    }
}
/// <summary>
/// prints all income to file
/// </summary>
/// <param name="file">file path</param>
/// <param name="AllIncome">number</param>
private void PrintData(string file, double AllIncome)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(@file), true))
    {
        sw.WriteLine();
        sw.WriteLine("All Publication income: {0:F2}", AllIncome);
        sw.WriteLine();
    }
}
/// <summary>
/// prints subscriber data to table in web
/// </summary>
/// <param name="list">list of subscribers</param>
private void PrintSubsToSubTable(SubList subs)
{
    if (!subs.Empty())
    {
        SubTLabel.Text = "Input subscriber data:";
        TableRow row = new TableRow();
        TableCell[] cell = new TableCell[6];
        for (int i = 0; i < 6; i++)
            cell[i] = new TableCell();
        cell[0].Text = "Last Name";
        cell[1].Text = "Adress";
        cell[2].Text = "Subscribtion start";
        cell[3].Text = "Subscribtion duration";
        cell[4].Text = "Subscribtion code";
        cell[5].Text = "Subscribtion amount";
        row.Cells.AddRange(cell);
        TableSubs.Rows.Add(row);
        for (subs.First(); !subs.End(); subs.Next())
        {
            row = new TableRow();
            for (int i = 0; i < 6; i++)
                cell[i] = new TableCell();
            cell[0].Text = subs.SubscriberData().LastName;
            cell[1].Text = subs.SubscriberData().Adress;
            cell[2].Text = subs.SubscriberData().SubscribtionStart.ToString();
            cell[3].Text = subs.SubscriberData().SubscribtionDuration.ToString();
            cell[4].Text = subs.SubscriberData().SubscribtionCode;
            cell[5].Text = subs.SubscriberData().SubscribtionAmount.ToString();
            row.Cells.AddRange(cell);
            TableSubs.Rows.Add(row);
        }
    }
    else
    {
        SubTLabel.Text = "No Subscribers Found";
    }

}
/// <summary>
/// prints publication data to web
/// </summary>
/// <param name="list">list of publiactions</param>
private void PrintSPublicationToPubTable(PubList pubs)
{
    if (!pubs.Empty())
    {
        PubTLabel.Text = "Input publication data:";
        TableRow row = new TableRow();
        TableCell[] cell = new TableCell[3];
        for (int i = 0; i < 3; i++)
            cell[i] = new TableCell();
        cell[0].Text = "Code";
        cell[1].Text = "Name";
        cell[2].Text = "Price";
        row.Cells.AddRange(cell);
        TablePubs.Rows.Add(row);
        for (pubs.First(); !pubs.End(); pubs.Next())
        {
            row = new TableRow();
            for (int i = 0; i < 3; i++)
                cell[i] = new TableCell();
            cell[0].Text = pubs.PublicationData().Code;
            cell[1].Text = pubs.PublicationData().Name;
            cell[2].Text = pubs.PublicationData().Price.ToString();
            row.Cells.AddRange(cell);
            TablePubs.Rows.Add(row);
        }
    }
    else
    {
        PubTLabel.Text = "No Publications Found.";
    }

}
/// <summary>
/// calls the prmethods
/// </summary>
/// <param name="subs">subscriber list</param>
/// <param name="pubs">publication list</param>
/// <param name="answerFile">asnwer file path</param>
private void PrintInputData(SubList subs, PubList pubs, string answerFile)
{
    PrintDataToFile(answerFile, subs);
    pubs.First();
    PrintDataToFile(answerFile, pubs.PublicationData().Header(), pubs);
}
#endregion

#region Show Input Data
/// <summary>
/// 3rd button. Used to make the input tables visible
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
protected void Button3_Click(object sender, EventArgs e)
{
    TableSubs.Visible = TablePubs.Visible = true;
```

```
            SubTLabel.Visible = PubTLabel.Visible = true;
            if (Button1.Enabled == false)
            {
                File.Delete(Server.MapPath("Answer.txt"));

                SubList Subscribers = new SubList();
                PubList Publications = new PubList();

                Publications = PublicationInfo(Publications);
                Subscribers = SubscriberInfo(Subscribers);

                PrintDataToFile("Answer.txt", Subscribers);
                PrintDataToFile("Answer.txt", "No publications found" ,Publications);
            }

        }
        #endregion

        #region Checking
        /// <summary>
        /// checks if the starting lists are empty
        /// and if so, disables the buttons, alerts the user
        /// </summary>
        /// <param name="Publications">list of publications</param>
        /// <param name="Subscribers">list of subscribers</param>
        private void Checking(PubList Publications, SubList Subscribers)
        {
            if (Publications.Empty() || Subscribers.Empty())
            {
                Button1.Enabled = Button2.Enabled = false;
                Label4.Text = "No Publications or Subscribers found!";
            }
        }
        #endregion
    }
```

2.7. *Pradiniai duomenys ir rezultatai*

Pirmas bandymas:

U3a.txt

```
123;Klaipedos L;19.99
142;Palangos Naujienos;25.50
193;Kauno min;14.19
662;Pasaulio Naujienos;40.20
251;Kauno Herbas;30.99
```

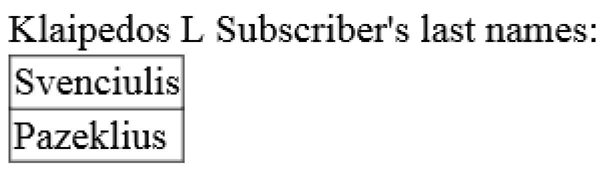
U3b.txt

```
Zajancas;Birstonietis g.;3;4;193;40
Rapanauskas;Siauliu g.;6;5;142;20
Drapanauskas;Gedimino kalnas;1;12;251;12
Solovjov;Dubai city;2;10;662;2
Juozapauskas;Kazkurios g.;2;7;193;12
Svenciulis;Klaipedos g.;1;12;123;13
Pazeklius;Kliumpiu g.;11;2;123;2
```

Answer.txt

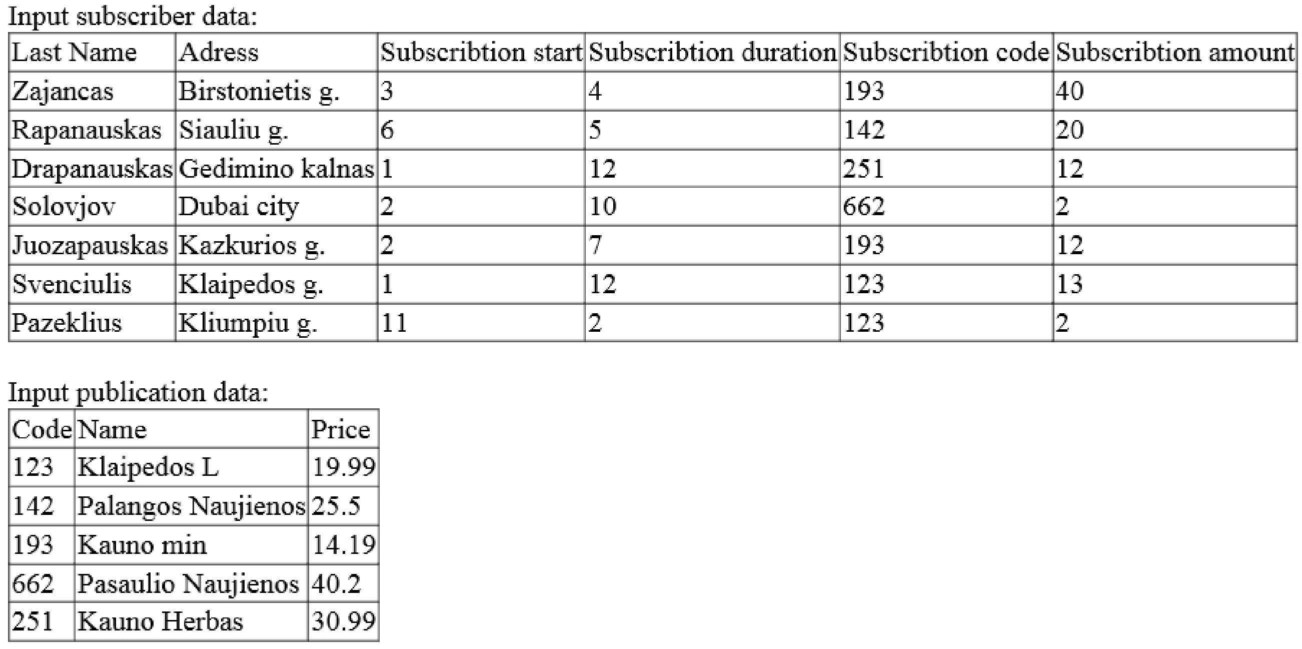
```
Highest income every month:
1 . |        Kauno Herbas|
2 . |        Kauno Herbas|
3 . |           Kauno min|
4 . |           Kauno min|
5 . |           Kauno min|
6 . |           Kauno min|
7 . |  Palangos Naujienos|
8 . |  Palangos Naujienos|
9 . |  Palangos Naujienos|
10. |  Palangos Naujienos|
11. |        Kauno Herbas|
12. |        Kauno Herbas|

All Publication income: 14477.32

Below average income publications:
-------------------------------------------
       142 | Palangos Naujienos   |  25.5 |
-------------------------------------------
       662 | Pasaulio Naujienos   |  40.2 |
-------------------------------------------

LastName            | Adress              |   Subscribtion Start |
Subscribtion Duration |         Subscribtion Code |  Subscribtion Amount |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Zajancas            | Birstonietis g.     | 3                    | 4
| 193                   | 40                    |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Rapanauskas         | Siauliu g.          | 6                    | 5
| 142                   | 20                    |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Drapanauskas        | Gedimino kalnas     | 1                    | 12
| 251                   | 12                    |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Solovjov            | Dubai city          | 2                    | 10
| 662                   | 2                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Juozapauskas        | Kazkurios g.        | 2                    | 7
| 193                   | 12                    |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Svenciulis          | Klaipedos g.        | 1                    | 12
| 123                   | 13                    |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Pazeklius           | Kliumpiu g.         | 11                   | 2
| 123                   | 2                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------

Code       | Name                 | Price |
-------------------------------------------
       123 | Klaipedos L          | 19.99 |
-------------------------------------------
       142 | Palangos Naujienos   |  25.5 |
-------------------------------------------
       193 | Kauno min            | 14.19 |
-------------------------------------------
       662 | Pasaulio Naujienos   |  40.2 |
-------------------------------------------
       251 | Kauno Herbas         | 30.99 |
-------------------------------------------
```

Į pirmą lauką įrašius 123, o į antrą 11, atspausdinama ši lentelė:


Klaipedos L Subscriber's last names:

Svenciulis
Pazeklius

```
Pradiniai duomenys vartotojo sąsajoje:
```

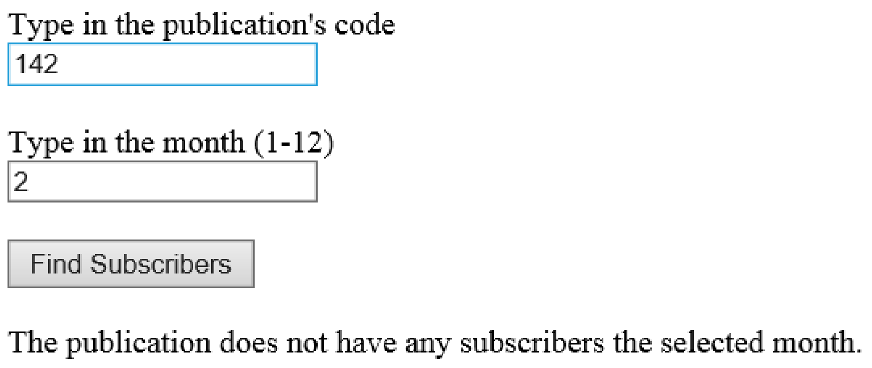

Input subscriber data:

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Zajancas	Birstonietis g.	3	4	193	40
Rapanauskas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	12	251	12
Solovjov	Dubai city	2	10	662	2
Juozapauskas	Kazkurios g.	2	7	193	12
Svenciulis	Klaipedos g.	1	12	123	13
Pazeklius	Kliumpiu g.	11	2	123	2

Input publication data:

Code	Name	Price
123	Klaipedos L	19.99
142	Palangos Naujienos	25.5
193	Kauno min	14.19
662	Pasaulio Naujienos	40.2
251	Kauno Herbas	30.99

```
Į pirmą lauką įrašius 142, o 5 antrą - 2, spausdinamas rezultatas yra:
The publication does not have any subscribers the selected month.
```


Type in the publication's code

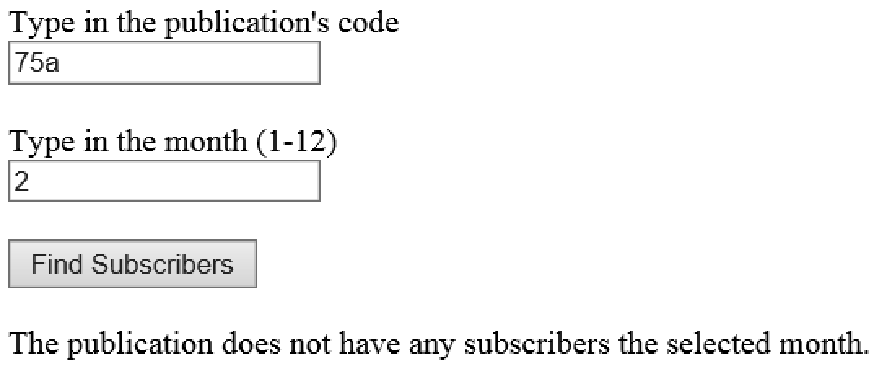
142

Type in the month (1-12)

2

The publication does not have any subscribers the selected month.

```
Įrašius 75a ir 2, rezultatas gaunamas toks pat:
```


Type in the publication's code

75a

Type in the month (1-12)

2

The publication does not have any subscribers the selected month.

Antras bandymas:
U3a.txt - tuščias;

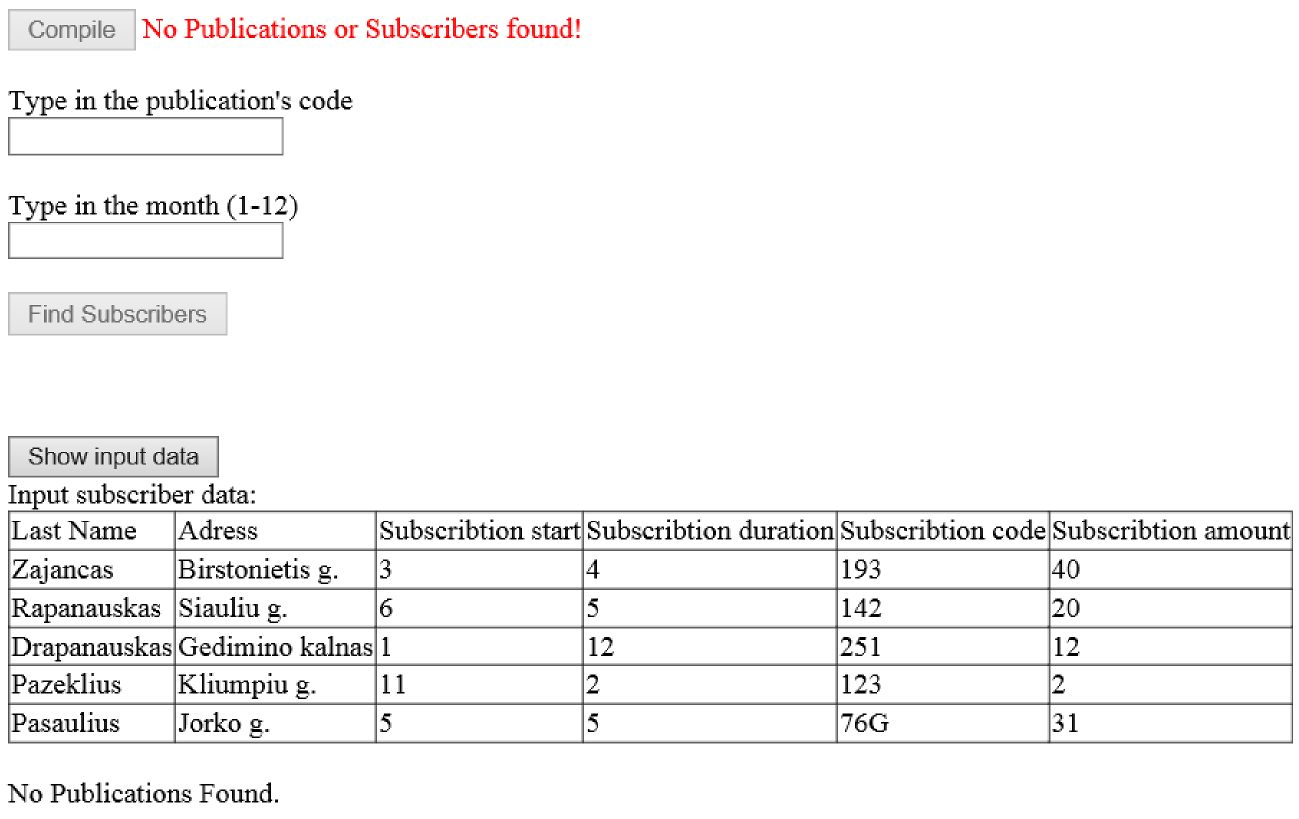
U3b.txt
```
Zajancas;Birstonietis g.;3;4;193;40
Rapanauskas;Siauliu g.;6;5;142;20
Drapanauskas;Gedimino kalnas;1;12;251;12
```

```
Pazeklius;Kliumpiu g.;11;2;123;2
Pasaulius;Jorko g.;5;5;76G;31
```

Answer.txt

```
LastName             | Adress               |   Subscribtion Start |
Subscribtion Duration |         Subscribtion Code |  Subscribtion Amount |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Zajancas             | Birstonietis g.      | 3                    | 4
| 193                       | 40                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Rapanauskas          | Siauliu g.           | 6                    | 5
| 142                       | 20                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Drapanauskas         | Gedimino kalnas      | 1                    | 12
| 251                       | 12                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Pazeklius            | Kliumpiu g.          | 11                   | 2
| 123                       | 2                      |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Pasaulius            | Jorko g.             | 5                    | 5
| 76G                       | 31                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------

No publications found
```

Vartotojo sąsaja:



No Publications or Subscribers found!

Type in the publication's code

Type in the month (1-12)

Input subscriber data:

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Zajancas	Birstonietis g.	3	4	193	40
Rapanauskas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	12	251	12
Pazeklius	Kliumpiu g.	11	2	123	2
Pasaulius	Jorko g.	5	5	76G	31

No Publications Found.

Trečias bandymas:

U3a.txt

```
123;Klaipedos L;19.99
142;Palangos Naujienos;25.50
```

```
193;Kauno min;14.19
662;Pasaulio Naujienos;40.20
251;Kauno Herbas;30.99
```

U3b.txt

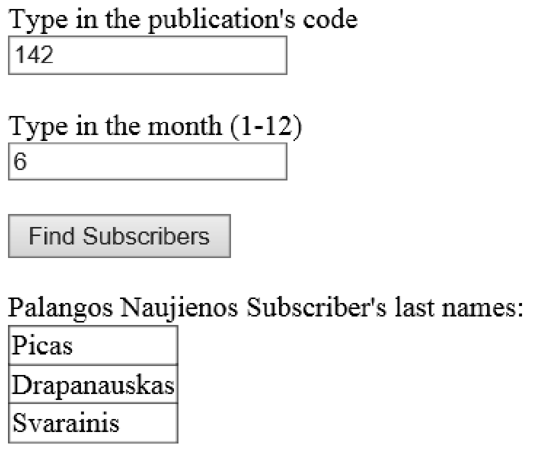
```
Kudlius;Birstonietis g.;3;4;193;40
Picas;Siauliu g.;6;5;142;20
Drapanauskas;Gedimino kalnas;1;10;142;12
Pazeklius;Kliumpiu g.;11;2;123;2
Pasaulius;Jorko g.;5;5;76G;31
Svarainis;Daukanto g.;4;10;142;31
Jonevicius;Paluonio g.;1;1;251;15
```

Answer.txt

```
Highest income every month:
1 . |       Kauno Herbas|
2 . |  Palangos Naujienos|
3 . |          Kauno min|
4 . |  Palangos Naujienos|
5 . |  Palangos Naujienos|
6 . |  Palangos Naujienos|
7 . |  Palangos Naujienos|
8 . |  Palangos Naujienos|
9 . |  Palangos Naujienos|
10. |  Palangos Naujienos|
11. |  Palangos Naujienos|
12. |  Palangos Naujienos|

All Publication income: 16330.21

Below average income publications:
-------------------------------------------
       193 | Kauno min           | 14.19 |
-------------------------------------------
       123 | Klaipedos L         | 19.99 |
-------------------------------------------
       251 | Kauno Herbas        | 30.99 |
-------------------------------------------
       662 | Pasaulio Naujienos  |  40.2 |
-------------------------------------------

LastName             | Adress                |   Subscribtion Start |
Subscribtion Duration |         Subscribtion Code |  Subscribtion Amount |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Kudlius              | Birstonietis g.      | 3                      | 4
| 193                    | 40                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Picas                | Siauliu g.           | 6                      | 5
| 142                    | 20                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Drapanauskas         | Gedimino kalnas      | 1                      | 10
| 142                    | 12                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Pazeklius            | Kliumpiu g.          | 11                     | 2
| 123                    | 2                      |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Pasaulius            | Jorko g.             | 5                      | 5
| 76G                    | 31                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
```

```
Svarainis            | Daukanto g.          | 4                      | 10
| 142                       | 31                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------
Jonevicius           | Paluonio g.          | 1                      | 1
| 251                       | 15                     |
----------------------------------------------------------------------------------
-----------------------------------------------------------------

Code       | Name                  | Price |
-------------------------------------------
       123 | Klaipedos L           | 19.99 |
-------------------------------------------
       142 | Palangos Naujienos    |  25.5 |
-------------------------------------------
       193 | Kauno min             | 14.19 |
-------------------------------------------
       662 | Pasaulio Naujienos    |  40.2 |
-------------------------------------------
       251 | Kauno Herbas          | 30.99 |
-------------------------------------------
```

Įvedus 142 į pirmą laukelį ir 6 į antrą, gaunamas toks atsakymas:



Type in the publication's code

142

Type in the month (1-12)

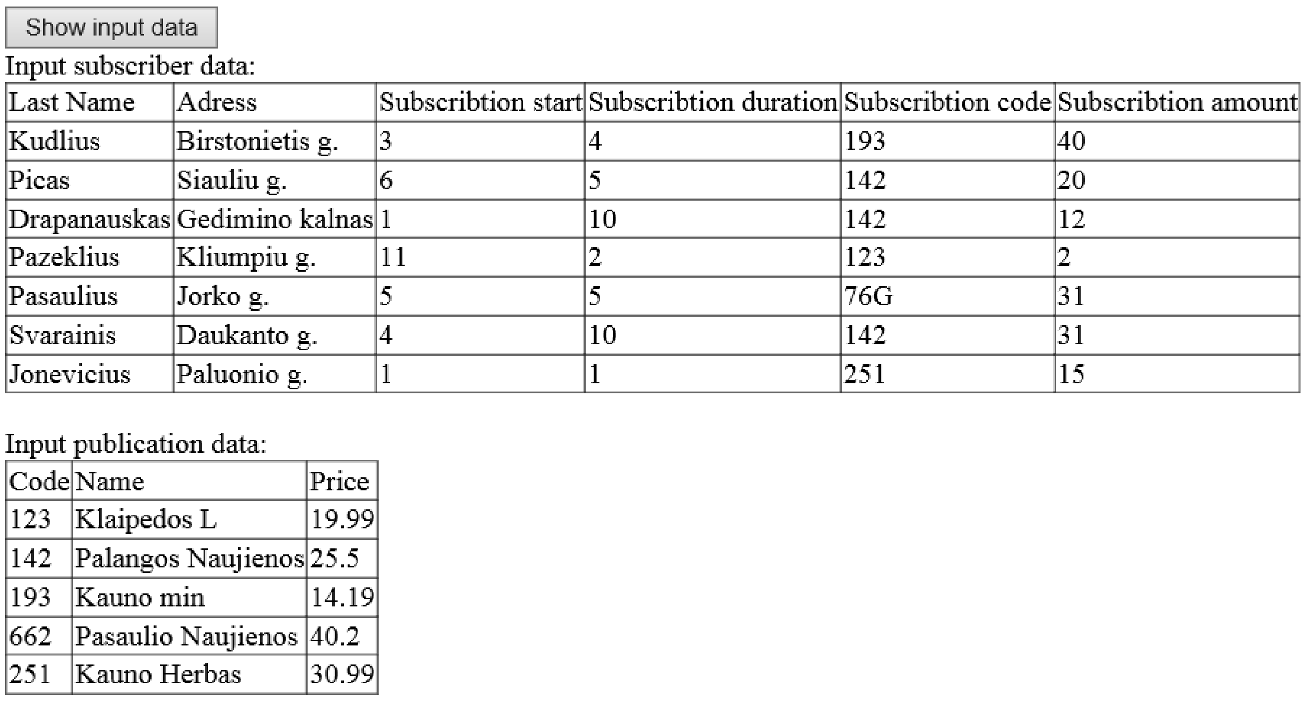
6

Find Subscribers

Palangos Naujienos Subscriber's last names:

Picas
Drapanauskas
Svarainis

Pradiniai duomenys lentele:



Show input data

Input subscriber data:

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Kudlius	Birstonietis g.	3	4	193	40
Picas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	10	142	12
Pazeklius	Kliumpiu g.	11	2	123	2
Pasaulius	Jorko g.	5	5	76G	31
Svarainis	Daukanto g.	4	10	142	31
Jonevicius	Paluonio g.	1	1	251	15

Input publication data:

Code	Name	Price
123	Klaipedos L	19.99
142	Palangos Naujienos	25.5
193	Kauno min	14.19
662	Pasaulio Naujienos	40.2
251	Kauno Herbas	30.99

2.8. Dėstytojo pastabos

Testo rezultatas: 1

3. Bendrinės klasės ir sąsajos (L3)

3.1. Darbo užduotis

LD_3. Leidiniai.

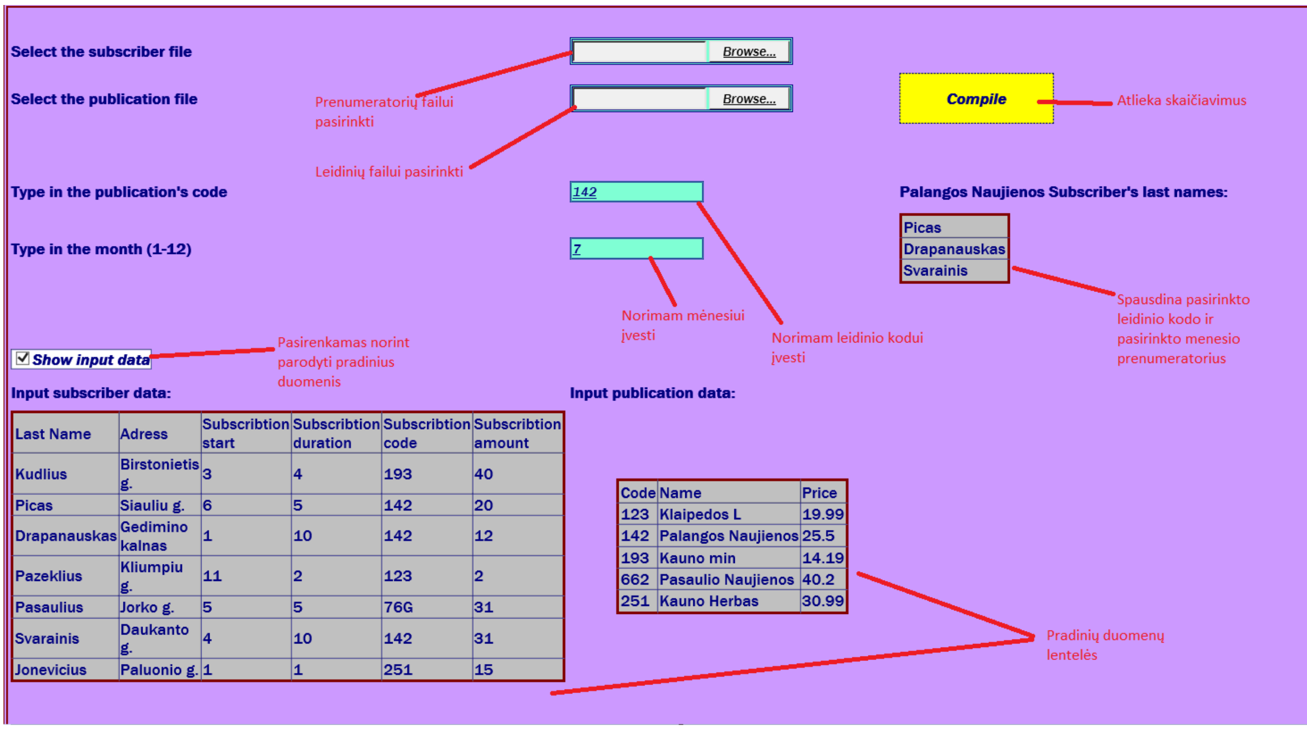
Žmonės užsisako spaudą. Užsakymas vyksta metų ribose. Leidiniai gauna dėl to pajamas. Nustatykite kiekvienam mėnesiui, kurio leidinio pajamos yra didžiausios. Nustatykite bendrąsias leidinių pajamas. Sudarykite sąrašą leidinių, kurių pajamos mažesnės už vidutines. Duomenys:

Tekstiniame faile U3a.txt yra tokia informacija apie leidinius: leidinio kodas, leidinio pavadinimas, vieno mėnesio leidinio kaina.

Tekstiniame faile U3b.txt yra informacija apie prenumeratorius: prenumeratoriaus pavardė, adresas, laikotarpio pradžia (sveikasis skaičius 1..12), laikotarpio ilgis, leidinio kodas, leidinių kiekis.

Spausdinamas sąrašas turi būti surikiuotas pagal vieno mėnesio leidinio kainą mažėjimo ir leidinio pavadinimą abėcėlės tvarka. Sudarykite nurodyto leidinio (įvedamas klaviatūra) nurodyto mėnesio ((įvedamas klaviatūra) prenumeratorių sąrašą.

3.2. Grafinės vartotojo sąsajos schema


Browse...
Browse...
Compile
Prenumeratorių failui pasirinkti
Leidinių failui pasirinkti
Atlieka skaičiavimus
142
7
Norimam mėnesiui įvesti
Norimam leidinio kodui įvesti
Spausdina pasirinkto leidinio kodo ir pasirinkto menesio prenumeratorius
Pasirenkamas norint parodyti pradinius duomenis
Show input data
Pradinių duomenų lentelės

Select the subscriber file

Select the publication file

Type in the publication's code

Type in the month (1-12)

Palangos Naujienos Subscriber's last names:

Picas
Drapanauskas
Svarainis

Input subscriber data:

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Kudlius	Birstonietis g.	3	4	193	40
Picas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	10	142	12
Pazeklius	Kliumpiu g.	11	2	123	2
Pasaulius	Jorko g.	5	5	76G	31
Svarainis	Daukanto g.	4	10	142	31
Jonevicius	Paluonio g.	1	1	251	15

Input publication data:

Code	Name	Price
123	Klaipedos L	19.99
142	Palangos Naujienos	25.5
193	Kauno min	14.19
662	Pasaulio Naujienos	40.2
251	Kauno Herbas	30.99

3.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Button1	Clickable	Atlieka skaičiavimus
Label1	Rodyti tekstą	Yra prieTextBox1, paaiškina ką rašyti
Label2	Rodyti tekstą	Yra prieTextBox2, paaiškina ką rašyti
Label3	Rodyti tekstą	Yra virš Table1, paaiškina kas vaizduojama
SubTLabel	Rodyti tekstą	Pradinių prenumeratorių duomenų lentelė
SubPLabel	Rodyti tekstą	Pradinių leidinių duomenų lentelė
Label4	Rodyti tekstą	Praneša vartotoją neradus duomenų
TextBox1	Įvesti tekstą	Leidinio kodo įvedimui
TextBox2	Įvesti tekstą	Mėnesio įvedimui

Table1	Rodytiduomenis lentele	Spausdinamas leidinio prenumeratorių sąrašas
TableSubs	Rodytiduomenis lentele	Pradinių duomenų prenumeratoriu lentelė
TablePubs	Rodytiduomenis lentele	Pradinių duomenų leidinių lentelė
CheckBox1	Pasirinkti	Pasirenkamas norint rodyti pradinius duomenis lentelėmis
FileUpload1	Ikeliamas failas	Naudojamas prenumeratoriu failui pasirinkti
FileUpload2	Ikeliamas failas	Naudojamas leidiniu failui pasirinkti

3.4. Klasių diagrama

Subscriber
+LastName : string +Adress : string +SubscribtionStart : integer +SubscribtionDuration : integer +SubscribtionCode : string +SubscribtionAmount : integer +Subscriber(in lastname :string, in adress :string, in subscribtionstart :integer, in subscribtionduration :integer, in subscribtioncode :string, in subscribtionamount :integer) +ToString() : string +Header() : string +CompareTo(in other :Subscriber) : integer +Equals(in other :Subscriber) : boolean

Publication
+Code : string +Name : string +Price : float +Income : float +Publication(in code :string, in name :string, in price :float) +CompareTo(in other :Publication) : integer +Equals(in other :Publication) : boolean +ToString() : string +Header() : string

KnotList
-first : Knot<type> -last : Knot<type> -current : Knot<type> +KnotList() +GetData() : type +AddToEnd(in newObject :type) +First() +Next() +Previous() +End() : boolean +Empty() : boolean +Sorting() +GetEnumerator() : IEnumerator<type>

Knot<type>
+data : type +next : Knot<type> +previous : Knot<type> +Knot(in input :type, in adrN :Knot<type>, in adrP :Knot<type>)

Forma.apsx

#Page_Load(in sender :object, in e :EventArgs)
#Button1_Click(in sender :object, in e :EventArgs)

```
-SubscriberInfo(in list :KnotList<Subscriber>, in File :Stream)
-PublicationInfo(in list :KnotList<Publication>, in File :Stream)
-MostIncomeByMonth(in P :KnotList<Publication>, in S :KnotList<Subscriber>)
-HighestIncome(in list :KnotList<Publication>) : Publication
-PrintBestMonthly(in pub :Publication, in month :integer)
-PublicationIncome(in Subs :KnotList<Subscriber>, in Publications :KnotList<Publication>)
-AllIncome(in Publications :KnotList<Publication>) : float
-LowIncomePublications(in All :KnotList<Publication>) : KnotList<Publication>
-Average(in All :KnotList<Publication>) : float
-SetIncomeToZero(in list :KnotList<Publication>)
-FindSelectedPubSubs(in Pubs :KnotList<Publication>, in Subs :KnotList<Subscriber>)
-FindPublicationWithCode(in list :KnotList<Publication>, in code :string) : Publication
-FindSubscribers(in Subs :KnotList<Subscriber>, in Code :string, in month :integer) : KnotList<Subscriber>
-PrintSubscribersToTable(in list :KnotList<Subscriber>, in pub :Publication)
-Print<type>(in file :string, in tableName :string, in tableHeader :string, in list :IEnumerable<type>)
-PrintData(in file :string, in AllIncome :float)
-PrintSubsToSubTable(in subs :KnotList<Subscriber>)
-PrintSPublicationToPubTable(in pubs :KnotList<Publication>)
-PrintInputData(in subs :KnotList<Subscriber>, in pubs :KnotList<Publication>, in answerFile :string)
-Checking(in Publications :KnotList<Publication>, in Subscribers :KnotList<Subscriber>) : boolean
```

3.5. *Programos vartotojo vadovas*

Pasirinkus prenumeratorių ir leidinių failus ir paspaudus Compile mygtuką, yra atliekami skaičiavimai. Norint rasti norimo leidinio prenumeratorių sąrašą norimą mėnesį, juos galima rasti pasinaudojus TextBox1 ir TextBox2. Bus spausdinama prenumeratorių lentelė. Pasirinkus Show Input Data check box, kitą kartą paspaudus kompiliavimo mygtuką, bus rodomi pradiniai duomenys

3.6. *Programos tekstas*

Publication.cs

```
public class Publication : IComparable<Publication>, IEquatable<Publication>
    {
        public string Code { get; set; }
        public string Name { get; set; }
        public double Price { get; set; }

        public double Income { get; set; }
        /// <summary>
        /// creates a new publication object
        /// </summary>
        /// <param name="code"></param>
        /// <param name="name"></param>
        /// <param name="price"></param>
        public Publication(string code, string name ,double price)
        {
            Code = code;
            Name = name;
            Price = price;

            Income = 0;
        }

        public int CompareTo(Publication other)
        {
            if (other == null)
                return 1;
            if (Price.CompareTo(other.Price) == 0)
            {
                return (Name.CompareTo(other.Name));
            }
            else return (Price.CompareTo(other.Price));
        }
        public bool Equals(Publication other)
        {
            if (other == null)
                return false;
            if (this.Name == other.Name && this.Code == other.Code)
                return true;
            else
                return false;
        }

        /// <summary>
        /// prinst all publication's data to one formated string
        /// </summary>
        /// <returns>formated publication's information</returns>
        public override string ToString()
        {
            return String.Format("{0, 10} | {1, -20} | {2, 5} |", Code, Name, Price);
        }
        /// <summary>
        /// prints formated header
        /// </summary>
        /// <returns>formated header string</returns>
        public string Header()
        {
            return String.Format("{0, -10} | {1, -20} | {2, -5} |","Code", "Name", "Price");
        }
    }
```

Subscriber.cs

```
public class Subscriber : IComparable<Subscriber>, IEquatable<Subscriber>
    {
        public string LastName { get; set; }
        public string Adress { get; set; }
        public int SubscribtionStart { get; set; }
        public int SubscribtionDuration { get; set; }
        public string SubscribtionCode { get; set; }
        public int SubscribtionAmount { get; set; }
        /// <summary>
        /// creats a new subscriber object
        /// </summary>
        /// <param name="lastname">subscriber's last name</param>
        /// <param name="adress">subscriber's adress</param>
        /// <param name="subscribtionstart">subscriber's subscribtion start</param>
        /// <param name="subscribtionduration">subscriber's subscribtion duration</param>
        /// <param name="subscribtioncode">subscriber's subscribtion code</param>
        /// <param name="subscribtionamount">subscriber's subscribtion amount</param>
        public Subscriber (string lastname, string adress, int subscribtionstart, int
              subscribtionduration, string subscribtioncode, int subscribtionamount)
        {
            LastName = lastname;
            Adress = adress;
            SubscribtionStart = subscribtionstart;
            SubscribtionDuration = subscribtionduration;
            SubscribtionCode = subscribtioncode;
            SubscribtionAmount = subscribtionamount;
        }
        /// <summary>
        /// prints subscriber's information to one formated string
        /// </summary>
        /// <returns>formated information</returns>
        public override string ToString()
        {
            return String.Format("{0, -20} | {1, -20} | {2, -20} | {3, -25} | {4, -25} | {5, -
              20} |", LastName, Adress, SubscribtionStart, SubscribtionDuration,
              SubscribtionCode, SubscribtionAmount);
        }
        /// <summary>
        /// prints the header of a table for a table of subscribers
        /// </summary>
        /// <returns>formated header</returns>
        public string Header()
        {
            return String.Format("{0, -20} | {1, -20} | {2, 20} | {3, 25} | {4, 25} | {5, 20}
              |", "LastName", "Adress", "Subscribtion Start", "Subscribtion Duration",
              "Subscribtion Code", "Subscribtion Amount");
        }

        public int CompareTo(Subscriber other)
        {
            if (other == null)
                return 1;
            if (LastName.CompareTo(other.LastName) == 0)
            {
                return (Adress.CompareTo(other.Adress));
            }
            else return (LastName.CompareTo(other.LastName));
        }

        public bool Equals(Subscriber other)
        {
            if (other == null)
                return false;
            if (this.LastName == other.LastName && this.Adress == other.Adress)
                return true;
            else
                return false;
        }
    }
```

KnotList.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Collections;

namespace _2Lab
{
    public class KnotList<type> : IEnumerable<type>
        where type : IComparable<type>, IEquatable<type>
    {
        private Knot<type> first;
        private Knot<type> last;
        private Knot<type> current;

        private sealed class Knot<type>
        {
            public type data { get; set; }
            public Knot<type> next { get; set; }
            public Knot<type> previous { get; set; }
            /// <summary>
            /// creates a new knot
            /// </summary>
            /// <param name="input">input data</param>
            /// <param name="adrN">previous knot</param>
            /// <param name="adrP">next knot</param>
            public Knot(type input, Knot<type> adrN, Knot<type> adrP)
            {
                data = input;
                previous = adrN;
                next = adrP;
            }
        }
        /// <summary>
        /// sets the first and last knots to null
        /// </summary>
        public KnotList()
        {
            this.first = this.last = current = null;
        }
        /// <summary>
        /// gets the current pointer's object's data
        /// </summary>
        /// <returns></returns>
        public type GetData()
        {
            return current.data;
        }
        /// <summary>
        /// Adds a object to the end of the list
        /// </summary>
        /// <param name="sub">subscriber data</param>
        public void AddToEnd(type newObject)
        {
            Knot<type> temp = new Knot<type>(newObject, last, null);
            if (first != null)
                last.next = temp;
            else
                first = temp;
            last = temp;
        }
        /// <summary>
        /// sets the current pointer to the first
        /// </summary>
        public void First()
        {
            current = first;
        }
        /// <summary>
        /// sets the current pointer to the next one
        /// </summary>
        public void Next()
        {
            current = current.next;
        }
        /// <summary>
        /// sets the current pointer to the previous one
        /// </summary>
        public void Previous()
        {
            current = current.previous;
        }
        /// <summary>
        /// checks of the current pointer is the last one
        /// </summary>
        /// <returns>true or false</returns>
        public bool End()
        {
            return current == null;
        }
        /// <summary>
        /// checks if the list is empty
        /// </summary>
        /// <returns>true or false</returns>
        public bool Empty()
        {
            return first == null;
        }
        /// <summary>
        /// sors the list
        /// </summary>
        public void Sorting()
        {
            for (Knot<type> n = first; n != null; n = n.next)
            {
                Knot<type> maxv = n;
                for (Knot<type> n2 = n; n2 != null; n2 = n2.next)
                    if (n2.data.CompareTo(maxv.data) < 0)
                        maxv = n2;
                type St = n.data;
                n.data = maxv.data;
                maxv.data = St;
            }
        }
        /// <summary>
        /// goes through list, saves last exit
        /// </summary>
        /// <returns>current one's data</returns>
        public IEnumerator<type> GetEnumerator()
        {
            for (Knot<type> dd = first; dd != null; dd = dd.next)
            {
                yield return dd.data;
            }
        }
        IEnumerator IEnumerable.GetEnumerator()
        {
            throw new NotImplementedException();
        }

    }
}
```

Forma.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Forma.aspx.cs"
Inherits="_2Lab.Forma" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style>
        .TextBoxStyle{
            border-style: double;
            background-color: aquamarine;
            font-style: italic;
            text-shadow: initial;
            font-family: "franklin Gothic Medium", "Arial Narrow", Arial, sans-serif;
            color: #000080;
            text-decoration: underline;
            cursor: text;
        }
        .auto-style1 {
            width: 100%;
        }
        .auto-style2 {
            width: 288px;
        }
        .auto-style3 {
            width: 288px;
            height: 34px;
        }
        .auto-style4 {
```

```
    height: 34px;
}
.auto-style5 {
    width: 317px;
}
.auto-style6 {
    height: 34px;
    width: 317px;
}
.auto-style7 {
    width: 288px;
    height: 26px;
}
.auto-style8 {
    width: 317px;
    height: 26px;
}
.auto-style9 {
    height: 26px;
}
.auto-style10 {
    width: 288px;
    height: 33px;
}
.auto-style11 {
    width: 317px;
    height: 33px;
}
.auto-style12 {
    height: 33px;
}
.ButtonStyle {
    font-family: "Franklin Gothic Medium", "Arial Narrow", Arial, sans-serif;
    font-weight: bold;
    font-style: oblique;
    font-variant: normal;
    text-transform: none;
    color: #000080;
    background-color: #FFFF00;
    padding: 0px;
    border: thin dashed #000080;
    font-size: inherit;
    cursor: pointer;
}
.LabelStyle {
    font-family: "Franklin Gothic Medium", "Arial Narrow", Arial, sans-serif;
    color: #000080;
    font-weight: bold;
}
.TableStyle {
    font-family: "Franklin Gothic Medium", "Arial Narrow", Arial, sans-serif;
    color: #000080;
    table-layout: fixed;
    border-collapse: collapse;
    border-spacing: 10px;
    border: 3px solid #800000;
    background-color: #C0C0C0;
}
.CheckBoxStyle {
    font-family: "Franklin Gothic Medium", "Arial Narrow", Arial, sans-serif;
    font-size: medium;
    font-style: oblique;
    font-weight: bolder;
    color: #000080;
    background-color: #FFFFFF;
    border-style: groove;
```

```
                border-width: thin;
                cursor: pointer;
            }
            .BackgroundStyle {
                background-color: #CC99FF;
                border: thick double #800000;
            }
            .IndividualCellColor {
                background-color: #9900CC;
            }
        </style>
    </head>
    <body>
        <form id="form1" runat="server">
            <div class="BackgroundStyle">
                <table class="auto-style1">
                    <tr>
                        <td class="auto-style2">&nbsp;</td>
                        <td class="auto-style5">
                            &nbsp;</td>
                        <td>&nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style3">
                            <asp:Label ID="Label5" runat="server" Text="Select the subscriber
                                file" CssClass="LabelStyle"></asp:Label>
                        </td>
                        <td class="auto-style6">
                <asp:FileUpload ID="FileUpload1" runat="server" CssClass="TextBoxStyle" />
                            <asp:CustomValidator ID="CustomValidator1" runat="server"
                                ControlToValidate="FileUpload1" ErrorMessage="*" ForeColor="Red"
                                ValidationGroup="Val 2"></asp:CustomValidator>
                        </td>
                        <td class="auto-style4">
                <asp:Label ID="Label4" runat="server" ForeColor="Red" Visible="False" Font-
                  Bold="True">No list found</asp:Label>
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            <asp:Label ID="Label6" runat="server" Text="Select the publication
                                file" CssClass="LabelStyle"></asp:Label>
                        </td>
                        <td class="auto-style5">
                            <asp:FileUpload ID="FileUpload2" runat="server"
                                CssClass="TextBoxStyle" />
                            <asp:CustomValidator ID="CustomValidator2" runat="server"
                                ControlToValidate="FileUpload2" ErrorMessage="*" ForeColor="Red"
                                ValidationGroup="Val 2"></asp:CustomValidator>
                        </td>
                        <td>
                <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="Compile"
                  ValidationGroup="Val 2" CssClass="ButtonStyle" Height="50px" Width="150px" />
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style2">&nbsp;</td>
                        <td class="auto-style5">&nbsp;</td>
                        <td>&nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style2">&nbsp;</td>
                        <td class="auto-style5">&nbsp;</td>
                        <td>&nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style3">
                <asp:Label ID="Label1" runat="server" Text="Type in the publication's code"
                  CssClass="LabelStyle"></asp:Label>
                        </td>
                        <td class="auto-style6">
                <asp:TextBox ID="TextBox1" runat="server" CssClass="TextBoxStyle"></asp:TextBox>
                <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
                  ControlToValidate="TextBox1" ErrorMessage="Code required" ForeColor="Red"
                  ValidationGroup="Val1" Enabled="False">*</asp:RequiredFieldValidator>
                        </td>
                        <td class="auto-style4">
                <asp:Label ID="Label3" runat="server" Visible="False"
                  CssClass="LabelStyle"></asp:Label>
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                <asp:Label ID="Label2" runat="server" Text="Type in the month (1-12)"
                  CssClass="LabelStyle"></asp:Label>
                        </td>
                        <td class="auto-style5">
                <asp:TextBox ID="TextBox2" runat="server" CssClass="TextBoxStyle"></asp:TextBox>
                <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
                  ControlToValidate="TextBox2" ErrorMessage="Month required" ForeColor="Red"
                  ValidationGroup="Val1" Enabled="False">*</asp:RequiredFieldValidator>
                        </td>
                        <td>
                <asp:Table ID="Table1" runat="server" GridLines="Both" CssClass="TableStyle">
                </asp:Table>
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style7">
                            &nbsp;</td>
                        <td class="auto-style8"></td>
                        <td class="auto-style9"></td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                <asp:ValidationSummary ID="ValidationSummary1" runat="server" ForeColor="Red"
                  ValidationGroup="Val1" />
                        </td>
                        <td class="auto-style5">&nbsp;</td>
                        <td>&nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style10">
                            <asp:CheckBox ID="CheckBox1" runat="server" Text="Show input data"
                                CssClass="CheckBoxStyle" />
                        </td>
                        <td class="auto-style11">&nbsp;</td>
                        <td class="auto-style12"></td>
                    </tr>
                    <tr>
                        <td class="auto-style7">
                            <asp:Label ID="SubTLabel" runat="server" Visible="False"
                                CssClass="LabelStyle"></asp:Label>
                            <br />
                            </td>
                        <td class="auto-style8">
                            <asp:Label ID="PubTLabel" runat="server" Visible="False"
                                BorderStyle="None" CssClass="LabelStyle"></asp:Label>
                            <br />
                            </td>
                        <td class="auto-style9"></td>
                    </tr>
```

```
                <tr>
                    <td class="auto-style2">
                        <asp:Table ID="TableSubs" runat="server" Visible="False"
                            GridLines="Both" CssClass="TableStyle" HorizontalAlign="Center">
                        </asp:Table>
                        </td>
                    <td class="auto-style5">
                        <asp:Table ID="TablePubs" runat="server" Visible="False"
                            GridLines="Both" CssClass="TableStyle" HorizontalAlign="Center">
                        </asp:Table>
                        </td>
                    <td>&nbsp;</td>
                </tr>
                <tr>
                    <td class="auto-style7">
                        </td>
                    <td class="auto-style8"></td>
                    <td class="auto-style9"></td>
                </tr>
                <tr>
                    <td class="auto-style2">
                        &nbsp;</td>
                    <td class="auto-style5">&nbsp;</td>
                    <td>&nbsp;</td>
                </tr>
                <tr>
                    <td class="auto-style2">
                        &nbsp;</td>
                    <td class="auto-style5">&nbsp;</td>
                    <td>&nbsp;</td>
                </tr>
                <tr>
                    <td class="auto-style2">
                        <br />
                        <br />
                        </td>
                    <td class="auto-style2">
                        <br />
                        </td>
                    <td>
                        <br />
                    </td>
                </tr>
                <tr>
                    <td class="auto-style2">
                        <br />
                        </td>
                    <td class="auto-style5">
                        &nbsp;</td>
                    <td>
                        &nbsp;</td>
                </tr>
            </table>
        </div>
    </form>
</body>
</html>
```

Forma.aspx.cs

```
public partial class Forma : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        /// <summary>
        /// Compiles the program
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        protected void Button1_Click(object sender, EventArgs e)
        {
            const string answerFile = "Answer.txt";
            File.Delete(Server.MapPath(answerFile));

            var Subscribers = new KnotList<Subscriber>();
            var Publications = new KnotList<Publication>();
            if (FileUpload1.HasFile != false && FileUpload2.HasFile != false)
            {
                PublicationInfo(Publications, FileUpload2.FileContent);
                SubscriberInfo(Subscribers, FileUpload1.FileContent);
                Label4.Visible = false;
            }
            else if (FileUpload1.HasFile == false || FileUpload2.HasFile == false)
                Label4.Visible = true;
            PrintInputData(Subscribers, Publications, answerFile);
            if (!Publications.Empty() && !Subscribers.Empty())
            {
                MostIncomeByMonth(Publications, Subscribers);

                PublicationIncome(Subscribers, Publications);
                double AllPubIncome = AllIncome(Publications);
                PrintData(answerFile, AllPubIncome);

                KnotList<Publication> LowIncomePubs = LowIncomePublications(Publications);
                LowIncomePubs.Sorting();
                LowIncomePubs.First();
                Print(answerFile, "Below average income publications",
                     LowIncomePubs.GetData().Header(), LowIncomePubs);

                FindSelectedPubSubs(Publications, Subscribers);

                Label4.Visible = false;
            }
            else if (Publications.Empty() || Subscribers.Empty())
                Label4.Visible = true;

        }

        #region ReadData
        /// <summary>
        /// Reads subscriber information from file
        /// and adds to the end of the list
        /// </summary>
        /// <param name="list">the list starting list</param>
        /// <returns>a list of subscribers with their information</returns>
        private void SubscriberInfo(KnotList<Subscriber> list, Stream File)
        {
            using (StreamReader sr = new StreamReader(File))
            {
                string line;
                while ((line = sr.ReadLine()) != null)
                {
                    string[] values = line.Split(';');
                    string lastName = values[0];
                    string city = values[1];
                    int beginning = int.Parse(values[2]);
                    int duration = int.Parse(values[3]);
                    string code = values[4];
                    int amount = int.Parse(values[5]);
                    Subscriber sub = new Subscriber(lastName, city, beginning, duration, code,
                     amount);
```

```
                list.AddToEnd(sub);
            }
        }
    }
    /// <summary>
    /// Reads publication informations from file
    /// and adds to the end of the given list
    /// </summary>
    /// <param name="list">a list that needs to be filled</param>
    /// <returns>updated list</returns>
    private void PublicationInfo(KnotList<Publication> list, Stream File)
    {
        using (StreamReader sr = new StreamReader(File))
        {
            string line;
            while ((line = sr.ReadLine()) != null)
            {
                string[] values = line.Split(';');
                string code = values[0];
                string name = values[1];
                double price = double.Parse(values[2]);
                Publication publication = new Publication(code, name, price);
                list.AddToEnd(publication);
            }
        }
    }
    #endregion

    #region Income by month
    /// <summary>
    /// Goes month by month, publication by publication
    /// and searches for it's subscribers and calculates
    /// the month's income. makes a list of all publications
    /// and their incomes each month
    /// </summary>
    /// <param name="P">list of publications</param>
    /// <param name="S">list of subscribers</param>
    private void MostIncomeByMonth(KnotList<Publication> P, KnotList<Subscriber> S)
    {
        for (int month = 1; month <= 12; month++)
        {
            KnotList<Publication> monthly = new KnotList<Publication>();
            SetIncomeToZero(P);
            for (P.First(); !P.End(); P.Next()) //eina per leidinius
            {

                for (S.First(); !S.End(); S.Next()) //eina per prenumeratorius
                    if (P.GetData().Code == S.GetData().SubscribtionCode)
                        if ((month >= S.GetData().SubscribtionStart) && (month <=
                              S.GetData().SubscribtionStart +
                              S.GetData().SubscribtionDuration - 1))
                            P.GetData().Income += S.GetData().SubscribtionAmount *
                              P.GetData().Price;
                monthly.AddToEnd(P.GetData());
            }
            Publication maxPub = HighestIncome(monthly);
            PrintBestMonthly(maxPub, month);
        }
    }
    /// <summary>
    /// Compares the publications
    /// finds the publication with the most income
    /// that month
    /// </summary>
    /// <param name="list">list of publications</param>
    /// <returns>the publication with the highest income</returns>
```

```
private Publication HighestIncome(KnotList<Publication> list)
{
    double max = 0;
    Publication maxPub = null;
    for (list.First(); !list.End(); list.Next())
        if (list.GetData().Income > max)
        {
            max = list.GetData().Income;
            maxPub = list.GetData();
        }
    return maxPub;
}
/// <summary>
/// Adds the publication with the highest income each month
/// to file
/// </summary>
/// <param name="pub"></param>
/// <param name="month"></param>
private void PrintBestMonthly(Publication pub, int month)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath("Answer.txt"), true))
    {
        if (month == 1)
            sw.WriteLine("Highest income every month:");
        if (pub == null)
            sw.WriteLine("{0, -2}. |{1, 20}|", month, "Nera");
        else
            sw.WriteLine("{0, -2}. |{1, 20}|", month, pub.Name);
    }
}



#endregion

#region All Publication Income
/// <summary>
/// goes through all publications, finds their subscribers
///  and calculates the publication's info
/// </summary>
/// <param name="Subs">list of subscribers</param>
/// <param name="Publications">list of publications</param>
private void PublicationIncome(KnotList<Subscriber> Subs, KnotList<Publication>
      Publications)
{
    SetIncomeToZero(Publications);
    for (Publications.First(); !Publications.End(); Publications.Next())
        for (Subs.First(); !Subs.End(); Subs.Next())
            if (Subs.GetData().SubscribtionCode == Publications.GetData().Code)
                Publications.GetData().Income += Publications.GetData().Price *
                    Subs.GetData().SubscribtionAmount *
                    Subs.GetData().SubscribtionDuration;
}
/// <summary>
/// sums up all of the publication's income
/// </summary>
/// <param name="Publications">list of publications</param>
/// <returns>the sum of all publication's income</returns>
private double AllIncome(KnotList<Publication> Publications)
{
    double sum = 0;
    for (Publications.First(); !Publications.End(); Publications.Next())
        sum += Publications.GetData().Income;
    return sum;
}
#endregion
```

```
#region Low Income Publications
/// <summary>
/// goes through all publications and adds
/// the publications with below average income to a new list
/// </summary>
/// <param name="All">list of publications</param>
/// <returns>a list of publications with below average income</returns>
private KnotList<Publication> LowIncomePublications(KnotList<Publication> All)
{
    double average = Average(All);
    KnotList<Publication> LowIncomePubs = new KnotList<Publication>();
    for (All.First(); !All.End(); All.Next())
        if (All.GetData().Income < average)
            LowIncomePubs.AddToEnd(All.GetData());
    return LowIncomePubs;
}

/// <summary>
/// finds the average income of all publications
/// </summary>
/// <param name="All">list of publications</param>
/// <returns>the average income</returns>
private double Average(KnotList<Publication> All)
{
    double sum = 0;
    int k = 0;
    for (All.First(); !All.End(); All.Next())
    {
        sum += All.GetData().Income;
        k++;
    }
    return sum / k;
}
private void SetIncomeToZero(KnotList<Publication> list)
{
    for (list.First(); !list.End(); list.Next())
        list.GetData().Income = 0;
}
#endregion

#region Selected Publication and month
/// <summary>
/// Finds the selected code's publications subscribers
/// </summary>
/// <param name="Pubs">list of publications</param>
/// <param name="Subs">list of subscribers</param>
private void FindSelectedPubSubs(KnotList<Publication> Pubs, KnotList<Subscriber>
      Subs)
{
    if (TextBox1.Text == "" || TextBox2.Text == "")
    {
        KnotList<Subscriber> a = new KnotList<Subscriber>();
        PrintSubscribersToTable(a, null);
    }
    else if (TextBox1.Text != "" && TextBox2.Text != "")
    {
        string SelectedCode = TextBox1.Text;
        int SelectedMonth = int.Parse(TextBox2.Text);
        Publication foundPubl = FindPublicationWithCode(Pubs, SelectedCode);
        KnotList<Subscriber> PubSubs = FindSubscribers(Subs, SelectedCode,
             SelectedMonth);
        PrintSubscribersToTable(PubSubs, foundPubl);
    }
}
/// <summary>
/// using the selected code, finds the publication
/// </summary>
/// <param name="list">list of publications</param>
/// <param name="code">the selected code</param>
/// <returns>the publication that was found using the code</returns>
private Publication FindPublicationWithCode(KnotList<Publication> list, string code)
{
    for (list.First(); !list.End(); list.Next())
        if (code == list.GetData().Code)
            return list.GetData();
    return null;
}
/// <summary>
/// finds the subscribers for the selected publication the selected month
/// </summary>
/// <param name="Subs">list o subscribers</param>
/// <param name="Code">the selected code</param>
/// <param name="month">the selected month</param>
/// <returns>a list of subscribers of the selected publication
/// the selected month</returns>
private KnotList<Subscriber> FindSubscribers(KnotList<Subscriber> Subs, string Code,
      int month)
{
    KnotList<Subscriber> PubSubs = new KnotList<Subscriber>();
    for (Subs.First(); !Subs.End(); Subs.Next())
        if (Subs.GetData().SubscribtionCode == Code)
            if (month >= Subs.GetData().SubscribtionStart && month <=
             Subs.GetData().SubscribtionStart + Subs.GetData().SubscribtionDuration -
             1)
                PubSubs.AddToEnd(Subs.GetData());
    return PubSubs;
}
/// <summary>
/// prints the found subscribers to a table
/// </summary>
/// <param name="list">list of subscribers</param>
/// <param name="pubName">the name of the publication</param>
private void PrintSubscribersToTable(KnotList<Subscriber> list, Publication pub)
{
    if (!list.Empty() && pub != null)
        for (list.First(); !list.End(); list.Next())
        {
            Label3.Visible = true;
            Label3.Text = pub.Name + " Subscriber's last names:";
            TableCell cell = new TableCell();
            string tempstring = String.Format("{0}", list.GetData().LastName);
            cell.Text = tempstring;

            TableRow row = new TableRow();
            row.Cells.Add(cell);

            Table1.Rows.Add(row);
        }
    else
    {
        Label3.Visible = true;
        Label3.Text = "The publication does not have any subscribers the selected
             month.";
    }
}

#endregion

#region PrintResults
/// <summary>
/// Prints the list to selected file
/// </summary>
/// <typeparam name="type">type of list</typeparam>
/// <param name="file">path of wanted answer file</param>
/// <param name="tableName">Header of the table</param>
/// <param name="tableHeader">header of the table</param>
/// <param name="list">list that is being printed</param>
private void Print<type>(string file, string tableName, string tableHeader,
      IEnumerable<type> list)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(@file), true))
    {
        if (list == null)
        {
            sw.WriteLine("No list found");
            sw.WriteLine();
        }
        else
        {
            string line = new string('-', tableHeader.Length);
            sw.WriteLine(tableName);

            sw.WriteLine(tableHeader);
            sw.WriteLine(line);
            foreach (type a in list)
            {
                sw.WriteLine(a.ToString());
                sw.WriteLine(line);
            }
            sw.WriteLine();
        }
    }
}

/// <summary>
/// prints all income to file
/// </summary>
/// <param name="file">file path</param>
/// <param name="AllIncome">number</param>
private void PrintData(string file, double AllIncome)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(@file), true))
    {
        sw.WriteLine();
        sw.WriteLine("All Publication income: {0:F2}", AllIncome);
        sw.WriteLine();
    }
}
/// <summary>
/// prints subscriber data to table in web
/// </summary>
/// <param name="list">list of subscribers</param>
private void PrintSubsToSubTable(KnotList<Subscriber> subs)
{
    if (!subs.Empty())
    {
        SubTLabel.Text = "Input subscriber data:";
        TableRow row = new TableRow();
        TableCell[] cell = new TableCell[6];
        for (int i = 0; i < 6; i++)
            cell[i] = new TableCell();
        cell[0].Text = "Last Name";
        cell[1].Text = "Adress";
        cell[2].Text = "Subscribtion start";
        cell[3].Text = "Subscribtion duration";
        cell[4].Text = "Subscribtion code";
        cell[5].Text = "Subscribtion amount";
```

```
            row.Cells.AddRange(cell);
            TableSubs.Rows.Add(row);
            for (subs.First(); !subs.End(); subs.Next())
            {
                row = new TableRow();
                for (int i = 0; i < 6; i++)
                    cell[i] = new TableCell();
                cell[0].Text = subs.GetData().LastName;
                cell[1].Text = subs.GetData().Adress;
                cell[2].Text = subs.GetData().SubscribtionStart.ToString();
                cell[3].Text = subs.GetData().SubscribtionDuration.ToString();
                cell[4].Text = subs.GetData().SubscribtionCode;
                cell[5].Text = subs.GetData().SubscribtionAmount.ToString();
                row.Cells.AddRange(cell);
                TableSubs.Rows.Add(row);
            }
        }
        else
        {
            SubTLabel.Text = "No Subscribers Found";
        }

    }
    /// <summary>
    /// prints publication data to web
    /// </summary>
    /// <param name="list">list of publiactions</param>
    private void PrintSPublicationToPubTable(KnotList<Publication> pubs)
    {
        if (!pubs.Empty())
        {
            PubTLabel.Text = "Input publication data:";
            TableRow row = new TableRow();
            TableCell[] cell = new TableCell[3];
            for (int i = 0; i < 3; i++)
                cell[i] = new TableCell();
            cell[0].Text = "Code";
            cell[1].Text = "Name";
            cell[2].Text = "Price";
            row.Cells.AddRange(cell);
            TablePubs.Rows.Add(row);
            for (pubs.First(); !pubs.End(); pubs.Next())
            {
                row = new TableRow();
                for (int i = 0; i < 3; i++)
                    cell[i] = new TableCell();
                cell[0].Text = pubs.GetData().Code;
                cell[1].Text = pubs.GetData().Name;
                cell[2].Text = pubs.GetData().Price.ToString();
                row.Cells.AddRange(cell);
                TablePubs.Rows.Add(row);
            }
        }
        else
        {
            PubTLabel.Text = "No Publications Found.";
        }

    }
    /// <summary>
    /// calls the prmethods
    /// </summary>
    /// <param name="subs">subscriber list</param>
    /// <param name="pubs">publication list</param>
    /// <param name="answerFile">asnwer file path</param>
```

```
        private void PrintInputData(KnotList<Subscriber> subs, KnotList<Publication> pubs,
              string answerFile)
        {
            PrintSubsToSubTable(subs);
            PrintSPublicationToPubTable(pubs);
            subs.First();
            pubs.First();
            if(!subs.Empty())
            Print(answerFile, "", subs.GetData().Header(), subs);
            else
                Print(answerFile, "", "", subs);
            if (!pubs.Empty())
            Print(answerFile, "", pubs.GetData().Header(), pubs);
            else
                Print(answerFile, "", "", pubs);
            if (CheckBox1.Checked == true)
            {
                TableSubs.Visible = TablePubs.Visible = true;
                SubTLabel.Visible = PubTLabel.Visible = true;
            }
        }
        #endregion

        #region Checking
        /// <summary>
        /// checks if the starting lists are empty
        /// and if so, disables the buttons, alerts the user
        /// </summary>
        /// <param name="Publications">list of publications</param>
        /// <param name="Subscribers">list of subscribers</param>
        private bool Checking(KnotList<Publication> Publications, KnotList<Subscriber>
              Subscribers)
        {
            if (Publications.Empty() || Subscribers.Empty())
            {
                Label4.Text = "No Publications or Subscribers found!";
                return false;
            }
            if (FileUpload1.HasFile == false || FileUpload2.HasFile == false)
            {
                Label4.Text = "No Publications or Subscribers found!";
                return false;
            }
            return true;
        }
        #endregion

    }
```

3.7. Pradiniai duomenys ir rezultatai

Pirmas bandymas

U3a.txt

```
123;Klaipedos L;19.99
142;Palangos Naujienos;25.50
193;Kauno min;14.19
662;Pasaulio Naujienos;40.20
251;Kauno Herbas;30.99
```

U3b.txt

```
Kudlius;Birstonietis g.;3;4;193;40
Picas;Siauliu g.;6;5;142;20
Drapanauskas;Gedimino kalnas;1;10;142;12
Pazeklius;Kliumpiu g.;11;2;123;2
Pasaulius;Jorko g.;5;5;76G;31
Svarainis;Daukanto g.;4;10;142;31
Jonevicius;Paluonio g.;1;1;251;15
```

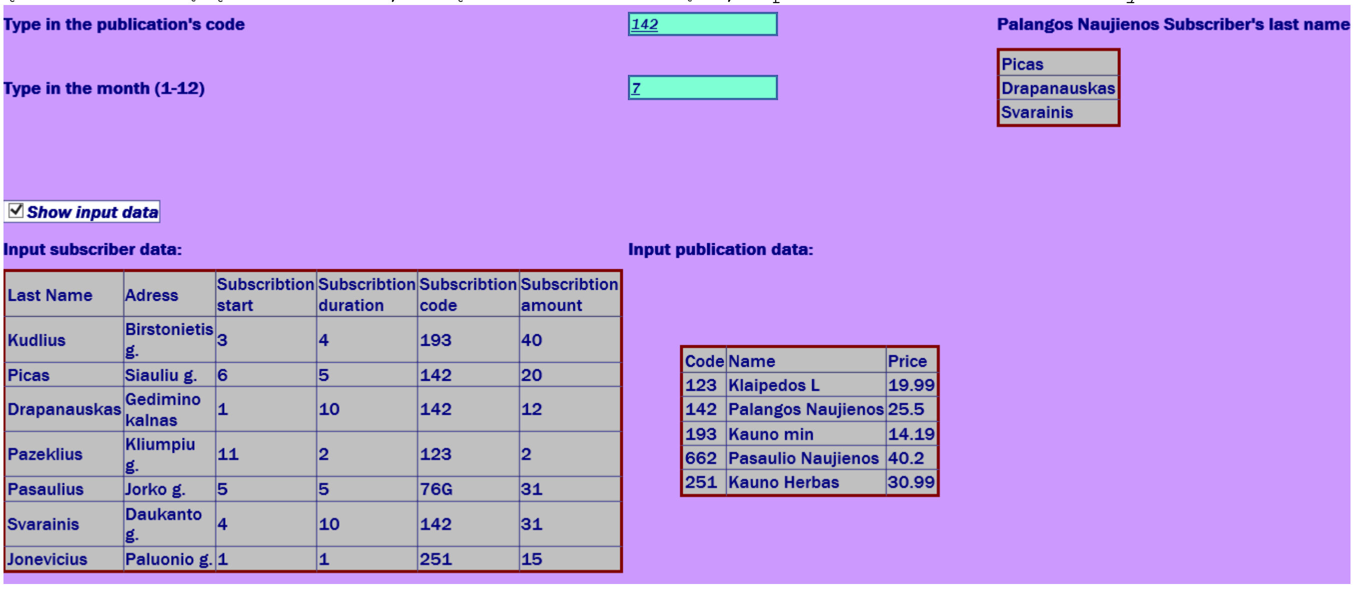
Answer.txt

```
LastName             | Adress                 |   Subscribtion Start |
Subscribtion Duration |         Subscribtion Code |  Subscribtion Amount |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Kudlius              | Birstonietis g.        | 3                      | 4
| 193                    | 40                     |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Picas                | Siauliu g.             | 6                      | 5
| 142                    | 20                     |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Drapanauskas         | Gedimino kalnas        | 1                      | 10
| 142                    | 12                     |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Pazeklius            | Kliumpiu g.            | 11                     | 2
| 123                    | 2                      |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Pasaulius            | Jorko g.               | 5                      | 5
| 76G                    | 31                     |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Svarainis            | Daukanto g.            | 4                      | 10
| 142                    | 31                     |
--------------------------------------------------------------------------------
-----------------------------------------------------------------
Jonevicius           | Paluonio g.            | 1                      | 1
| 251                    | 15                     |
--------------------------------------------------------------------------------
-----------------------------------------------------------------


Code       | Name                  | Price |
-------------------------------------------
       123 | Klaipedos L           | 19.99 |
-------------------------------------------
       142 | Palangos Naujienos    |  25.5 |
-------------------------------------------
       193 | Kauno min             | 14.19 |
-------------------------------------------
       662 | Pasaulio Naujienos    |  40.2 |
-------------------------------------------
       251 | Kauno Herbas          | 30.99 |
-------------------------------------------

Highest income every month:
1 . |        Kauno Herbas|
2 . |  Palangos Naujienos|
3 . |           Kauno min|
4 . |  Palangos Naujienos|
5 . |  Palangos Naujienos|
6 . |  Palangos Naujienos|
7 . |  Palangos Naujienos|
8 . |  Palangos Naujienos|
9 . |  Palangos Naujienos|
10. |  Palangos Naujienos|
11. |  Palangos Naujienos|
12. |  Palangos Naujienos|

All Publication income: 16330.21
```

```
Below average income publications
Code        | Name                      | Price |
--------------------------------------------
       193 | Kauno min                 | 14.19 |
--------------------------------------------
       123 | Klaipedos L               | 19.99 |
--------------------------------------------
       251 | Kauno Herbas              | 30.99 |
--------------------------------------------
       662 | Pasaulio Naujienos        |  40.2 |
--------------------------------------------
```

Vartotojo sąsaja

Į kodo lauką įrašius 142, o į mėnesio lauką 7, spausdinamas toks atsakymas:



Type in the publication's code 142

Type in the month (1-12) 7

Palangos Naujienos Subscriber's last name

Picas
Drapanauskas
Svarainis

✓ ***Show input data***

Input subscriber data:

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Kudlius	Birstonietis g.	3	4	193	40
Picas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	10	142	12
Pazeklius	Kliumpiu g.	11	2	123	2
Pasaulius	Jorko g.	5	5	76G	31
Svarainis	Daukanto g.	4	10	142	31
Jonevicius	Paluonio g.	1	1	251	15

Input publication data:

Code	Name	Price
123	Klaipedos L	19.99
142	Palangos Naujienos	25.5
193	Kauno min	14.19
662	Pasaulio Naujienos	40.2
251	Kauno Herbas	30.99

Antras bandymas

U3a.txt – tuščias

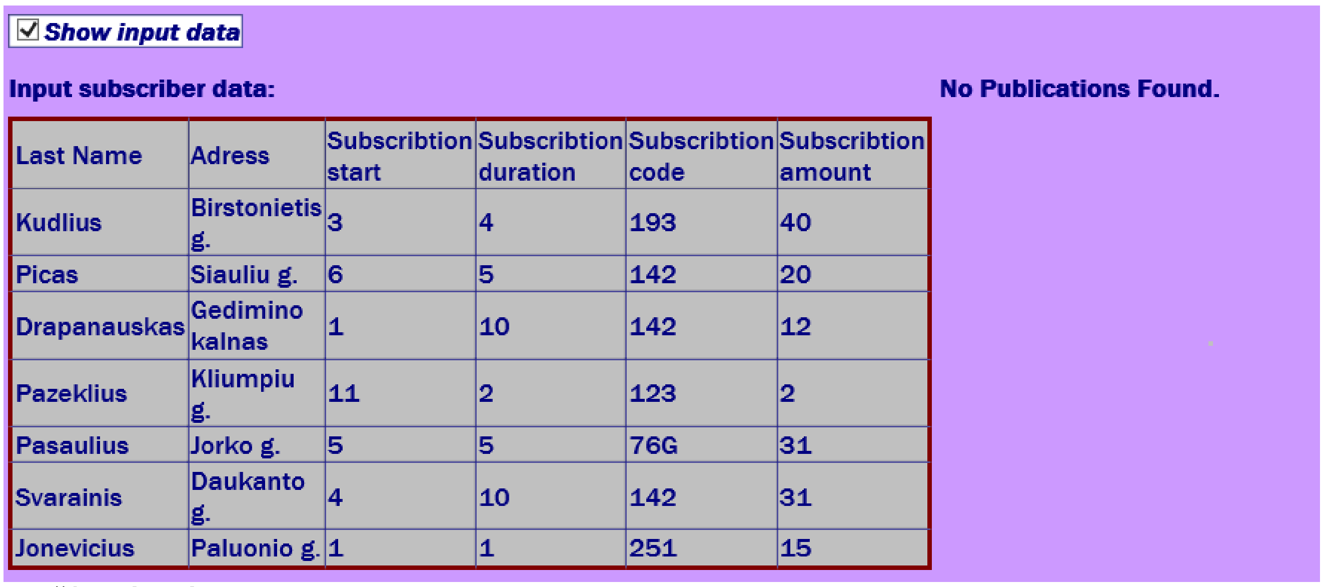
U3b.txt

```
Kudlius;Birstonietis g.;3;4;193;40
Picas;Siauliu g.;6;5;142;20
Drapanauskas;Gedimino kalnas;1;10;142;12
Pazeklius;Kliumpiu g.;11;2;123;2
Pasaulius;Jorko g.;5;5;76G;31
Svarainis;Daukanto g.;4;10;142;31
Jonevicius;Paluonio g.;1;1;251;15
```

Answer.txt

```
LastName             | Adress                |   Subscribtion Start |     Subscribtion Duration |         Subscribtion Code |  Subscribtion Amount |
----------------------------------------------------------------------------------------------------------------------------------------------------
Kudlius              | Birstonietis g.       | 3                    | 4                          | 193                        | 40                    |
----------------------------------------------------------------------------------------------------------------------------------------------------
Picas                | Siauliu g.            | 6                    | 5                          | 142                        | 20                    |
----------------------------------------------------------------------------------------------------------------------------------------------------
Drapanauskas         | Gedimino kalnas       | 1                    | 10                         | 142                        | 12                    |
----------------------------------------------------------------------------------------------------------------------------------------------------
Pazeklius            | Kliumpiu g.           | 11                   | 2                          | 123                        | 2                     |
----------------------------------------------------------------------------------------------------------------------------------------------------
Pasaulius            | Jorko g.              | 5                    | 5                          | 76G                        | 31                    |
----------------------------------------------------------------------------------------------------------------------------------------------------
Svarainis            | Daukanto g.           | 4                    | 10                         | 142                        | 31                    |
----------------------------------------------------------------------------------------------------------------------------------------------------
Jonevicius           | Paluonio g.           | 1                    | 1                          | 251                        | 15                    |
----------------------------------------------------------------------------------------------------------------------------------------------------
```

Vartotojo sąsaja



☑ ***Show input data***

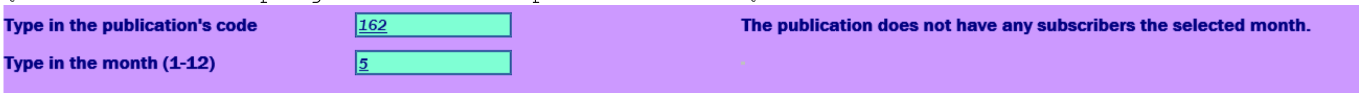
Input subscriber data:

No Publications Found.

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Kudlius	Birstonietis g.	3	4	193	40
Picas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	10	142	12
Pazeklius	Kliumpiu g.	11	2	123	2
Pasaulius	Jorko g.	5	5	76G	31
Svarainis	Daukanto g.	4	10	142	31
Jonevicius	Paluonio g.	1	1	251	15

Trečias bandymas:

U3a.txt ir U3b.txt yra tokie patys, kaip pirmame bandyme, tačiau į teksto laukus įvedus 162 ir 5 programa neranda prenumeratorių.



Type in the publication's code	162
Type in the month (1-12)	5

The publication does not have any subscribers the selected month.

Ketvirtas bandymas:

U3a.txt

```
123;Klaipedos L;19.99
142;Palangos Naujienos;25.50
193;Kauno min;14.19
662;Pasaulio Naujienos;40.20
251;Kauno Herbas;30.99
```

U3b.txt

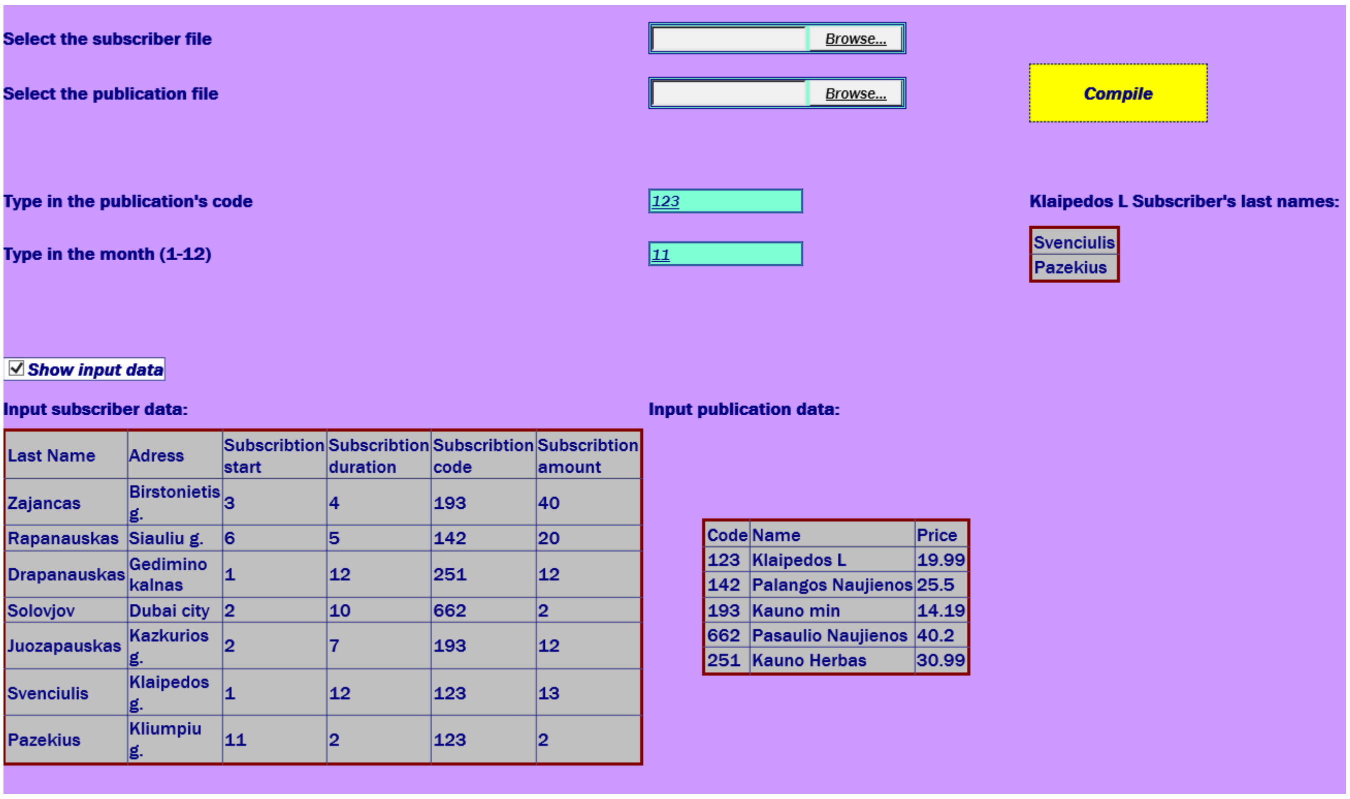
```
Zajancas;Birstonietis g.;3;4;193;40
Rapanauskas;Siauliu g.;6;5;142;20
Drapanauskas;Gedimino kalnas;1;12;251;12
Solovjov;Dubai city;2;10;662;2
Juozapauskas;Kazkurios g.;2;7;193;12
Svenciulis;Klaipedos g.;1;12;123;13
Pazekius;Kliumpiu g.;11;2;123;2
```

Answer.txt

```
LastName           | Adress              |  Subscribtion Start |  Subscribtion Duration |       Subscribtion Code |
Subscribtion Amount |
--------------------------------------------------------------------------------------------------------------------------------
Zajancas          | Birstonietis g.    | 3                | 4                  | 193                | 40                |
--------------------------------------------------------------------------------------------------------------------------------
Rapanauskas       | Siauliu g.         | 6                | 5                  | 142                | 20                |
--------------------------------------------------------------------------------------------------------------------------------
Drapanauskas      | Gedimino kalnas    | 1                | 12                 | 251                | 12                |
--------------------------------------------------------------------------------------------------------------------------------
Solovjov          | Dubai city         | 2                | 10                 | 662                | 2                 |
--------------------------------------------------------------------------------------------------------------------------------
Juozapauskas      | Kazkurios g.       | 2                | 7                  | 193                | 12                |
--------------------------------------------------------------------------------------------------------------------------------
Svenciulis        | Klaipedos g.       | 1                | 12                 | 123                | 13                |
--------------------------------------------------------------------------------------------------------------------------------
Pazekius          | Kliumpiu g.        | 11               | 2                  | 123                | 2                 |
--------------------------------------------------------------------------------------------------------------------------------
```

```
Code      | Name             | Price |
-----------------------------------------
     123 | Klaipedos L       | 19.99 |
-----------------------------------------
     142 | Palangos Naujienos  |  25.5 |
-----------------------------------------
     193 | Kauno min         | 14.19 |
-----------------------------------------
     662 | Pasaulio Naujienos  |  40.2 |
-----------------------------------------
     251 | Kauno Herbas      | 30.99 |
-----------------------------------------

Highest income every month:
1 . |       Kauno Herbas|
2 . |       Kauno Herbas|
3 . |          Kauno min|
4 . |          Kauno min|
5 . |          Kauno min|
6 . |          Kauno min|
7 . |  Palangos Naujienos|
8 . |  Palangos Naujienos|
9 . |  Palangos Naujienos|
10. |  Palangos Naujienos|
11. |       Kauno Herbas|
12. |       Kauno Herbas|

All Publication income: 14477.32

Below average income publications
Code      | Name             | Price |
-----------------------------------------
     142 | Palangos Naujienos  |  25.5 |
-----------------------------------------
     662 | Pasaulio Naujienos  |  40.2 |
-----------------------------------------
```

Vartotojo sąsaja



Type in the publication's code: 123

Type in the month (1-12): 11

Klaipedos L Subscriber's last names:

Svenciulis
Pazekius

Input subscriber data:

Last Name	Adress	Subscribtion start	Subscribtion duration	Subscribtion code	Subscribtion amount
Zajancas	Birstonietis g.	3	4	193	40
Rapanauskas	Siauliu g.	6	5	142	20
Drapanauskas	Gedimino kalnas	1	12	251	12
Solovjov	Dubai city	2	10	662	2
Juozapauskas	Kazkurios g.	2	7	193	12
Svenciulis	Klaipedos g.	1	12	123	13
Pazekius	Kliumpiu g.	11	2	123	2

Input publication data:

Code	Name	Price
123	Klaipedos L	19.99
142	Palangos Naujienos	25.5
193	Kauno min	14.19
662	Pasaulio Naujienos	40.2
251	Kauno Herbas	30.99

3.8. Dėstytojo pastabos

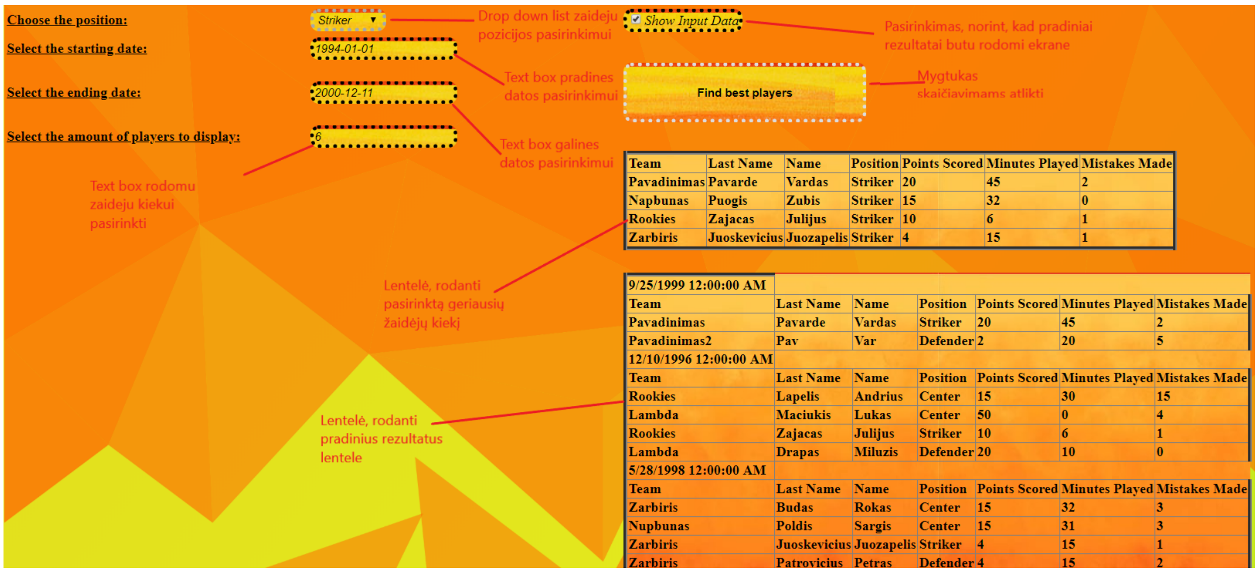
Testo rezultatas:1;

4. Kolekcijos ir išimčių valdymas (L4)

4.1. Darbo užduotis

LDD_4. Žaidėjai. Pirmojoje failo eilutėje nurodyta rungtynių data (failų daug). Tolesnėse eilutėse nurodyta komandos pavadinimas, krepšininko pavardė, vardas, žaistų minučių skaičius, pelnytų taškų skaičius, padarytų klaidų skaičius. Atskirame faile nurodyta komandos pavadinimas, krepšininko pavardė, vardas, žaidimo pozicija (puolėjas, gynėjas, centras). Sudarykite nurodytos pozicijos (įvedama klaviatūra) nurodytame periode (įvedama klaviatūra, datos nuo iki) naudingiausių žaidėjų nurodyto kiekio (įvedama klaviatūra) sąrašą. Naudingiausias žaidėjas tas, kuris pelnė daugiausiai taškų, žaidė mažiausiai minučių ir padarė mažiausiai klaidų.

4.2. Grafinės vartotojo sąsajos schema


Drop down list zaideju pozicijos pasirinkimui
Text box pradines datos pasirinkimui
Text box galines datos pasirinkimui
Text box rodomu zaideju kiekui pasirinkti
Show Input Data
Pasirinkimas, norint, kad pradiniai rezultatai butu rodomi ekrane
Find best players
Mygtukas skaičiavimams atlikti
Lentelė, rodanti pasirinktą geriausių žaidėjų kiekį
Lentelė, rodanti pradinius rezultatus lentele

Choose the position: Striker

Select the starting date: 1994-01-01

Select the ending date: 2000-12-11

Select the amount of players to display: 6

☑ Show Input Data

Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Pavadinimas	Pavarde	Vardas	Striker	20	45	2
Napbunas	Puogis	Zubis	Striker	15	32	0
Rookies	Zajacas	Julijus	Striker	10	6	1
Zarbiris	Juoskevicius	Juozapelis	Striker	4	15	1

9/25/1999 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Pavadinimas	Pavarde	Vardas	Striker	20	45	2
Pavadinimas2	Pav	Var	Defender	2	20	5
12/10/1996 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Rookies	Lapelis	Andrius	Center	15	30	15
Lambda	Maciukis	Lukas	Center	50	0	4
Rookies	Zajacas	Julijus	Striker	10	6	1
Lambda	Drapas	Miluzis	Defender	20	10	0
5/28/1998 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Zarbiris	Budas	Rokas	Center	15	32	3
Nupbunas	Poldis	Sargis	Center	15	31	3
Zarbiris	Juoskevicius	Juozapelis	Striker	4	15	1
Zarbiris	Patrovicius	Petras	Defender	4	15	2

4.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	**Savybė**	**Reikšmė**
Label1	Rodo tekstą	Nurodo, kad reikia pasirinkti poziciją
Label2	Rodo tekstą	Nurodo, kad reikia įvesti datą
Label3	Rodo tekstą	Nurodo, kad reikia įvesti datą
Label4	Rodo tekstą	Nurodo, kad reikia nurodyti žaidėjų kiekį
DropDownList1	Leidžia pasirinkti vieną iš pateiktų variantų	Pozicijos pasirinkimas
TextBox1	Teksto įvedimui	Datos įvedimas
TextBox2	Teksto įvedimui	Datos įvedimas
TextBox3	Teksto įvedimui	Žaidėjų kiekio pasirinkimas
CheckBox1	Pasrinkti taip arba ne	Pasirinkimas ar rodyti pradinius duomenis ar ne
Button1	Paspaudžiamas	Atlieka skaičiavimus
Table1	Rodyti informaciją lentele	Rodo rezultatus
Table2	Rodyti informaciją lentele	Rodo pradinius duomenis

4.4. Klasių diagrama

Forma.aspx.cs

#Page_Load(in sender :object, in e :EventArgs)
#Button1_Click(in sender :object, in e :EventArgs)
+ReadData(, in dictionary :List<Player>>)
+ReadMatchData(in file :string, in list :List<Player>,)
+ReadPosition(in file :string, in list :List<Player>)
+FindBestPlayers(in WishedPosition :string, in StartD :DateTime, in EndD :DateTime, , in list :List<Player>>, in MaxPlayers :integer) : List<Player>
+GetBestPlayer(in list :List<Player>, in position :string) : Player
+RemovePlayer(, in list :List<Player>>, in toBeRemoved :Player)
+PrintInputDataToFile(in file :string, , in list :List<Player>>)
+PrintAnswersToFile(in file :string, in list :List<Player>, in position :string)
+PrintInputDataToTable(, in list :List<Player>>)
+PrintAnswersToTable(in list :List<Player>)
+FileExceptionControl(in filePaths :string[], in PlayerInfo :string)
+MatchFileControl(in val :string[], in counter :integer, in file :string)
+PlayerInfoControl(in val :string[], in counter :integer, in file :string)
+UserInterfaceExceptions(in S :DateTime, in E :DateTime, in Amount :string, in WPos :string)
+InputDataShow()
+EmptyFileException(in file :string)

Player.cs
+Team : string +LastName : string +Name : string +MinutesPlayed : integer +PointsGained : integer +MistakesMade : integer +Position : string +Player(in team :string, in lastName :string, in name :string, in minutesPlayed :integer, in pointsGained :integer, in mistakesMade :integer) +Player(in team :string, in lastName :string, in name :string, in position :string) +SetPosition(in position :string) +CompareTo(in other :Player) : integer +Equals(in other :Player) : boolean +ToString() : string +Header() : string

4.5. Programos vartotojo vadovas

Vartotojui įvedus norimą poziciją, pradinę ir galinę rungtynių data ir norimą žaidėjų kiekį ir paspaudus skaičiavimų mygtuką, lentele atspausdinami geriausi pasirinktos pozicijos žaidėjai. Vartotojas gali pasirinkti, kad progama rodytų ir pradinius duomenis. Tai gali padaryti uždėjęs varnelę Check box dezuteje pries atliekant skaičiavimus.

4.6. Programos tekstas

Player.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace _4lab
{
```

```
public class Player: IComparable<Player>, IEquatable<Player>
{
    public string Team { get; set; }
    public string LastName { get; set; }
    public string Name { get; set; }
    public int MinutesPlayed { get; set; }
    public int PointsGained { get; set; }
    public int MistakesMade { get; set; }
    public string Position { get; set; }

    /// <summary>
    /// Player object constructor
    /// </summary>
    /// <param name="team"></param>
    /// <param name="lastName"></param>
    /// <param name="name"></param>
    /// <param name="minutesPlayed"></param>
    /// <param name="pointsGained"></param>
    /// <param name="mistakesMade"></param>
    public Player(string team, string lastName, string name, int minutesPlayed, int
          pointsGained, int mistakesMade)
    {
        Team = team;
        LastName = lastName;
        Name = name;
        MinutesPlayed = minutesPlayed;
        PointsGained = pointsGained;
        MistakesMade = mistakesMade;
    }
    /// <summary>
    /// playr object constructor
    /// </summary>
    /// <param name="team"></param>
    /// <param name="lastName"></param>
    /// <param name="name"></param>
    /// <param name="position"></param>
    public Player(string team, string lastName, string name, string position)
    {
        Team = team;
        LastName = lastName;
        Name = name;
        Position = position;
    }
    /// <summary>
    /// sets the postion from playerinfo file
    /// </summary>
    /// <param name="position"></param>
    public void SetPosition(string position)
    {
        Position = position;
    }
    /// <summary>
    /// compares players by points, minutes played and mistakes made
    /// </summary>
    /// <param name="other">other player</param>
    /// <returns>integer</returns>
    public int CompareTo(Player other)
    {
        if (this == null)
            return 0;
        if (other == null)
            return 1;
        if(PointsGained == other.PointsGained)
        {
            if (MinutesPlayed == other.MinutesPlayed)
                return other.MistakesMade.CompareTo(MistakesMade);
            return other.MinutesPlayed.CompareTo(MinutesPlayed);
        }
        return this.PointsGained.CompareTo(other.PointsGained);
    }
    /// <summary>
    /// compares two players
    /// </summary>
    /// <param name="other">otehr player</param>
    /// <returns>true or false</returns>
    public bool Equals(Player other)
    {
        if (other == null)
            return false;
        if (LastName == other.LastName && Name == other.Name && Team == other.Team)
            return true;
        return false;
    }
    /// <summary>
    /// Prinst out player information in a fromated string
    /// </summary>
    /// <returns>formated string </returns>
    public override string ToString()
    {
        return String.Format("{0, 15} | {1, 15} | {2, 15} | {3, 15} | {4, 15} | {5, 15} |
          {6, 15} |", Team, LastName, Name, Position, PointsGained, MinutesPlayed,
          MistakesMade);
    }
    /// <summary>
    /// prints the header of a table for the player object
    /// </summary>
    /// <returns>formated header</returns>
    public string Header()
    {
        return String.Format("{0, 15} | {1, 15} | {2, 15} | {3, 15} | {4, 15} | {5, 15} |
          {6, 15} |", "Team", "LastName", "Name", "Position", "PointsGained",
          "MinutesPlayed", "MistakesMade");
    }
}
}
```
Form.aspx.cs

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Form.aspx.cs" Inherits="_4lab.Form"
%>

<!DOCTYPE html>

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

        .auto-style1 {
            width: 100%;
        }
        .auto-style2 {
            width: 346px;
        }
        .auto-style3 {
            width: 346px;
            height: 33px;
        }
        .auto-style4 {
            height: 33px;
        }
        DarkMode{
            background-color: dimgray;
        }
        .auto-style6 {
```

```
    height: 33px;
    width: 357px;
}
.auto-style7 {
    width: 357px;
}
.darker {
    background-color: black;
}
.backgr{
    background-color:dimgray
}
.Background{
    background-image:url("Triangles.png");
    width:100vw;
    height:100vh;
    background-size:100%;
    margin-left:-7px;
    margin-top:-7px;
}
.Tables{
    background-image:url("TableB2.jpg");
    text-decoration:double;
    border-width: thick;
    font-weight:900;
    text-shadow:initial;
}
.Buttons{
    background-image:url("TableB.jpg");
    border-width: thick;
    border-style: dotted;
    width: 50%;
    height: 200%;
    cursor:pointer;
    border-radius:10px;
}
.TextBox{
    border-width:thick;
    border-style:dotted;
    background-image:url("TableB.jpg");
    font-style:oblique;
    cursor:text;
    border-radius:10px;
}
.DropDownL {
    border-width:thick;
    border-style:dotted;
    background-image:url("TableB.jpg");
    font-style:oblique;
    text-shadow:initial;
    cursor:pointer;
    border-radius:10px;
}
.LabelS{
    font-weight:800;
    font-size:16px;
    text-decoration:underline;
}
@-a-keyframes rainbow {
    0%{background-position:0% 82%}
    50%{background-position:100% 19%}
    100%{background-position:0% 82%}
}
@-b-keyframes rainbow {
    0%{background-position:0% 82%}
    50%{background-position:100% 19%}
```

```
                100%{background-position:0% 82%}
            }
            @-c-keyframes rainbow {
                0%{background-position:0% 82%}
                50%{background-position:100% 19%}
                100%{background-position:0% 82%}
            }
            @keyframes rainbow {
                0%{background-position:0% 82%}
                50%{background-position:100% 19%}
                100%{background-position:0% 82%}
            }
            .wzoom{
                            background: linear-gradient(124deg, #ff2400, #e81d1d, #e8b71d, #e3e81d,
                            #1de840, #1ddde8, #2b1de8, #dd00f3, #dd00f3);
                //background-image:url("Triangles.png");
                background-size: 1800% 1800%;

                -a-animation: rainbow 18s ease infinite;
                -b-animation: rainbow 18s ease infinite;
                -b-animation: rainbow 18s ease infinite;
                  animation: rainbow 18s ease infinite;
                  }
            </style>
    </head>
    <body class="wzoom" >
        <div class="Background">
        <form id="form1" runat="server">
            <div>
                <table class="auto-style1">
                    <tr>
                        <td class="auto-style3">
                            <asp:Label ID="Label1" runat="server" Text="Choose the position:"
                                CssClass="LabelS"></asp:Label>
                        </td>
                        <td class="auto-style6">
                            <asp:DropDownList ID="DropDownList1" runat="server"
                                CssClass="DropDownL">
                            </asp:DropDownList>
                        </td>
                        <td class="auto-style4">
                            <asp:CheckBox ID="CheckBox1" runat="server" CssClass="TextBox"
                                Text="Show Input Data" />
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            <asp:Label ID="Label2" runat="server" Text="Select the starting date:"
                                CssClass="LabelS"></asp:Label>
                        </td>
                        <td class="auto-style7">
                            <asp:TextBox ID="TextBox1" runat="server"
                                CssClass="TextBox"></asp:TextBox>
                            <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
                                runat="server" ControlToValidate="TextBox1" ErrorMessage="Starting
                                date field has to be filled"
                                ForeColor="Red">*</asp:RequiredFieldValidator>
                        </td>
                        <td>
                            &nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            <asp:Label ID="Label3" runat="server" Text="Select the ending date:"
                                CssClass="LabelS"></asp:Label>
                        </td>
                        <td class="auto-style7">
                            <asp:TextBox ID="TextBox2" runat="server"
                                CssClass="TextBox"></asp:TextBox>
                            <asp:RequiredFieldValidator ID="RequiredFieldValidator3"
                                runat="server" ControlToValidate="TextBox2" ErrorMessage="Starting
                                date field has to be filled"
                                ForeColor="Red">*</asp:RequiredFieldValidator>
                        </td>
                        <td>
                <asp:Button ID="Button1" runat="server" Text="Find best players"
                        OnClick="Button1_Click" CssClass="Buttons" Height="68px" Width="279px"
                        Font-Bold="True" />
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            <asp:Label ID="Label4" runat="server" Text="Select the amount of
                                players to display:" CssClass="LabelS"></asp:Label>
                        </td>
                        <td class="auto-style7">
                            <asp:TextBox ID="TextBox3" runat="server"
                                CssClass="TextBox"></asp:TextBox>
                            <asp:RequiredFieldValidator ID="RequiredFieldValidator4"
                                runat="server" ControlToValidate="TextBox3" ErrorMessage="Starting
                                date field has to be filled"
                                ForeColor="Red">*</asp:RequiredFieldValidator>
                        </td>
                        <td>
                            &nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            <asp:ValidationSummary ID="ValidationSummary1" runat="server"
                                ForeColor="Red" />
                        </td>
                        <td class="auto-style7">
                            &nbsp;</td>
                        <td>
                            <asp:Table ID="Table2" runat="server" GridLines="Both"
                                CssClass="Tables">
                            </asp:Table>
                        </td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            &nbsp;</td>
                        <td class="auto-style7">&nbsp;</td>
                        <td>
                            &nbsp;</td>
                    </tr>
                    <tr>
                        <td class="auto-style2">
                            &nbsp;</td>
                        <td class="auto-style7">
                            &nbsp;</td>
                        <td>
                            <asp:Table ID="Table1" runat="server" GridLines="Both"
                                CssClass="Tables" Height="19px">
                            </asp:Table>
                        </td>
                    </tr>
                </table>
                <br />
            </div>
        </form>
            </div>
```

```
</body>
</html>
```
Form.aspx
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.IO;

namespace _4lab
{
    public partial class Form : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (DropDownList1.Items.Count == 0)
            {
                DropDownList1.Items.Add("-");
                DropDownList1.Items.Add("Striker");
                DropDownList1.Items.Add("Defender");
                DropDownList1.Items.Add("Center");
            }
        }
        protected void Button1_Click(object sender, EventArgs e)
        {
            UserInterfaceExceptions(TextBox1.Text, TextBox2.Text, TextBox3.Text,
              DropDownList1.SelectedValue);
            string WantedPosition = DropDownList1.SelectedValue;
            DateTime StartD = DateTime.Parse(TextBox1.Text);
            DateTime EndD = DateTime.Parse(TextBox2.Text);
            int PAmount = int.Parse(TextBox3.Text);

            Dictionary<DateTime, List<Player>> MatchList = new Dictionary<DateTime,
              List<Player>>();
            ReadData(MatchList);
            PrintInputDataToFile("Answer.txt", MatchList);
            PrintInputDataToTable(MatchList);
            InputDataShow();

            List<Player> BestPlayers = FindBestPlayers(WantedPosition, StartD, EndD,
              MatchList, PAmount);
            PrintAnswersToFile("Answer.txt", BestPlayers, WantedPosition);
            PrintAnswersToTable(BestPlayers);
        }
        #region Read Data
        /// <summary>
        /// Reads all files in given location that start with the string Match
        /// </summary>
        /// <param name="dictionary">the list that needs to be filled</param>
        public void ReadData(Dictionary<DateTime, List<Player>> dictionary)
        {
            string[] filePaths = Directory.GetFiles(Server.MapPath(@"App_Data"),
              "Match*.txt");
            string PlayerInfo = Server.MapPath(@"App_Data/PlayerInfo.txt");
            FileExceptionControl(filePaths, PlayerInfo);
            foreach (string path in filePaths)
            {
                List<Player> list = new List<Player>();
                DateTime date;
                ReadMatchData(path, list, out date);
                ReadPosition(PlayerInfo, list);
                dictionary.Add(date, list);
            }
        }
        /// <summary>
        /// Reads individual match file
        /// </summary>
        /// <param name="file">given match file</param>
        /// <param name="list">given list</param>
        /// <param name="date">date from the first line of the file</param>
        public void ReadMatchData(string file, List<Player> list, out DateTime date)
        {
            using (StreamReader sr = new StreamReader(file))
            {
                int counter = 1;
                EmptyFileException(file);
                date = DateTime.Parse(sr.ReadLine());
                string line;
                while ((line = sr.ReadLine()) != null)
                {
                    string[] val = line.Split(';');
                    MatchFileControl(val, counter, file);
                    string TName = val[0];
                    string LName = val[1];
                    string Name = val[2];
                    int MPlayed = int.Parse(val[3]);
                    int PScored = int.Parse(val[4]);
                    int MMade = int.Parse(val[5]);
                    Player pl = new Player(TName, LName, Name, MPlayed, PScored, MMade);
                    if (!list.Contains(pl))
                        list.Add(pl);
                    counter++;
                }
            }
        }
        /// <summary>
        /// Reads the position file
        /// </summary>
        /// <param name="file">file location</param>
        /// <param name="list">player list</param>
        public void ReadPosition(string file, List<Player> list)
        {
            using (StreamReader sr = new StreamReader(file))
            {
                string line;
                int counter = 1;
                while ((line = sr.ReadLine()) != null)
                {
                    string[] val = line.Split(';');
                    PlayerInfoControl(val, counter, file);
                    Player temp = new Player(val[0], val[1], val[2], val[3]);
                    for (int i = 0; i < list.Count; i++)
                    {
                        if (list[i].Equals(temp))
                        {
                            list[i].SetPosition(temp.Position);
                            break;
                        }
                    }
                    counter++;
                }
            }
        }
        #endregion

        #region Get Best Players
        /// <summary>
        /// finds the best player from the list and deletes him after adding to new list
        /// </summary>
        /// <param name="WishedPosition">user chosen position</param>
        /// <param name="StartD">user chosen starting date</param>
        /// <param name="EndD">user chosen ending date</param>
        /// <param name="list">the list of players </param>
        /// <param name="MaxPlayers">The amount of players to be selected</param>
        /// <returns>list of best players</returns>
        public List<Player> FindBestPlayers(string WishedPosition, DateTime StartD, DateTime
              EndD, Dictionary<DateTime, List<Player>> list, int MaxPlayers)
        {
            List<Player> best = new List<Player>();
            for (int i = 0; i < MaxPlayers; i++)
            {
                Player BestPlayer = null;
                foreach (var entry in list)
                {
                    if (entry.Key >= StartD && entry.Key <= EndD)
                    {
                        Player BestListP = GetBestPlayer(entry.Value, WishedPosition);
                        if (BestListP != null && BestListP.CompareTo(BestPlayer) > 0)
                            BestPlayer = BestListP;
                    }
                }
                RemovePlayer(list, BestPlayer);
                if (BestPlayer != null)
                    best.Add(BestPlayer);
            }
            return best;
        }
        /// <summary>
        /// Gets the best player from the given list
        /// </summary>
        /// <param name="list">given list</param>
        /// <param name="position">user chosen position</param>
        /// <returns>best player from the list</returns>
        public Player GetBestPlayer(List<Player> list, string position)
        {
            Player BestP = null;
            for (int i = 0; i < list.Count; i++)
            {
                if (list[i].Position == position && list[i].CompareTo(BestP) > 0)
                    BestP = list[i];
            }
            return BestP;
        }
        /// <summary>
        /// Removes the chosen player from the list
        /// </summary>
        /// <param name="list">given list</param>
        /// <param name="toBeRemoved">player that is chosen to be removed</param>
        public void RemovePlayer(Dictionary<DateTime, List<Player>> list, Player toBeRemoved)
        {
            foreach (var entry in list)
            {
                entry.Value.Remove(toBeRemoved);
            }
        }
        #endregion

        #region Print Data
        /// <summary>
        /// Prints input data to answer file
        /// </summary>
        /// <param name="file">answer file location</param>
        /// <param name="list">list to be printed</param>
        public void PrintInputDataToFile(string file, Dictionary<DateTime, List<Player>> list)
        {
            using (StreamWriter sw = new StreamWriter(Server.MapPath(file)))
            {
                foreach (var entry in list)
                {
                    if (entry.Value.Count != 0)
                    {
                        sw.WriteLine(entry.Key);
                        sw.WriteLine(entry.Value[0].Header());
                        sw.WriteLine(new string('-', entry.Value[0].Header().Length));
                        for (int i = 0; i < entry.Value.Count; i++)
                            sw.WriteLine(entry.Value[i].ToString());
                        sw.WriteLine();
                    }
                    else
                    {
                        sw.WriteLine(entry.Key);
                        sw.WriteLine("List is empty");
                        sw.WriteLine();
                    }
                }
            }
        }
        /// <summary>
        /// Prints the answers to file
        /// </summary>
        /// <param name="file">file path</param>
        /// <param name="list">given list</param>
        /// <param name="position">user wished position</param>
        public void PrintAnswersToFile(string file, List<Player> list, string position)
        {
            using (StreamWriter sw = new StreamWriter(Server.MapPath(file), true))
            {
                if (list.Count != 0)
                {
                    sw.WriteLine("Chosen position by the user: {0}", position);
                    sw.WriteLine(list[0].Header());
                    sw.WriteLine(new string('-', list[0].Header().Length));
                    for (int i = 0; i < list.Count; i++)
                        sw.WriteLine(list[i].ToString());
                    sw.WriteLine();
                }
                else
                    sw.WriteLine("The list is empty");
            }
        }
        /// <summary>
        /// Prints input data to user interface
        /// </summary>
        /// <param name="list">given list</param>
        public void PrintInputDataToTable(Dictionary<DateTime, List<Player>> list)
        {
            foreach (var entry in list)
            {
                TableRow row = new TableRow();
                TableCell cella = new TableCell();
                cella.Text = entry.Key.ToString();
                row.Cells.Add(cella);
                Table1.Rows.Add(row);
                row = new TableRow();
                TableCell[] cell = new TableCell[7];
                for (int i = 0; i < 7; i++)
                    cell[i] = new TableCell();
                cell[0].Text = "Team";
                cell[1].Text = "Last Name";
                cell[2].Text = "Name";
                cell[3].Text = "Position";
                cell[4].Text = "Points Scored";
                cell[5].Text = "Minutes Played";
                cell[6].Text = "Mistakes Made";
                row.Cells.AddRange(cell);
                Table1.Rows.Add(row);
                for (int i = 0; i < entry.Value.Count; i++)
                {
                    row = new TableRow();
                    for (int j = 0; j < 7; j++)
                        cell[j] = new TableCell();
                    cell[0].Text = entry.Value[i].Team;
                    cell[1].Text = entry.Value[i].LastName;
                    cell[2].Text = entry.Value[i].Name;
                    cell[3].Text = entry.Value[i].Position;
                    cell[4].Text = entry.Value[i].PointsGained.ToString();
                    cell[5].Text = entry.Value[i].MinutesPlayed.ToString();
                    cell[6].Text = entry.Value[i].MistakesMade.ToString();
                    row.Cells.AddRange(cell);
                    Table1.Rows.Add(row);
                }
            }
        }
        /// <summary>
        /// Prints answers to user interface
        /// </summary>
        /// <param name="list">given list </param>
        public void PrintAnswersToTable(List<Player> list)
        {
            TableRow row = new TableRow();
            TableCell[] cell = new TableCell[7];
            for (int i = 0; i < 7; i++)
                cell[i] = new TableCell();
            cell[0].Text = "Team";
            cell[1].Text = "Last Name";
            cell[2].Text = "Name";
            cell[3].Text = "Position";
            cell[4].Text = "Points Scored";
            cell[5].Text = "Minutes Played";
            cell[6].Text = "Mistakes Made";
            row.Cells.AddRange(cell);
            Table2.Rows.Add(row);
            for (int i = 0; i < list.Count; i++)
            {
                row = new TableRow();
                for (int j = 0; j < 7; j++)
                    cell[j] = new TableCell();
                cell[0].Text = list[i].Team;
                cell[1].Text = list[i].LastName;
                cell[2].Text = list[i].Name;
                cell[3].Text = list[i].Position;
                cell[4].Text = list[i].PointsGained.ToString();
                cell[5].Text = list[i].MinutesPlayed.ToString();
                cell[6].Text = list[i].MistakesMade.ToString();
                row.Cells.AddRange(cell);
                Table2.Rows.Add(row);
            }
        }
        #endregion

        #region Exception Control
        /// <summary>
        /// Looks through file array, throws an exception if none are found
        /// </summary>
        /// <param name="filePaths">string array of files</param>
        /// <param name="PlayerInfo">player info file path</param>
        public void FileExceptionControl(string[] filePaths, string PlayerInfo)
```

```
{
    try
    {
        if (filePaths.Length == 0) throw new Exception("Cannot find any Match*.txt
             files in given location");
        else if (!File.Exists(PlayerInfo)) throw new Exception("Cannot find
             PlayerInfo.txt file in given location");
    }
    catch (Exception ex) { throw ex; }
}
/// <summary>
/// Looks for an error in the match files.
/// </summary>
/// <param name="val">player information</param>
/// <param name="counter">line counter</param>
/// <param name="file">file location</param>
public void MatchFileControl(string[] val, int counter, string file)
{
    try
    {
        if (val.Length != 6) throw new Exception(String.Format("There is a mistake in
             line {0} in the file {1}.", counter, file));
        for (int i = 3; i < 6; i++)
            if (int.TryParse(val[i], out int rez) == false)
                throw new Exception(string.Format("There is mistake in the line {0} in
             the file {1}. There should be a number instead of a char.", counter,
             file));
    }
    catch (Exception ex) { throw ex; }
}
/// <summary>
/// looks for mistakes in playerinfo file
/// </summary>
/// <param name="val">player information</param>
/// <param name="counter">line counter</param>
/// <param name="file">file path</param>
public void PlayerInfoControl(string[] val, int counter, string file)
{
    try
    {
        if (val.Length != 4) throw new Exception(String.Format("There is a mistake in
             line {0} in the file {1}.", counter, file));
    }
    catch (Exception ex) { throw ex; }
}
/// <summary>
/// User interface exception control
/// </summary>
/// <param name="S">user selected starting date</param>
/// <param name="E">user selected ending date</param>
/// <param name="Amount">user selected amount of players</param>
/// <param name="WPos">user selected wanted position</param>
public void UserInterfaceExceptions(string S, string E, string Amount, string WPos)
{
    try
    {
        if (DateTime.TryParse(S, out DateTime rez) == false)
            throw new Exception(string.Format("Starting date is in the incorrect
             format."));
        if (DateTime.TryParse(E, out DateTime rezz) == false)
            throw new Exception(string.Format("Ending date is in the incorrect
             format."));
        if (DateTime.Parse(E) < DateTime.Parse(S)) throw new
             Exception(string.Format("The starting date has to be before the ending
             date."));
        if (WPos != "Striker" && WPos != "Defender" && WPos != "Center")
```

```
                    throw new Exception(string.Format("The chosen position does not exist."));
                if (int.TryParse(Amount, out int rezzz) == false)
                    throw new Exception(string.Format("The chosen amount is not a number."));
            }
            catch (Exception ex) { throw ex; }
        }
        /// <summary>
        /// checkbox for input data
        /// </summary>
        public void InputDataShow()
        {
            if (CheckBox1.Checked)
                Table1.Visible = true;
            if (!CheckBox1.Checked)
                Table1.Visible = false;
        }
        /// <summary>
        /// empty file exception control
        /// </summary>
        /// <param name="file">file path</param>
        public void EmptyFileException(string file)
        {
            try
            {
                string[] a = File.ReadAllLines(file);
                if (a.Length == 0)
                    throw new Exception(string.Format("The File {0} is empty", file));
            }
            catch (Exception ex) { throw ex; }
        }
        #endregion
    }
}
```

4.7. Pradiniai duomenys ir rezultatai

Match1.txt:

```
1999-09-25
Pavadinimas;Pavarde;Vardas;45;20;2
Pavadinimas2;Pav;Var;20;2;5
```

Match2.txt

```
1996-12-10
Rookies;Lapelis;Andrius;30;15;15
Lambda;Maciukis;Lukas;0;50;4
Rookies;Zajacas;Julijus;6;10;1
Lambda;Drapas;Miluzis;10;20;0
```

Match3.txt

```
1998-05-28
Zarbiris;Budas;Rokas;32;15;3
Nupbunas;Poldis;Sargis;31;15;3
Zarbiris;Juoskevicius;Juozapelis;15;4;1
Zarbiris;Patrovicius;Petras;15;4;2
Napbunas;Puogis;Zubis;32;15;0
Napbunas;Laurius;Aurius;29;2;4
Zarbiris;Nuokalnis;Laurius;2;2;5
Napbunas;Pagalius;Zitas;31;15;0
```

Match4.txt

```
1998-04-12
```

PlayerInfo.txt

```
Pavadinimas;Pavarde;Vardas;Striker
Pavadinimas2;Pav;Var;Defender
Rookies;Lapelis;Andrius;Center
Lambda;Maciukis;Lukas;Center
```

```
Rookies;Zajacas;Julijus;Striker
Lambda;Drapas;Miluzis;Defender
Zarbiris;Budas;Rokas;Center
Nupbunas;Poldis;Sargis;Center
Zarbiris;Juoskevicius;Juozapelis;Striker
Zarbiris;Patrovicius;Petras;Defender
Napbunas;Puogis;Zubis;Striker
Napbunas;Laurius;Aurius;Center
Zarbiris;Nuokalnis;Laurius;Center
Napbunas;Pagalius;Zitas;Center
```

Answer.txt

```
9/25/1999 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Pavadinimas |         Pavarde |          Vardas |         Striker |              20 |              45 |               2 |
   Pavadinimas2 |             Pav |             Var |        Defender |               2 |              20 |               5 |

12/10/1996 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
        Rookies |         Lapelis |         Andrius |          Center |              15 |              30 |              15 |
         Lambda |        Maciukis |           Lukas |          Center |              50 |               0 |               4 |
        Rookies |         Zajacas |         Julijus |         Striker |              10 |               6 |               1 |
         Lambda |          Drapas |         Miluzis |        Defender |              20 |              10 |               0 |

5/28/1998 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
       Zarbiris |           Budas |           Rokas |          Center |              15 |              32 |               3 |
       Nupbunas |          Poldis |          Sargis |          Center |              15 |              31 |               3 |
       Zarbiris |    Juoskevicius |      Juozapelis |         Striker |               4 |              15 |               1 |
       Zarbiris |     Patrovicius |          Petras |        Defender |               4 |              15 |               2 |
       Napbunas |          Puogis |           Zubis |         Striker |              15 |              32 |               0 |
       Napbunas |         Laurius |          Aurius |          Center |               2 |              29 |               4 |
       Zarbiris |       Nuokalnis |         Laurius |          Center |               2 |               2 |               5 |
       Napbunas |        Pagalius |           Zitas |          Center |              15 |              31 |               0 |

4/12/1998 12:00:00 AM
List is empty

Chosen position by the user: Striker
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Pavadinimas |         Pavarde |          Vardas |         Striker |              20 |              45 |               2 |
       Napbunas |          Puogis |           Zubis |         Striker |              15 |              32 |               0 |
        Rookies |         Zajacas |         Julijus |         Striker |              10 |               6 |               1 |
       Zarbiris |    Juoskevicius |      Juozapelis |         Striker |               4 |              15 |               1 |
```

2 bandymas:

Match1.txt

```
1999-09-25
Pavadinimas;Pavarde;Vardas;45;20;2
Pavadinimas2;Pav;Var;20;2;5
```

Match2.txt

```
1996-12-10
Rookies;Lapelis;Andrius;30;15;15
Lambda;Maciukis;Lukas;20;31;4
Rookies;Zajacas;Julijus;6;10;1
Lambda;Drapas;Miluzis;10;20;0
```

Match3.txt

```
1998-05-28
Zarbiris;Budas;Rokas;32;15;3
Nupbunas;Poldis;Sargis;31;15;3
Zarbiris;Juoskevicius;Juozapelis;15;4;1
Zarbiris;Patrovicius;Petras;15;4;2
Napbunas;Puogis;Zubis;32;15;0
Napbunas;Laurius;Aurius;29;2;4
Zarbiris;Nuokalnis;Laurius;2;2;5
```

```
Napbunas;Pagalius;Zitas;31;15;0
```

Match4.txt

```
1998-04-12
Pomidorai;Kardas;Auksinis;29;15;3
Pomidorai;Princese;Auksuole;15;2;0
Agurkai;Giedriauskas;Lukis;0;0;0
Pomidorai;Faker;Drauguzis;5;2;4
Agurkai;Mandruolis;Sirijus;50;34;2
Agurkai;Poniulis;Dragunas;45;34;1
Pomidorai;Zykiukas;Sauliukas;45;34;0
```

Match5.txt

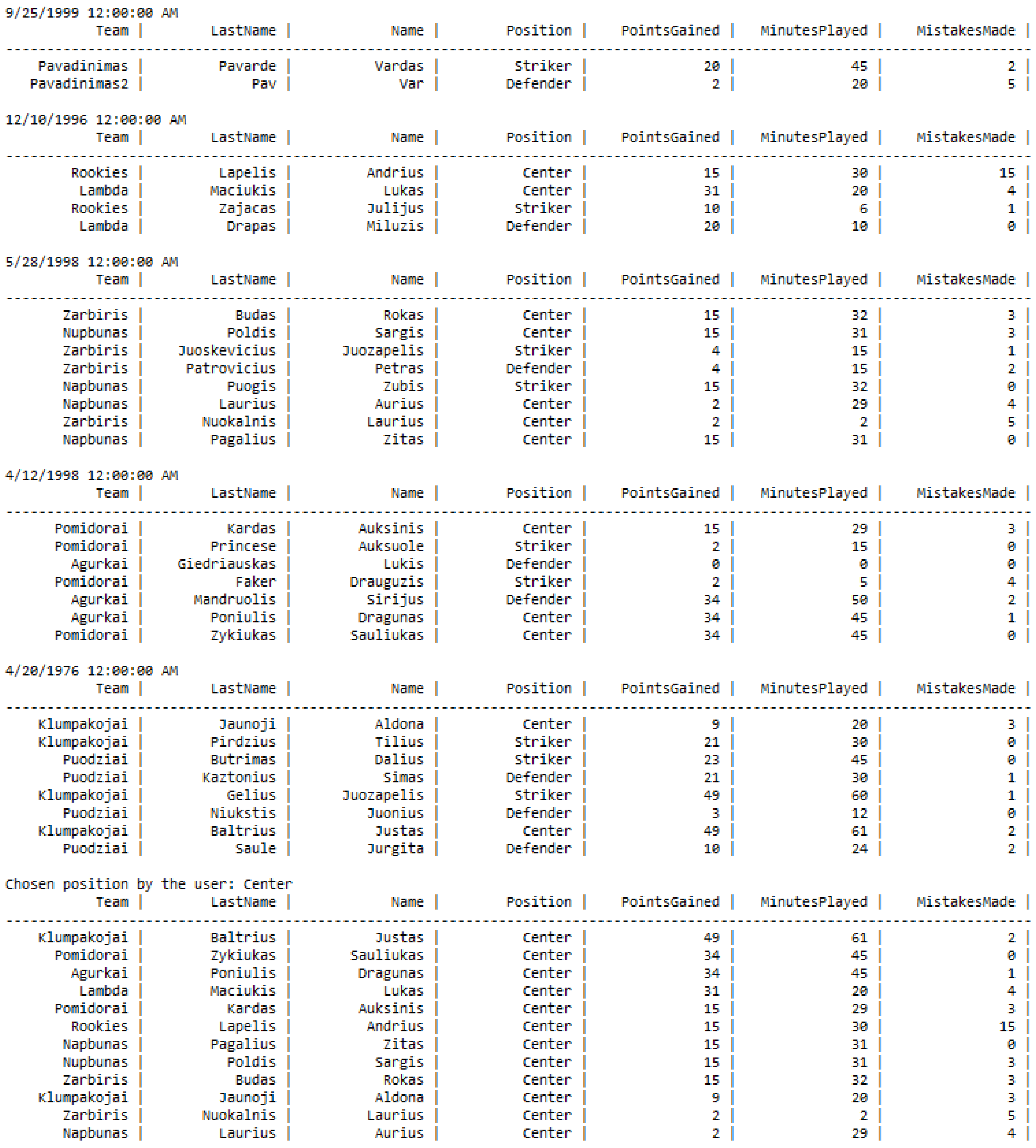
```
1976-04-20
Klumpakojai;Jaunoji;Aldona;20;9;3
Klumpakojai;Pirdzius;Tilius;30;21;0
Puodziai;Butrimas;Dalius;45;23;0
Puodziai;Kaztonius;Simas;30;21;1
Klumpakojai;Gelius;Juozapelis;60;49;1
Puodziai;Niukstis;Juonius;12;3;0
Klumpakojai;Baltrius;Justas;61;49;2
Puodziai;Saule;Jurgita;24;10;2
```

PlayerInfo.txt

```
Pavadinimas;Pavarde;Vardas;Striker
Pavadinimas2;Pav;Var;Defender
Rookies;Lapelis;Andrius;Center
Lambda;Maciukis;Lukas;Center
Rookies;Zajacas;Julijus;Striker
Lambda;Drapas;Miluzis;Defender
Zarbiris;Budas;Rokas;Center
Nupbunas;Poldis;Sargis;Center
Zarbiris;Juoskevicius;Juozapelis;Striker
Zarbiris;Patrovicius;Petras;Defender
Napbunas;Puogis;Zubis;Striker
Napbunas;Laurius;Aurius;Center
Zarbiris;Nuokalnis;Laurius;Center
Napbunas;Pagalius;Zitas;Center
Pomidorai;Kardas;Auksinis;Center
Pomidorai;Princese;Auksuole;Striker
Agurkai;Giedriauskas;Lukis;Defender
Pomidorai;Faker;Drauguzis;Striker
Agurkai;Mandruolis;Sirijus;Defender
Agurkai;Poniulis;Dragunas;Center
Pomidorai;Zykiukas;Sauliukas;Center
Klumpakojai;Jaunoji;Aldona;Center
Klumpakojai;Pirdzius;Tilius;Striker
Puodziai;Butrimas;Dalius;Striker
Puodziai;Kaztonius;Simas;Defender
Klumpakojai;Gelius;Juozapelis;Striker
Puodziai;Niukstis;Juonius;Defender
Klumpakojai;Baltrius;Justas;Center
Puodziai;Saule;Jurgita;Defender
```

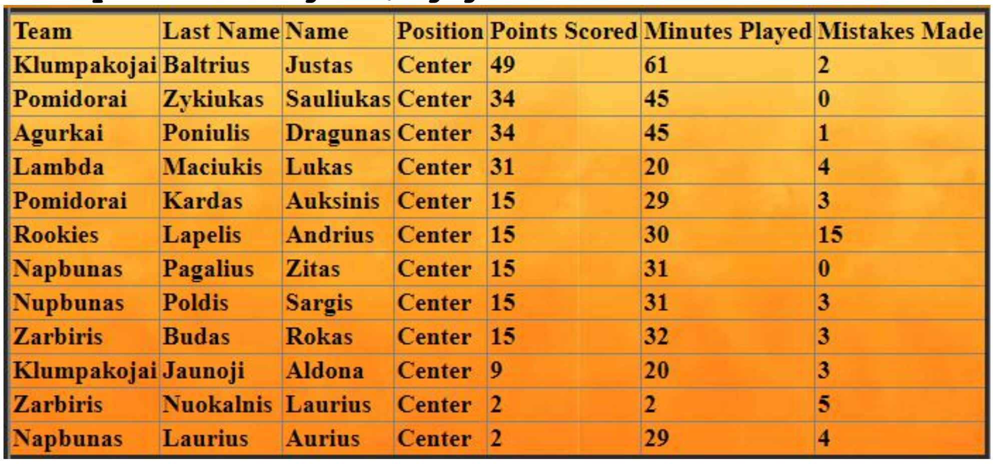
Pasirinkus pradinę data 1960-01-01, galinę 2000-12-11, poziciją: Center, o rodomų žaidėjų skaičių 20, gaunami tokie atsakymai:

Answer.txt



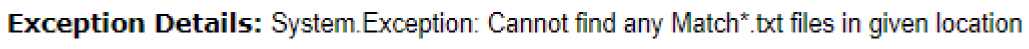
```
9/25/1999 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Pavadinimas |         Pavarde |          Vardas |         Striker |              20 |              45 |               2 |
   Pavadinimas2 |             Pav |             Var |        Defender |               2 |              20 |               5 |

12/10/1996 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
        Rookies |         Lapelis |         Andrius |          Center |              15 |              30 |              15 |
         Lambda |        Maciukis |           Lukas |          Center |              31 |              20 |               4 |
        Rookies |         Zajacas |         Julijus |         Striker |              10 |               6 |               1 |
         Lambda |          Drapas |         Miluzis |        Defender |              20 |              10 |               0 |

5/28/1998 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
       Zarbiris |           Budas |           Rokas |          Center |              15 |              32 |               3 |
       Nupbunas |          Poldis |          Sargis |          Center |              15 |              31 |               3 |
       Zarbiris |    Juoskevicius |      Juozapelis |         Striker |               4 |              15 |               1 |
       Zarbiris |     Patrovicius |          Petras |        Defender |               4 |              15 |               2 |
       Napbunas |          Puogis |           Zubis |         Striker |              15 |              32 |               0 |
       Napbunas |         Laurius |          Aurius |          Center |               2 |              29 |               4 |
       Zarbiris |       Nuokalnis |         Laurius |          Center |               2 |               2 |               5 |
       Napbunas |        Pagalius |           Zitas |          Center |              15 |              31 |               0 |

4/12/1998 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
      Pomidorai |          Kardas |        Auksinis |          Center |              15 |              29 |               3 |
      Pomidorai |        Princese |        Auksuole |         Striker |               2 |              15 |               0 |
        Agurkai |    Giedriauskas |           Lukis |        Defender |               0 |               0 |               0 |
      Pomidorai |           Faker |       Drauguzis |         Striker |               2 |               5 |               4 |
        Agurkai |      Mandruolis |         Sirijus |        Defender |              34 |              50 |               2 |
        Agurkai |        Poniulis |        Dragunas |          Center |              34 |              45 |               1 |
      Pomidorai |        Zykiukas |       Sauliukas |          Center |              34 |              45 |               0 |

4/20/1976 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Klumpakojai |         Jaunoji |          Aldona |          Center |               9 |              20 |               3 |
    Klumpakojai |        Pirdzius |          Tilius |         Striker |              21 |              30 |               0 |
       Puodziai |        Butrimas |          Dalius |         Striker |              23 |              45 |               0 |
       Puodziai |       Kaztonius |           Simas |        Defender |              21 |              30 |               1 |
    Klumpakojai |          Gelius |      Juozapelis |         Striker |              49 |              60 |               1 |
       Puodziai |        Niukstis |         Juonius |        Defender |               3 |              12 |               0 |
    Klumpakojai |        Baltrius |          Justas |          Center |              49 |              61 |               2 |
       Puodziai |           Saule |         Jurgita |        Defender |              10 |              24 |               2 |

Chosen position by the user: Center
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Klumpakojai |        Baltrius |          Justas |          Center |              49 |              61 |               2 |
      Pomidorai |        Zykiukas |       Sauliukas |          Center |              34 |              45 |               0 |
        Agurkai |        Poniulis |        Dragunas |          Center |              34 |              45 |               1 |
         Lambda |        Maciukis |           Lukas |          Center |              31 |              20 |               4 |
      Pomidorai |          Kardas |        Auksinis |          Center |              15 |              29 |               3 |
        Rookies |         Lapelis |         Andrius |          Center |              15 |              30 |              15 |
       Napbunas |        Pagalius |           Zitas |          Center |              15 |              31 |               0 |
       Nupbunas |          Poldis |          Sargis |          Center |              15 |              31 |               3 |
       Zarbiris |           Budas |           Rokas |          Center |              15 |              32 |               3 |
    Klumpakojai |         Jaunoji |          Aldona |          Center |               9 |              20 |               3 |
       Zarbiris |       Nuokalnis |         Laurius |          Center |               2 |               2 |               5 |
       Napbunas |         Laurius |          Aurius |          Center |               2 |              29 |               4 |
```

Atsakymai vartotojo sąsajoje:



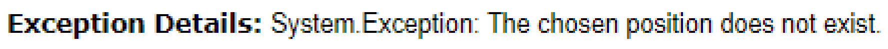
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Klumpakojai	Baltrius	Justas	Center	49	61	2
Pomidorai	Zykiukas	Sauliukas	Center	34	45	0
Agurkai	Poniulis	Dragunas	Center	34	45	1
Lambda	Maciukis	Lukas	Center	31	20	4
Pomidorai	Kardas	Auksinis	Center	15	29	3
Rookies	Lapelis	Andrius	Center	15	30	15
Napbunas	Pagalius	Zitas	Center	15	31	0
Nupbunas	Poldis	Sargis	Center	15	31	3
Zarbiris	Budas	Rokas	Center	15	32	3
Klumpakojai	Jaunoji	Aldona	Center	9	20	3
Zarbiris	Nuokalnis	Laurius	Center	2	2	5
Napbunas	Laurius	Aurius	Center	2	29	4

3 bandymas:
Šiame bandyme programai neduosime Match*.txt failų. Gausime tokį exception:

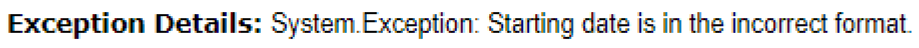

Exception Details: System.Exception: Cannot find any Match*.txt files in given location

4 bandymas:
Dabar vartotojo sąsajoje įrašyisime netinkamus duomenis:

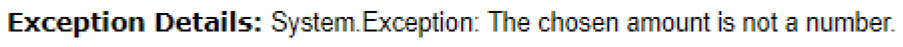
Nepasirinkę pozicijos, gauname tokį exception:


Exception Details: System.Exception: The chosen position does not exist.

Įrašę netinkamą datą, gauname tokį exception:


Exception Details: System.Exception: Starting date is in the incorrect format.

Į player amound lauką įrašę ne skaičių, guname tokį exception:


Exception Details: System.Exception: The chosen amount is not a number.

4.8. Dėstytojo pastabos

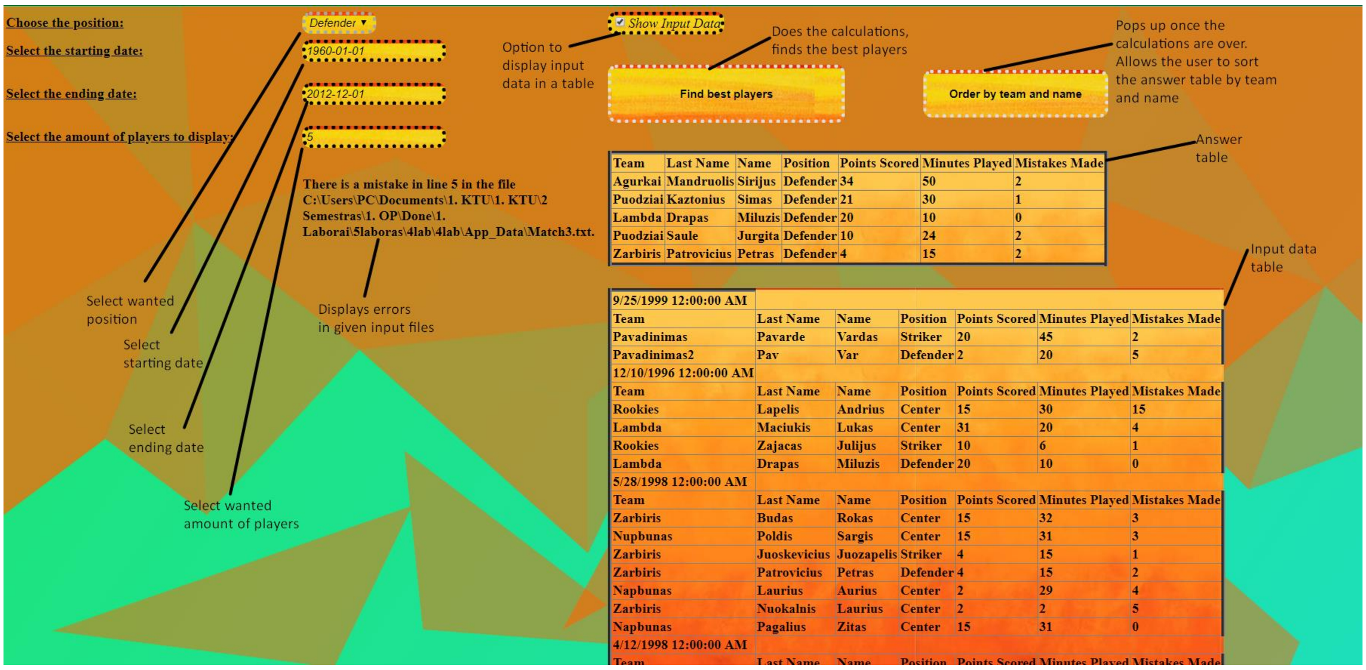
Testo rezultatas: 0;

5. Deklaratyvusis programavimas (L5)

5.1. *Darbo užduotis*

LDD_5. Žaidėjai. Pirmojoje failo eilutėje nurodyta rungtynių data (failų daug). Tolesnėse eilutėse nurodyta komandos pavadinimas, krepšininko pavardė, vardas, žaistų minučių skaičius, pelnytų taškų skaičius, padarytų klaidų skaičius. Atskirame faile nurodyta komandos pavadinimas, krepšininko pavardė, vardas, žaidimo pozicija (puolėjas, gynėjas, centras). Sudarykite nurodytos pozicijos (įvedama klaviatūra) nurodytame periode (įvedama klaviatūra, datos nuo iki) naudingiausių žaidėjų nurodyto kiekio (įvedama klaviatūra) sąrašą. Naudingiausias žaidėjas tas, kuris pelnė daugiausiai taškų, žaidė mažiausiai minučių ir padarė mažiausiai klaidų. Rikiuoti (komanda, krepšininko pavardė).

5.2. *Grafinės vartotojo sąsajos schema*


Choose the position:
Defender
Select the starting date:
1960-01-01
Select the ending date:
2012-12-01
Select the amount of players to display:
5
Show Input Data
Find best players
Order by team and name
Option to display input data in a table
Does the calculations, finds the best players
Pops up once the calculations are over. Allows the user to sort the answer table by team and name
Answer table
Input data table
Select wanted position
Select starting date
Select ending date
Select wanted amount of players
Displays errors in given input files

There is a mistake in line 5 in the file C:\Users\PC\Documents\1. KTU\1. KTU\2 Semestras\1. OP\Done\1. Laborai\5laboras\4lab\4lab\App_Data\Match3.txt.

Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Agurkai	Mandruolis	Sirijus	Defender	34	50	2
Puodziai	Kaztonius	Simas	Defender	21	30	1
Lambda	Drapas	Miluzis	Defender	20	10	0
Puodziai	Saule	Jurgita	Defender	10	24	2
Zarbiris	Patrovicius	Petras	Defender	4	15	2

9/25/1999 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Pavadinimas	Pavarde	Vardas	Striker	20	45	2
Pavadinimas2	Pav	Var	Defender	2	20	5
12/10/1996 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Rookies	Lapelis	Andrius	Center	15	30	15
Lambda	Maciukis	Lukas	Center	31	20	4
Rookies	Zajacas	Julijus	Striker	10	6	1
Lambda	Drapas	Miluzis	Defender	20	10	0
5/28/1998 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Zarbiris	Budas	Rokas	Center	15	32	3
Nupbunas	Poldis	Sargis	Center	15	31	3
Zarbiris	Juoskevicius	Juozapelis	Striker	4	15	1
Zarbiris	Patrovicius	Petras	Defender	4	15	2
Napbunas	Laurius	Aurius	Center	2	29	4
Zarbiris	Nuokalnis	Laurius	Center	2	2	5
Napbunas	Pagalius	Zitas	Center	15	31	0
4/12/1998 12:00:00 AM						
Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made

5.3. *Sąsajoje panaudotų komponentų keičiamos savybės*

Komponentas	**Savybė**	**Reikšmė**
Label1	Rodo tekstą	Nurodo, kad reikia pasirinkti poziciją
Label2	Rodo tekstą	Nurodo, kad reikia įvesti datą
Label3	Rodo tekstą	Nurodo, kad reikia įvesti datą
Label4	Rodo tekstą	Nurodo, kad reikia nurodyti žaidėjų kiekį
Label5	Rodo tekstą	Nurodo klaidas failuose
DropDownList1	Leidžia pasirinkti vieną iš pateiktų variantų	Pozicijos pasirinkimas
TextBox1	Teksto įvedimui	Datos įvedimas
TextBox2	Teksto įvedimui	Datos įvedimas
TextBox3	Teksto įvedimui	Žaidėjų kiekio pasirinkimas
CheckBox1	Pasrinkti taip arba ne	Pasirinkimas ar rodyti pradinius duomenis ar ne
Button1	Paspaudžiamas	Atlieka skaičiavimus
Button2	Paspaudžiamas	Rikiuoja atsakymus pagal komandą ir vardą
Table1	Rodyti informaciją lentele	Rodo rezultatus
Table2	Rodyti informaciją lentele	Rodo pradinius duomenis

5.4. Klasių diagrama

Forma.aspx.cs
-BestPlayers : List<Player>
-List<Player>> : Dictionary<DateTime,
#Page_Load(in sender :object, in e :EventArgs)
#Button1_Click(in sender :object, in e :EventArgs)
#Button2_Click(in sender :object, in e :EventArgs)
+ReadData(, in dictionary :List<Player>>)
+ReadMatchData(in file :string, in list :List<Player>,)
+ReadPosition(in file :string, in list :List<Player>)
+FindBestPlayers(in WishedPosition :string, in StartD :DateTime, in EndD :DateTime, , in list :List<Player>>, in MaxPlayers :integer) : List<Player>
+PrintInputDataToFile(in file :string, , in list :List<Player>>)
+PrintAnswersToFile(in file :string, in list :List<Player>, in position :string)
+PrintInputDataToTable(, in list :List<Player>>)
+PrintAnswersToTable(in list :List<Player>)
+FileExceptionControl(in filePaths :string[], in PlayerInfo :string)
+MatchFileControl(in val :string[], in counter :integer, in file :string) : boolean
+PlayerInfoControl(in val :string[], in counter :integer, in file :string) : boolean
+UserInterfaceExceptions(in S :string, in E :string, in Amount :string, in WPos :string)
+InputDataShow()
+EmptyFileException(in file :string)

Player.cs
+Team : string
+LastName : string
+Name : string
+MinutesPlayed : integer
+PointsGained : integer
+MistakesMade : integer
+Position : string
+Player(in team :string, in lastName :string, in name :string, in minutesPlayed :integer, in pointsGained :integer, in mistakesMade :integer)
+Player(in team :string, in lastName :string, in name :string, in position :string)
+SetPosition(in position :string)
+CompareTo(in other :Player) : integer
+Equals(in other :Player) : boolean
+ToString() : string
+Header() : string

5.5. Programos vartotojo vadovas

Vartotojui įvedus norimą poziciją, pradinę ir galinę rungtynių datas ir norimą žaidėjų kiekį ir paspaudus skaičiavimų mygtuką, lentele atspausdinami geriausi pasirinktos pozicijos žaidėjai. Vartotojas gali pasirinkti, kad progama rodytų ir pradinius duomenis. Tai gali padaryti uždėjęs varnelę Check box dezuteje pries atliekant skaičiavimus. Programai atlikus skaičiavimus atsiranda dar vienas mygtukas. Jį paspaudus programa išrikiuoja geriausių žaidėjų lentelę pagal komandos pavadinimą ir žaidėjų pavardes. Failuose esant klaidoms, programa praleidžia klaidingas eilutes, atlikdama skaičiavimus su kitomis eilutėmis. Vartotojas yra pranešamas apie klaidingas eilutes vartotojo sąsajoje.

5.6. Programos tekstas

Player.cs
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace _4lab
{
    public class Player: IComparable<Player>, IEquatable<Player>
    {
        public string Team { get; set; }
        public string LastName { get; set; }
        public string Name { get; set; }
        public int MinutesPlayed { get; set; }
        public int PointsGained { get; set; }
        public int MistakesMade { get; set; }
        public string Position { get; set; }

        /// <summary>
        /// Player object constructor
        /// </summary>
        /// <param name="team"></param>
        /// <param name="lastName"></param>
        /// <param name="name"></param>
        /// <param name="minutesPlayed"></param>
        /// <param name="pointsGained"></param>
        /// <param name="mistakesMade"></param>
        public Player(string team, string lastName, string name, int minutesPlayed, int
              pointsGained, int mistakesMade)
        {
            Team = team;
            LastName = lastName;
            Name = name;
            MinutesPlayed = minutesPlayed;
            PointsGained = pointsGained;
            MistakesMade = mistakesMade;
        }
        /// <summary>
        /// playr object constructor
        /// </summary>
        /// <param name="team"></param>
        /// <param name="lastName"></param>
        /// <param name="name"></param>
        /// <param name="position"></param>
        public Player(string team, string lastName, string name, string position)
        {
            Team = team;
            LastName = lastName;
            Name = name;
            Position = position;
        }
        /// <summary>
        /// sets the postion from playerinfo file
        /// </summary>
        /// <param name="position"></param>
        public void SetPosition(string position)
        {
            Position = position;
        }
        /// <summary>
        /// compares players by points, minutes played and mistakes made
        /// </summary>
        /// <param name="other">other player</param>
        /// <returns>integer</returns>
        public int CompareTo(Player other)
        {
            if (this == null)
                return 0;
            if (other == null)
                return 1;
            if(PointsGained == other.PointsGained)
            {
                if (MinutesPlayed == other.MinutesPlayed)
                    return other.MistakesMade.CompareTo(MistakesMade);
                return other.MinutesPlayed.CompareTo(MinutesPlayed);
            }
            return this.PointsGained.CompareTo(other.PointsGained);
        }
        /// <summary>
        /// compares two players
        /// </summary>
        /// <param name="other">otehr player</param>
        /// <returns>true or false</returns>
        public bool Equals(Player other)
        {
            if (other == null)
                return false;
            if (LastName == other.LastName && Name == other.Name && Team == other.Team)
                return true;
            return false;
        }
        /// <summary>
        /// Prinst out player information in a fromated string
        /// </summary>
        /// <returns>formated string </returns>
        public override string ToString()
        {
            return String.Format("{0, 15} | {1, 15} | {2, 15} | {3, 15} | {4, 15} | {5, 15} |
              {6, 15} |", Team, LastName, Name, Position, PointsGained, MinutesPlayed,
              MistakesMade);
        }
        /// <summary>
        /// prints the header of a table for the player object
        /// </summary>
        /// <returns>formated header</returns>
        public string Header()
        {
            return String.Format("{0, 15} | {1, 15} | {2, 15} | {3, 15} | {4, 15} | {5, 15} |
              {6, 15} |", "Team", "LastName", "Name", "Position", "PointsGained",
              "MinutesPlayed", "MistakesMade");
        }
    }
       }
```

Forma.aspx.cs

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

namespace _4lab
{
    public partial class Form : System.Web.UI.Page
    {
        private List<Player> BestPlayers;
        private Dictionary<DateTime, List<Player>> MatchList;
        protected void Page_Load(object sender, EventArgs e)
        {
            BestPlayers = (List<Player>)Session["Best"];
            MatchList = (Dictionary<DateTime, List<Player>>)Session["Input"];
            if (DropDownList1.Items.Count == 0)
            {
```

```
            DropDownList1.Items.Add("-");
            DropDownList1.Items.Add("Striker");
            DropDownList1.Items.Add("Defender");
            DropDownList1.Items.Add("Center");
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        UserInterfaceExceptions(TextBox1.Text, TextBox2.Text, TextBox3.Text,
          DropDownList1.SelectedValue);
        string WantedPosition = DropDownList1.SelectedValue;
        DateTime StartD = DateTime.Parse(TextBox1.Text);
        DateTime EndD = DateTime.Parse(TextBox2.Text);
        int PAmount = int.Parse(TextBox3.Text);

        Dictionary<DateTime, List<Player>> MatchList = new Dictionary<DateTime,
                 List<Player>>();
        ReadData(MatchList);
        PrintInputDataToFile("Answer.txt", MatchList);
        PrintInputDataToTable(MatchList);
        InputDataShow();

        List<Player> BestPlayers = FindBestPlayers(WantedPosition, StartD, EndD,
          MatchList, PAmount);
        PrintAnswersToFile("Answer.txt", BestPlayers, WantedPosition);
        PrintAnswersToTable(BestPlayers);

        Session["Best"] = BestPlayers;
        Session["Input"] = MatchList;
        Button2.Visible = true;
    }
    /// <summary>
    /// Second button for sorting
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void Button2_Click(object sender, EventArgs e)
    {
        BestPlayers = BestPlayers.OrderBy(x => x.Team).ThenBy(x => x.LastName).ToList();
        PrintAnswersToTable(BestPlayers);
        if (CheckBox1.Checked == true)
            PrintInputDataToTable(MatchList);
    }
    #region Read Data
    /// <summary>
    /// Reads all files in given location that start with the string Match
    /// </summary>
    /// <param name="dictionary">the list that needs to be filled</param>
    public void ReadData(Dictionary<DateTime, List<Player>> dictionary)
    {
        string[] filePaths = Directory.GetFiles(Server.MapPath(@"App_Data"),
          "Match*.txt");
        string PlayerInfo = Server.MapPath(@"App_Data/PlayerInfo.txt");
        FileExceptionControl(filePaths, PlayerInfo);
        foreach (string path in filePaths)
        {
            List<Player> list = new List<Player>();
            DateTime date;
            ReadMatchData(path, list, out date);
            ReadPosition(PlayerInfo, list);
            dictionary.Add(date, list);
        }

    }
    /// <summary>
    /// Reads individual match file
```

```
/// </summary>
/// <param name="file">given match file</param>
/// <param name="list">given list</param>
/// <param name="date">date from the first line of the file</param>
public void ReadMatchData(string file, List<Player> list, out DateTime date)
{
    using (StreamReader sr = new StreamReader(file))
    {
        int counter = 1;
        EmptyFileException(file);
        date = DateTime.Parse(sr.ReadLine());
        string line;
        while ((line = sr.ReadLine()) != null)
        {
            string[] val = line.Split(';');
            if (MatchFileControl(val, counter, file))
                continue;
            string TName = val[0];
            string LName = val[1];
            string Name = val[2];
            int MPlayed = int.Parse(val[3]);
            int PScored = int.Parse(val[4]);
            int MMade = int.Parse(val[5]);
            Player pl = new Player(TName, LName, Name, MPlayed, PScored, MMade);
            if (!list.Contains(pl))
                list.Add(pl);
            counter++;
        }
    }
}
/// <summary>
/// Reads the position file
/// </summary>
/// <param name="file">file location</param>
/// <param name="list">player list</param>
public void ReadPosition(string file, List<Player> list)
{
    using (StreamReader sr = new StreamReader(file))
    {
        string line;
        int counter = 1;
        while ((line = sr.ReadLine()) != null)
        {
            string[] val = line.Split(';');
            if (PlayerInfoControl(val, counter, file))
                continue;

            var a = list.Find(x => x.Team == val[0] && x.LastName == val[1] && x.Name
             == val[2]);
            if (a != null) a.SetPosition(val[3]);
            counter++;
        }
    }
}
#endregion

#region Get Best Players
/// <summary>
/// finds the best player from the list and deletes him after adding to new list
/// </summary>
/// <param name="WishedPosition">user chosen position</param>
/// <param name="StartD">user chosen starting date</param>
/// <param name="EndD">user chosen ending date</param>
/// <param name="list">the list of players </param>
/// <param name="MaxPlayers">The amount of players to be selected</param>
/// <returns>list of best players</returns>
```

```
public List<Player> FindBestPlayers(string WishedPosition, DateTime StartD, DateTime
      EndD, Dictionary<DateTime, List<Player>> list, int MaxPlayers)
{
    return list.Where(nn => nn.Key >= StartD && nn.Key <= EndD).SelectMany(nn =>
      nn.Value)
        .Where(nn => nn.Position == WishedPosition).ToList().OrderByDescending(nn =>
             nn.PointsGained)
        .ThenBy(nn => nn.MinutesPlayed).ThenBy(nn =>
      nn.MistakesMade).Take(MaxPlayers).ToList();
}
#endregion

#region Print Data
/// <summary>
/// Prints input data to answer file
/// </summary>
/// <param name="file">answer file location</param>
/// <param name="list">list to be printed</param>
public void PrintInputDataToFile(string file, Dictionary<DateTime, List<Player>> list)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(file)))
    {
        foreach (var entry in list)
        {
            if (entry.Value.Count != 0)
            {
                sw.WriteLine(entry.Key);
                sw.WriteLine(entry.Value[0].Header());
                sw.WriteLine(new string('-', entry.Value[0].Header().Length));
                entry.Value.ForEach(aa => { sw.WriteLine(aa.ToString()); });
                sw.WriteLine();
            }
            else
            {
                sw.WriteLine(entry.Key);
                sw.WriteLine("List is empty");
                sw.WriteLine();
            }
        }
    }
}
/// <summary>
/// Prints the answers to file
/// </summary>
/// <param name="file">file path</param>
/// <param name="list">given list</param>
/// <param name="position">user wished position</param>
public void PrintAnswersToFile(string file, List<Player> list, string position)
{
    using (StreamWriter sw = new StreamWriter(Server.MapPath(file), true))
    {
        if (list.Count != 0)
        {
            sw.WriteLine("Chosen position by the user: {0}", position);
            sw.WriteLine(list[0].Header());
            sw.WriteLine(new string('-', list[0].Header().Length));
            for (int i = 0; i < list.Count; i++)
                sw.WriteLine(list[i].ToString());
            sw.WriteLine();
        }
        else
            sw.WriteLine("The list is empty");
    }
}
/// <summary>
/// Prints input data to user interface
/// </summary>
/// <param name="list">given list</param>
public void PrintInputDataToTable(Dictionary<DateTime, List<Player>> list)
{
    foreach (var entry in list)
    {
        TableRow row = new TableRow();
        TableCell cella = new TableCell();
        cella.Text = entry.Key.ToString();
        row.Cells.Add(cella);
        Table1.Rows.Add(row);
        row = new TableRow();
        TableCell[] cell = new TableCell[7];
        for (int i = 0; i < 7; i++)
            cell[i] = new TableCell();
        cell[0].Text = "Team";
        cell[1].Text = "Last Name";
        cell[2].Text = "Name";
        cell[3].Text = "Position";
        cell[4].Text = "Points Scored";
        cell[5].Text = "Minutes Played";
        cell[6].Text = "Mistakes Made";
        row.Cells.AddRange(cell);
        Table1.Rows.Add(row);
        for (int i = 0; i < entry.Value.Count; i++)
        {
            row = new TableRow();
            for (int j = 0; j < 7; j++)
                cell[j] = new TableCell();
            cell[0].Text = entry.Value[i].Team;
            cell[1].Text = entry.Value[i].LastName;
            cell[2].Text = entry.Value[i].Name;
            cell[3].Text = entry.Value[i].Position;
            cell[4].Text = entry.Value[i].PointsGained.ToString();
            cell[5].Text = entry.Value[i].MinutesPlayed.ToString();
            cell[6].Text = entry.Value[i].MistakesMade.ToString();
            row.Cells.AddRange(cell);
            Table1.Rows.Add(row);
        }
    }
}
/// <summary>
/// Prints answers to user interface
/// </summary>
/// <param name="list">given list </param>
public void PrintAnswersToTable(List<Player> list)
{
    TableRow row = new TableRow();
    TableCell[] cell = new TableCell[7];
    for (int i = 0; i < 7; i++)
        cell[i] = new TableCell();
    cell[0].Text = "Team";
    cell[1].Text = "Last Name";
    cell[2].Text = "Name";
    cell[3].Text = "Position";
    cell[4].Text = "Points Scored";
    cell[5].Text = "Minutes Played";
    cell[6].Text = "Mistakes Made";
    row.Cells.AddRange(cell);
    Table2.Rows.Add(row);
    for (int i = 0; i < list.Count; i++)
    {
        row = new TableRow();
        for (int j = 0; j < 7; j++)
            cell[j] = new TableCell();
        cell[0].Text = list[i].Team;
```

```
            cell[1].Text = list[i].LastName;
            cell[2].Text = list[i].Name;
            cell[3].Text = list[i].Position;
            cell[4].Text = list[i].PointsGained.ToString();
            cell[5].Text = list[i].MinutesPlayed.ToString();
            cell[6].Text = list[i].MistakesMade.ToString();
            row.Cells.AddRange(cell);
            Table2.Rows.Add(row);
        }
    }
    #endregion

    #region Exception Control
    /// <summary>
    /// Looks through file array, throws an exception if none are found
    /// </summary>
    /// <param name="filePaths">string array of files</param>
    /// <param name="PlayerInfo">player info file path</param>
    public void FileExceptionControl(string[] filePaths, string PlayerInfo)
    {
        try
        {
            if (filePaths.Length == 0) throw new Exception("Cannot find any Match*.txt
                files in given location");
            else if (!File.Exists(PlayerInfo)) throw new Exception("Cannot find
                PlayerInfo.txt file in given location");
        }
        catch (Exception ex) { throw ex; }
    }
    /// <summary>
    /// Looks for an error in the match files.
    /// </summary>
    /// <param name="val">player information</param>
    /// <param name="counter">line counter</param>
    /// <param name="file">file location</param>
    public bool MatchFileControl(string[] val, int counter, string file)
    {
        try
        {
            if (val.Length != 6) throw new Exception(String.Format("There is a mistake in
                line {0} in the file {1}.", counter, file));
            for (int i = 3; i < 6; i++)
                if (int.TryParse(val[i], out int rez) == false)
                    throw new Exception(string.Format("There is mistake in the line {0} in
                the file {1}. There should be a number instead of a char.", counter,
                file));
        }
        catch (Exception)
        {
            Label5.Text += String.Format("There is a mistake in line {0} in the file
                {1}.{2}", counter, file, Environment.NewLine);
            return true;
        }
        return false;
    }
    /// <summary>
    /// looks for mistakes in playerinfo file
    /// </summary>
    /// <param name="val">player information</param>
    /// <param name="counter">line counter</param>
    /// <param name="file">file path</param>
    public bool PlayerInfoControl(string[] val, int counter, string file)
    {
        try
        {
            if (val.Length != 4) throw new Exception(String.Format("There is a mistake in
                 line {0} in the file {1}.", counter, file));
        }
        catch (Exception)
        {
            if (!Label5.Text.Contains(String.Format("There is a mistake in line {0} in the
                 file {1}.", counter, file)))
                Label5.Text += String.Format("There is a mistake in line {0} in the file
                 {1}.{2}", counter, file, Environment.NewLine);
            return true;
        }
        return false;
    }
    /// <summary>
    /// User interface exception control
    /// </summary>
    /// <param name="S">user selected starting date</param>
    /// <param name="E">user selected ending date</param>
    /// <param name="Amount">user selected amount of players</param>
    /// <param name="WPos">user selected wanted position</param>
    public void UserInterfaceExceptions(string S, string E, string Amount, string WPos)
    {
        try
        {
            if (DateTime.TryParse(S, out DateTime rez) == false)
                throw new Exception(string.Format("Starting date is in the incorrect
                 format."));
            if (DateTime.TryParse(E, out DateTime rezz) == false)
                throw new Exception(string.Format("Ending date is in the incorrect
                 format."));
            if (DateTime.Parse(E) < DateTime.Parse(S))
                throw new Exception(string.Format("The starting date has to be before the
                 ending date."));
            if (WPos != "Striker" && WPos != "Defender" && WPos != "Center")
                throw new Exception(string.Format("The chosen position does not exist."));
            if (int.TryParse(Amount, out int rezzz) == false)
                throw new Exception(string.Format("The chosen amount is not a number."));
        }
        catch (Exception ex) { throw ex; }
    }
    /// <summary>
    /// checkbox for input data
    /// </summary>
    public void InputDataShow()
    {
        if (CheckBox1.Checked)
            Table1.Visible = true;
        if (!CheckBox1.Checked)
            Table1.Visible = false;
    }
    /// <summary>
    /// empty file exception control
    /// </summary>
    /// <param name="file">file path</param>
    public void EmptyFileException(string file)
    {
        try
        {
            string[] a = File.ReadAllLines(file);
            if (a.Length == 0)
                throw new Exception(string.Format("The File {0} is empty", file));
        }
        catch (Exception ex) { throw ex; }
    }
    #endregion
    }
            }
```

5.7 Pradiniai duomenys ir rezultatai

Pirmas bandymas:

Match1.txt:

```
1999-09-25
Pavadinimas;Pavarde;Vardas;45;20;2
Pavadinimas2;Pav;Var;20;2;5
```

Match2.txt

```
1996-12-10
Rookies;Lapelis;Andrius;30;15;15
Lambda;Maciukis;Lukas;0;50;4
Rookies;Zajacas;Julijus;6;10;1
Lambda;Drapas;Miluzis;10;20;0
```

Match3.txt

```
1998-05-28
Zarbiris;Budas;Rokas;32;15;3
Nupbunas;Poldis;Sargis;31;15;3
Zarbiris;Juoskevicius;Juozapelis;15;4;1
Zarbiris;Patrovicius;Petras;15;4;2
Napbunas;Puogis;Zubis;32;15;0
Napbunas;Laurius;Aurius;29;2;4
Zarbiris;Nuokalnis;Laurius;2;2;5
Napbunas;Pagalius;Zitas;31;15;0
```

Match4.txt

```
1998-04-12
```

PlayerInfo.txt

```
Pavadinimas;Pavarde;Vardas;Striker
Pavadinimas2;Pav;Var;Defender
Rookies;Lapelis;Andrius;Center
Lambda;Maciukis;Lukas;Center
Rookies;Zajacas;Julijus;Striker
Lambda;Drapas;Miluzis;Defender
Zarbiris;Budas;Rokas;Center
Nupbunas;Poldis;Sargis;Center
Zarbiris;Juoskevicius;Juozapelis;Striker
Zarbiris;Patrovicius;Petras;Defender
Napbunas;Puogis;Zubis;Striker
Napbunas;Laurius;Aurius;Center
Zarbiris;Nuokalnis;Laurius;Center
Napbunas;Pagalius;Zitas;Center
```

Answer.txt

```
9/25/1999 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Pavadinimas |         Pavarde |          Vardas |         Striker |              20 |              45 |               2 |
   Pavadinimas2 |             Pav |             Var |        Defender |               2 |              20 |               5 |

12/10/1996 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
        Rookies |         Lapelis |         Andrius |          Center |              15 |              30 |              15 |
         Lambda |        Maciukis |           Lukas |          Center |              50 |               0 |               4 |
        Rookies |         Zajacas |         Julijus |         Striker |              10 |               6 |               1 |
         Lambda |          Drapas |         Miluzis |        Defender |              20 |              10 |               0 |

5/28/1998 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
       Zarbiris |           Budas |           Rokas |          Center |              15 |              32 |               3 |
       Nupbunas |          Poldis |          Sargis |          Center |              15 |              31 |               3 |
       Zarbiris |    Juoskevicius |      Juozapelis |         Striker |               4 |              15 |               1 |
       Zarbiris |     Patrovicius |          Petras |        Defender |               4 |              15 |               2 |
       Napbunas |          Puogis |           Zubis |         Striker |              15 |              32 |               0 |
       Napbunas |         Laurius |          Aurius |          Center |               2 |              29 |               4 |
       Zarbiris |       Nuokalnis |         Laurius |          Center |               2 |               2 |               5 |
       Napbunas |        Pagalius |           Zitas |          Center |              15 |              31 |               0 |

4/12/1998 12:00:00 AM
List is empty

Chosen position by the user: Striker
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
-----------------------------------------------------------------------------------------------------------------------------
    Pavadinimas |         Pavarde |          Vardas |         Striker |              20 |              45 |               2 |
       Napbunas |          Puogis |           Zubis |         Striker |              15 |              32 |               0 |
        Rookies |         Zajacas |         Julijus |         Striker |              10 |               6 |               1 |
       Zarbiris |    Juoskevicius |      Juozapelis |         Striker |               4 |              15 |               1 |
```

2 bandymas:

Match1.txt

```
1999-09-25
Pavadinimas;Pavarde;Vardas;45;20;2
Pavadinimas2;Pav;Var;20;2;5
```

Match2.txt

```
1996-12-10
Rookies;Lapelis;Andrius;30;15;15
Lambda;Maciukis;Lukas;20;31;4
Rookies;Zajacas;Julijus;6;10;1
Lambda;Drapas;Miluzis;10;20;0
```

Match3.txt

```
1998-05-28
Zarbiris;Budas;Rokas;32;15;3
Nupbunas;Poldis;Sargis;31;15;3
Zarbiris;Juoskevicius;Juozapelis;15;4;1
Zarbiris;Patrovicius;Petras;15;4;2
Napbunas;Puogis;Zubis;32;15;0
Napbunas;Laurius;Aurius;29;2;4
Zarbiris;Nuokalnis;Laurius;2;2;5
Napbunas;Pagalius;Zitas;31;15;0
```

Match4.txt

```
1998-04-12
Pomidorai;Kardas;Auksinis;29;15;3
Pomidorai;Princese;Auksuole;15;2;0
Agurkai;Giedriauskas;Lukis;0;0;0
Pomidorai;Faker;Drauguzis;5;2;4
Agurkai;Mandruolis;Sirijus;50;34;2
Agurkai;Poniulis;Dragunas;45;34;1
Pomidorai;Zykiukas;Sauliukas;45;34;0
```

Match5.txt

```
1976-04-20
Klumpakojai;Jaunoji;Aldona;20;9;3
Klumpakojai;Pirdzius;Tilius;30;21;0
Puodziai;Butrimas;Dalius;45;23;0
Puodziai;Kaztonius;Simas;30;21;1
Klumpakojai;Gelius;Juozapelis;60;49;1
Puodziai;Niukstis;Juonius;12;3;0
Klumpakojai;Baltrius;Justas;61;49;2
Puodziai;Saule;Jurgita;24;10;2
```

PlayerInfo.txt

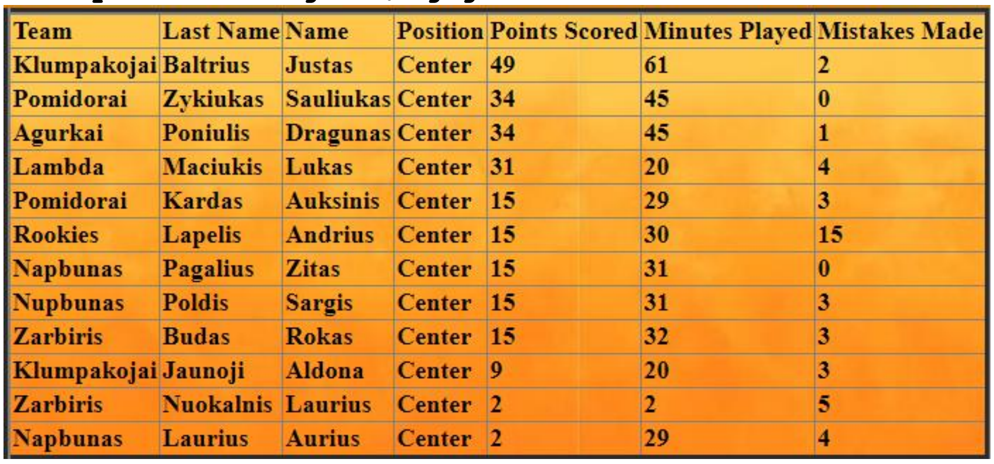
```
Pavadinimas;Pavarde;Vardas;Striker
Pavadinimas2;Pav;Var;Defender
Rookies;Lapelis;Andrius;Center
Lambda;Maciukis;Lukas;Center
Rookies;Zajacas;Julijus;Striker
Lambda;Drapas;Miluzis;Defender
Zarbiris;Budas;Rokas;Center
Nupbunas;Poldis;Sargis;Center
Zarbiris;Juoskevicius;Juozapelis;Striker
Zarbiris;Patrovicius;Petras;Defender
Napbunas;Puogis;Zubis;Striker
Napbunas;Laurius;Aurius;Center
Zarbiris;Nuokalnis;Laurius;Center
Napbunas;Pagalius;Zitas;Center
Pomidorai;Kardas;Auksinis;Center
Pomidorai;Princese;Auksuole;Striker
Agurkai;Giedriauskas;Lukis;Defender
Pomidorai;Faker;Drauguzis;Striker
Agurkai;Mandruolis;Sirijus;Defender
Agurkai;Poniulis;Dragunas;Center
Pomidorai;Zykiukas;Sauliukas;Center
Klumpakojai;Jaunoji;Aldona;Center
Klumpakojai;Pirdzius;Tilius;Striker
Puodziai;Butrimas;Dalius;Striker
Puodziai;Kaztonius;Simas;Defender
Klumpakojai;Gelius;Juozapelis;Striker
Puodziai;Niukstis;Juonius;Defender
Klumpakojai;Baltrius;Justas;Center
Puodziai;Saule;Jurgita;Defender
```

Pasirinkus pradinę data 1960-01-01, galinę 2000-12-11, poziciją: Center, o rodomų žaidėjų skaičių 20, gaunami tokie atsakymai:

Answer.txt

```
9/25/1999 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
----------------------------------------------------------------------------------------------------------------------------
    Pavadinimas |         Pavarde |          Vardas |         Striker |              20 |              45 |               2 |
   Pavadinimas2 |             Pav |             Var |        Defender |               2 |              20 |               5 |

12/10/1996 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
----------------------------------------------------------------------------------------------------------------------------
        Rookies |         Lapelis |         Andrius |          Center |              15 |              30 |              15 |
         Lambda |        Maciukis |           Lukas |          Center |              31 |              20 |               4 |
        Rookies |         Zajacas |         Julijus |         Striker |              10 |               6 |               1 |
         Lambda |          Drapas |         Miluzis |        Defender |              20 |              10 |               0 |

5/28/1998 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
----------------------------------------------------------------------------------------------------------------------------
       Zarbiris |           Budas |           Rokas |          Center |              15 |              32 |               3 |
       Nupbunas |          Poldis |          Sargis |          Center |              15 |              31 |               3 |
       Zarbiris |    Juoskevicius |      Juozapelis |         Striker |               4 |              15 |               1 |
       Zarbiris |     Patrovicius |          Petras |        Defender |               4 |              15 |               2 |
       Napbunas |          Puogis |           Zubis |         Striker |              15 |              32 |               0 |
       Napbunas |         Laurius |          Aurius |          Center |               2 |              29 |               4 |
       Zarbiris |       Nuokalnis |         Laurius |          Center |               2 |               2 |               5 |
       Napbunas |        Pagalius |           Zitas |          Center |              15 |              31 |               0 |

4/12/1998 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
----------------------------------------------------------------------------------------------------------------------------
      Pomidorai |          Kardas |        Auksinis |          Center |              15 |              29 |               3 |
      Pomidorai |        Princese |        Auksuole |         Striker |               2 |              15 |               0 |
        Agurkai |    Giedriauskas |           Lukis |        Defender |               0 |               0 |               0 |
      Pomidorai |           Faker |       Drauguzis |         Striker |               2 |               5 |               4 |
        Agurkai |      Mandruolis |         Sirijus |        Defender |              34 |              50 |               2 |
        Agurkai |        Poniulis |        Dragunas |          Center |              34 |              45 |               1 |
      Pomidorai |        Zykiukas |       Sauliukas |          Center |              34 |              45 |               0 |

4/20/1976 12:00:00 AM
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
----------------------------------------------------------------------------------------------------------------------------
    Klumpakojai |         Jaunoji |          Aldona |          Center |               9 |              20 |               3 |
    Klumpakojai |        Pirdzius |          Tilius |         Striker |              21 |              30 |               0 |
       Puodziai |        Butrimas |          Dalius |         Striker |              23 |              45 |               0 |
       Puodziai |       Kaztonius |           Simas |        Defender |              21 |              30 |               1 |
    Klumpakojai |          Gelius |      Juozapelis |         Striker |              49 |              60 |               1 |
       Puodziai |        Niukstis |         Juonius |        Defender |               3 |              12 |               0 |
    Klumpakojai |        Baltrius |          Justas |          Center |              49 |              61 |               2 |
       Puodziai |           Saule |         Jurgita |        Defender |              10 |              24 |               2 |

Chosen position by the user: Center
           Team |        LastName |            Name |        Position |    PointsGained |   MinutesPlayed |    MistakesMade |
----------------------------------------------------------------------------------------------------------------------------
    Klumpakojai |        Baltrius |          Justas |          Center |              49 |              61 |               2 |
      Pomidorai |        Zykiukas |       Sauliukas |          Center |              34 |              45 |               0 |
        Agurkai |        Poniulis |        Dragunas |          Center |              34 |              45 |               1 |
         Lambda |        Maciukis |           Lukas |          Center |              31 |              20 |               4 |
      Pomidorai |          Kardas |        Auksinis |          Center |              15 |              29 |               3 |
        Rookies |         Lapelis |         Andrius |          Center |              15 |              30 |              15 |
       Napbunas |        Pagalius |           Zitas |          Center |              15 |              31 |               0 |
       Nupbunas |          Poldis |          Sargis |          Center |              15 |              31 |               3 |
       Zarbiris |           Budas |           Rokas |          Center |              15 |              32 |               3 |
    Klumpakojai |         Jaunoji |          Aldona |          Center |               9 |              20 |               3 |
       Zarbiris |       Nuokalnis |         Laurius |          Center |               2 |               2 |               5 |
       Napbunas |         Laurius |          Aurius |          Center |               2 |              29 |               4 |
```

Atsakymai vartotojo sąsajoje:



Team	Last Name	Name	Position	Points Scored	Minutes Played	Mistakes Made
Klumpakojai	Baltrius	Justas	Center	49	61	2
Pomidorai	Zykiukas	Sauliukas	Center	34	45	0
Agurkai	Poniulis	Dragunas	Center	34	45	1
Lambda	Maciukis	Lukas	Center	31	20	4
Pomidorai	Kardas	Auksinis	Center	15	29	3
Rookies	Lapelis	Andrius	Center	15	30	15
Napbunas	Pagalius	Zitas	Center	15	31	0
Nupbunas	Poldis	Sargis	Center	15	31	3
Zarbiris	Budas	Rokas	Center	15	32	3
Klumpakojai	Jaunoji	Aldona	Center	9	20	3
Zarbiris	Nuokalnis	Laurius	Center	2	2	5
Napbunas	Laurius	Aurius	Center	2	29	4

3 bandymas:
Šiame bandyme programai neduosime Match*.txt failų. Gausime tokį exception:

Exception Details: System.Exception: Cannot find any Match*.txt files in given location

4 bandymas:
Dabar vartotojo sąsajoje įrašyisime netinkamus duomenis:

Nepasirinkę pozicijos, gauname tokį exception:

Exception Details: System.Exception: The chosen position does not exist.

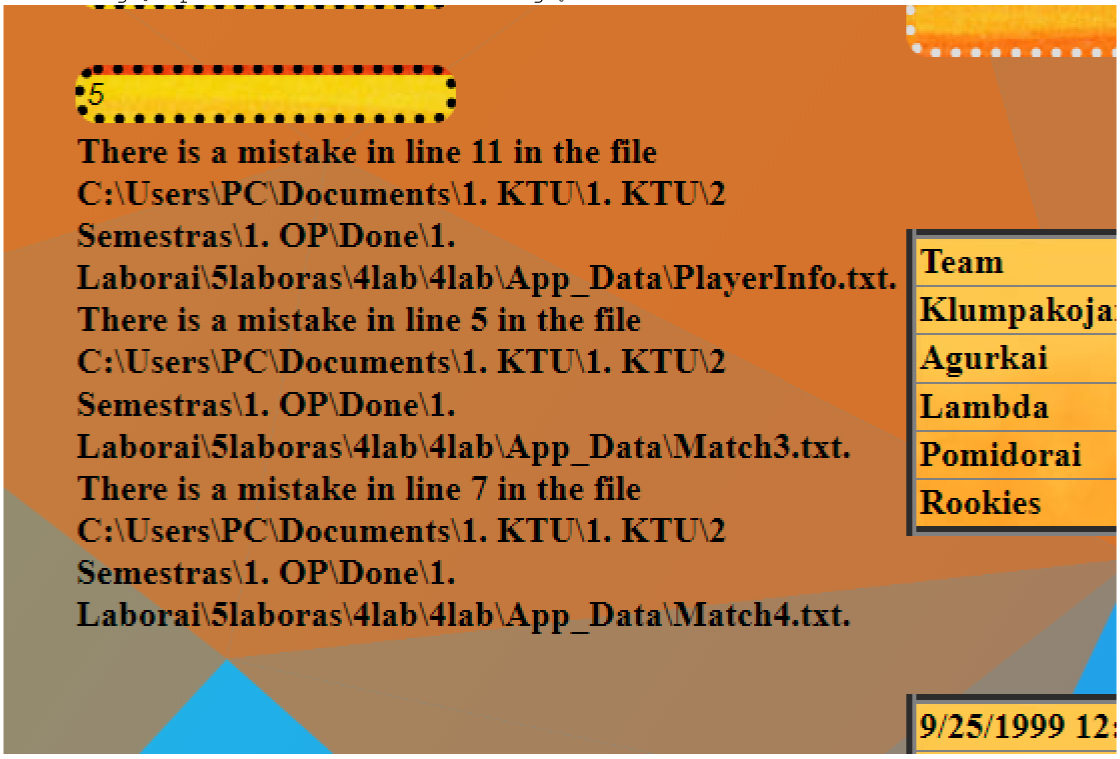
Įrašę netinkamą datą, gauname tokį exception:

Exception Details: System.Exception: Starting date is in the incorrect format.

Į player amound lauką įrašę ne skaičių, guname tokį exception:

Exception Details: System.Exception: The chosen amount is not a number.

5 bandymas:
Failuose esnat klaidoms, programa toliau atlieka skaičiavimus bei praneša vartotoją apie esamas klaidas ir jų vietas failuose:



5

There is a mistake in line 11 in the file C:\Users\PC\Documents\1. KTU\1. KTU\2 Semestras\1. OP\Done\1. Laborai\5laboras\4lab\4lab\App_Data\PlayerInfo.txt. There is a mistake in line 5 in the file C:\Users\PC\Documents\1. KTU\1. KTU\2 Semestras\1. OP\Done\1. Laborai\5laboras\4lab\4lab\App_Data\Match3.txt. There is a mistake in line 7 in the file C:\Users\PC\Documents\1. KTU\1. KTU\2 Semestras\1. OP\Done\1. Laborai\5laboras\4lab\4lab\App_Data\Match4.txt.

Team
Klumpakoja
Agurkai
Lambda
Pomidorai
Rookies

9/25/1999 12:

5.8 Dėstytojo pastabos

Testo rezultatas: 0