Kauno technologijos universitetas
Informatikos fakultetas

Objektinis programavimas 2 (P175B123)

Laboratorinių darbų ataskaita

Edvinas Urbonas IFF4-6

Studentas

Lekt. Simonavičius Kęstutis

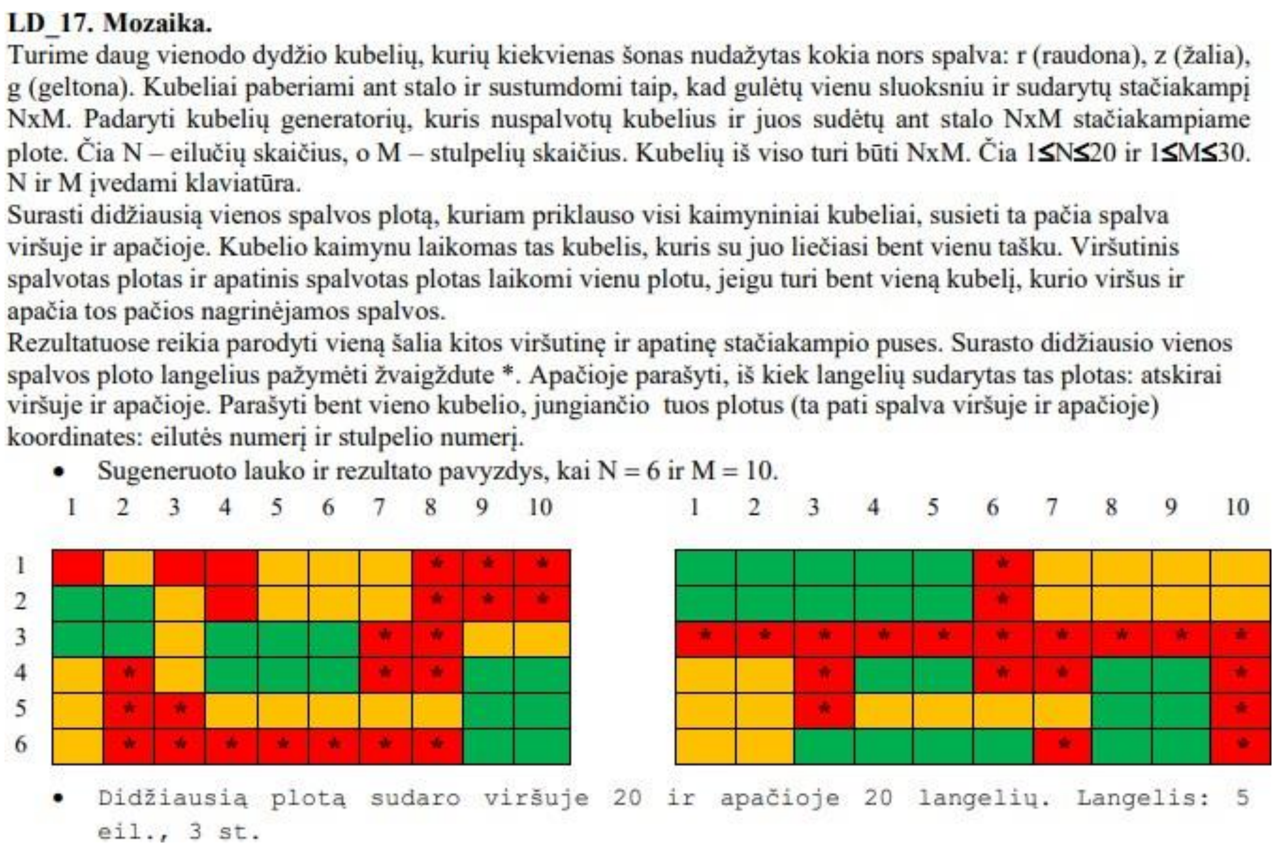
Dėstytojas

Kaunas 2025

TURINYS

1. Rekursija (L1)

1.1. Darbo užduotis



LD_17. Mozaika.
Turime daug vienodo dydžio kubelių, kurių kiekvienas šonas nudažytas kokia nors spalva: r (raudona), z (žalia), g (geltona). Kubeliai paberiami ant stalo ir sustumdomi taip, kad gulėtų vienu sluoksniu ir sudarytų stačiakampį NxM. Padaryti kubelių generatorių, kuris nuspalvotų kubelius ir juos sudėtų ant stalo NxM stačiakampiame plote. Čia N – eilučių skaičius, o M – stulpelių skaičius. Kubelių iš viso turi būti NxM. Čia 1≤N≤20 ir 1≤M≤30. N ir M įvedami klaviatūra.
Surasti didžiausią vienos spalvos plotą, kuriam priklauso visi kaimyniniai kubeliai, susieti ta pačia spalva viršuje ir apačioje. Kubelio kaimynu laikomas tas kubelis, kuris su juo liečiasi bent vienu tašku. Viršutinis spalvotas plotas ir apatinis spalvotas plotas laikomi vienu plotu, jeigu turi bent vieną kubelį, kurio viršus ir apačia tos pačios nagrinėjamos spalvos.
Rezultatuose reikia parodyti vieną šalia kitos viršutinę ir apatinę stačiakampio puses. Surasto didžiausio vienos spalvos ploto langelius pažymėti žvaigždute *. Apačioje parašyti, iš kiek langelių sudarytas tas plotas: atskirai viršuje ir apačioje. Parašyti bent vieno kubelio, jungiančio tuos plotus (ta pati spalva viršuje ir apačioje) koordinates: eilutės numerį ir stulpelio numerį.

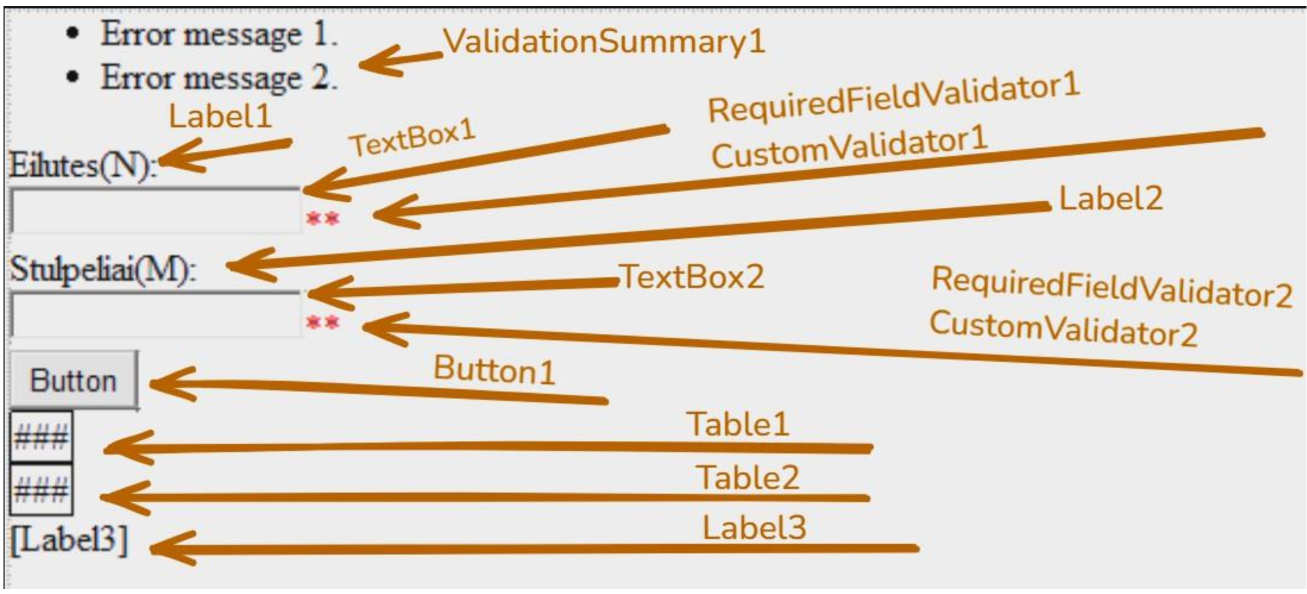
- Sugeneruoto lauko ir rezultato pavyzdys, kai N = 6 ir M = 10.

	1	2	3	4	5	6	7	8	9	10
1	🟥	🟨	🟥	🟥	🟨	🟨	🟨	🟥 *	🟥 *	🟥 *
2	🟩	🟩	🟨	🟥	🟨	🟨	🟨	🟥 *	🟥 *	🟥 *
3	🟩	🟩	🟨	🟩	🟩	🟩	🟥 *	🟥 *	🟨	🟨
4	🟨	🟥 *	🟨	🟩	🟩	🟩	🟥 *	🟥 *	🟩	🟩
5	🟨	🟥 *	🟥 *	🟨	🟨	🟨	🟨	🟨	🟩	🟩
6	🟨	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟩	🟩

1	2	3	4	5	6	7	8	9	10
🟩	🟩	🟩	🟩	🟩	🟥 *	🟨	🟨	🟨	🟨
🟩	🟩	🟩	🟩	🟩	🟥 *	🟨	🟨	🟨	🟨
🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *	🟥 *
🟨	🟨	🟥 *	🟩	🟩	🟥 *	🟥 *	🟩	🟩	🟥 *
🟨	🟨	🟥 *	🟨	🟨	🟨	🟨	🟩	🟩	🟥 *
🟨	🟨	🟩	🟩	🟩	🟩	🟥 *	🟩	🟩	🟥 *

- `Didžiausią plotą sudaro viršuje 20 ir apačioje 20 langelių. Langelis: 5 eil., 3 st.`

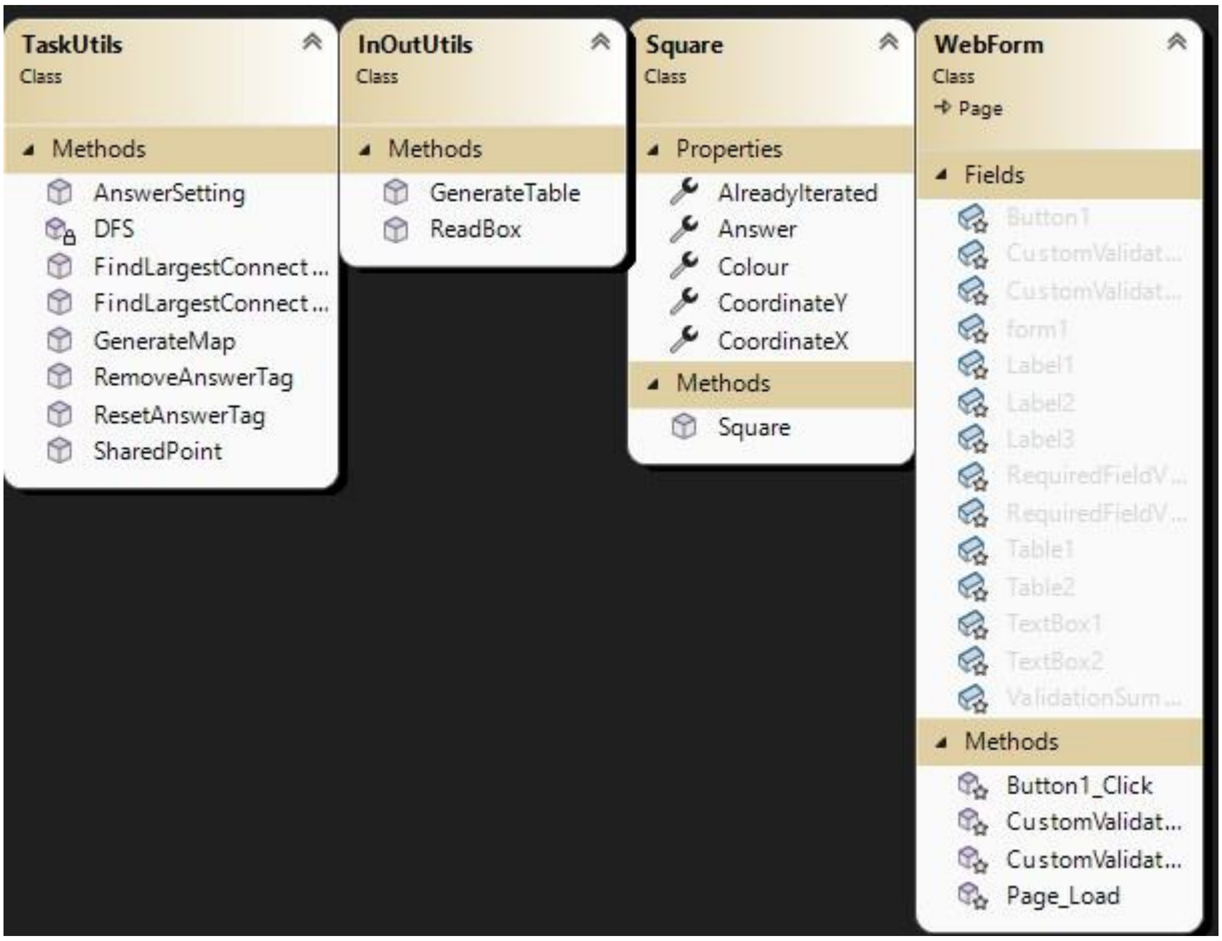
1.2. Grafinės naudotojo sąsajos schema



1.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
ValidatorSummary	Text	
Label1	Text	Eilutės(N):
Label2	Text	Stulpeliai(M):
Label3	Text	
Table1	BorderColor	Black
Table1	BorderStyle	Solid
Table1	BorderWidth	1px
Table1	GridLines	Both
Table2	BorderColor	Black
Table2	BorderStyle	Solid
Table2	BorderWidth	1px
Table2	GridLines	Both
RequiredFieldValidator1	ControlToValidate	TextBox1
RequiredFieldValidator1	ForeColor	Red
RequiredFieldValidator1	Text	*
RequiredFieldValidator1	ErrorMessage	Do not leave empty spaces
RequiredFieldValidator2	ControlToValidate	TextBox2
RequiredFieldValidator2	ForeColor	Red
RequiredFieldValidator2	Text	*
RequiredFieldValidator2	ErrorMessage	Do not leave empty spaces
CustomFieldValidator1	ControlToValidate	TextBox1
CustomFieldValidator1	TextMessage	*
CustomFieldValidator1	ErrorMessage	Digit, 1<=N<=20
CustomFieldValidator1	ForeColor	Red
CustomFieldValidator2	ControlToValidate	TextBox2
CustomFieldValidator2	TextMessage	*
CustomFieldValidator2	ErrorMessage	Digit, 1<=M<=30
CustomFieldValidator2	ForeColor	Red

1.4. Klasių diagrama



1.5. Programos naudotojo vadovas

Vartotojas atsidaręs svetainę įrašo skaičius į eilutes. Pirmoje eilutėje įrašomas eilučių skaičius N ($1 \leq N \leq 20$) ir M ($1 \leq M \leq 30$) ir paspaudžia mygtuką „Button“. Jį paspaudus mozaika išspausdinama žemiau mygtuko, o tekstinis atsakymas žemiau mozaikos.

1.6. Programos tekstas

```
namespace L1
{
    /// <summary>
    /// class meant to send out and receive data
    /// </summary>
    public class InOutUtils
    {

        /// <summary>
        /// Reads the user input
        /// </summary>
        /// <param name="Box"></param>
        /// <returns></returns>
        public static int ReadBox(TextBox Box)
        {
            return int.Parse(Box.Text);
        }
        /// <summary>
        /// prints results to TXT
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <param name="filePath"></param>
        public static void PrintToTxt(Square[,] map, int N, int M, string filePath)
        {
            for (int i = 0; i < N; i++)
            {
                for (int j = 0; j < M; j++)
                {
                    File.AppendAllText(filePath, map[i,j].Colour.ToString());
                }
                File.AppendAllText(filePath, "\n");
            }
            File.AppendAllText(filePath, "\n");
        }
        /// <summary>
        /// Prints the header to TXT seperatly
        /// </summary>
        /// <param name="header"></param>
        /// <param name="filePath"></param>
        public static void PrintHeaderToTxt(string header, string filePath)
        {
            File.AppendAllText(filePath, header);
        }
    }
}
```

```
namespace L1
{
    /// <summary>
    /// Class that defines one part of the blob. AlreadyIterated and Answer are
false by default
    /// </summary>
    public class Square
    {
        public int CoordinateX { get; set; }
        public int CoordinateY { get; set; }
        public int Colour { get; set; } // 0 - green, 1 - red, 2 - yellow
        public bool AlreadyIterated { get; set; } = false;
        public bool Answer { get; set; } = false;
        /// <summary>
        /// constructor for the class
        /// </summary>
        /// <param name="coordinateX"></param>
        /// <param name="coordinateY"></param>
        /// <param name="colour"></param>
        public Square(int coordinateX, int coordinateY, int colour)
        {
            this.CoordinateX = coordinateX;
            this.CoordinateY = coordinateY;
            this.Colour = colour;
        }
    }
}
-----------------------------------------------------------------------------

namespace L1
{
    /// <summary>
    /// Class meant to store task related methods
    /// </summary>
    public class TaskUtils
    {
        /// <summary>
        /// generates the random numbers used for the random colour layout
        /// </summary>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <returns></returns>
        public static Square[,] GenerateMap(int N, int M)
        {
            Random rnd = new Random();
            Square[,] Map = new Square[N, M];

            for (int i = 0; i < N; i++)
            {
                for (int j = 0; j < M; j++)
                {
                    int colour = rnd.Next(0, 3);
                    Map[i, j] = new Square(i, j, colour);
                }
            }
            return Map;
        }
        /// <summary>
        /// Finds the largest area
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <param name="size"></param>
        /// <param name="largestColor"></param>
        public static void FindLargestConnectedArea(Square[,] map, int N, int M,
    ref int size, ref int largestColor)
        {
            int largestArea = 0;
            List<(int, int)> largestRegion = new List<(int, int)>();
            for (int i = 0; i < N; i++)
            {
                for (int j = 0; j < M; j++)
                {
                    if (!map[i, j].AlreadyIterated)
                    {
                        List<(int, int)> currentRegion = new List<(int, int)>();
                        int areaSize = DFS(map, N, M, i, j, map[i, j].Colour,
    currentRegion);

                        if (areaSize > largestArea)
                        {
                            largestArea = areaSize;
                            largestRegion = new List<(int, int)>(currentRegion);
                            largestColor = map[i, j].Colour;
                        }
                    }
                }
            }
            AnswerSetting(map, largestRegion, ref size);
        }
        /// <summary>
        /// finds the largest area that is connected but only of the specified
    "Winner" colour
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <param name="size"></param>
        /// <param name="largestColor"></param>
        public static void FindLargestConnectedAreaColour(Square[,] map, int N,
    int M, ref int size, ref int largestColor)
        {
            int largestArea = 0;
            List<(int, int)> largestRegion = new List<(int, int)>();

            ResetAnswerTag(map, N, M);

            for (int i = 0; i < N; i++)
            {
                for (int j = 0; j < M; j++)
                {
                    if (!map[i, j].AlreadyIterated && map[i, j].Colour ==
    largestColor)
                    {
                        List<(int, int)> currentRegion = new List<(int, int)>();
                        int areaSize = DFS(map, N, M, i, j, largestColor,
    currentRegion);

                        if (areaSize > largestArea)
                        {
                            largestArea = areaSize;
                            largestRegion = new List<(int, int)>(currentRegion);
                        }
                    }
                }
            }
            AnswerSetting(map, largestRegion, ref size);
        }
        /// <summary>
        /// Resets the visited before status. Used so that it can count the second
(smaller) blob without logic problems
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        public static void ResetAnswerTag(Square[,] map, int N, int M)
        {
            for (int i = 0; i < N; i++)
            {
                for (int j = 0; j < M; j++)
                {
                    map[i, j].AlreadyIterated = false;
                }
            }
        }


        /// <summary>
        /// Recursive path finding method, that finds the largest blob of the same
colour
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <param name="x"></param>
        /// <param name="y"></param>
        /// <param name="color"></param>
        /// <param name="region"></param>
        /// <returns></returns>
        static int DFS(Square[,] map, int N, int M, int x, int y, int color,
List<(int, int)> region)
        {
            if (x < 0 || x >= N || y < 0 || y >= M || map[x, y].AlreadyIterated ||
map[x, y].Colour != color)
                return 0;

            map[x, y].AlreadyIterated = true;
            region.Add((x, y));

            int size = 1;
            size += DFS(map, N, M, x - 1, y, color, region); // Up
            size += DFS(map, N, M, x + 1, y, color, region); // Down
            size += DFS(map, N, M, x, y - 1, color, region); // Left
            size += DFS(map, N, M, x, y + 1, color, region); // Right

            return size;
        }
        /// <summary>
        /// Purges the previously set .Answer tag to false. Used to correctly set
the path in the second (smaller) square
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        public static void RemoveAnswerTag(Square[,] map, int N, int M)
        {
            for (int i = 0; i < N; i++)
            {
                for (int j = 0; j < M; j++)
                {
                    map[i, j].Answer = false;
                }
            }
        }
        /// <summary>
        /// Sets the Square.Answer property to true if it was found to the in the
largest area blob
        /// </summary>
        /// <param name="map"></param>
        /// <param name="largestRegion"></param>
        /// <param name="size"></param>
        public static void AnswerSetting(Square[,] map, List<(int, int)>
largestRegion, ref int size)
        {
            foreach (var (x, y) in largestRegion)
            {
                map[x, y].Answer = true;
                size++;
            }
        }
        /// <summary>
        /// Finds the coordinates of the shared vectical point (same colour above
and bellow)
        /// </summary>
        /// <param name="mapOne"></param>
        /// <param name="mapTwo"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <param name="sharedX"></param>
        /// <param name="sharedY"></param>
        public static void SharedPoint(Square[,] mapOne, Square[,] mapTwo, int N,
int M, ref int sharedX, ref int sharedY)
        {
            if (N > 2 && M > 2)
            {
                for (int i = 1; i < N - 1; i++)
                {
                    for (int j = 0; j < M; j++)
                    {
                        if (mapOne[i, j].Answer && mapTwo[i, j].Answer)
                        {
                            bool hasAbove = (mapOne[i - 1, j].Answer || mapTwo[i -
1, j].Answer);
                            bool hasBelow = (mapOne[i + 1, j].Answer || mapTwo[i +
1, j].Answer);

                            if (hasAbove && hasBelow)
                            {
                                sharedX = i;
                                sharedY = j;
                                return;
                            }
                        }
                    }
                }
            }
        }
    }
}
```

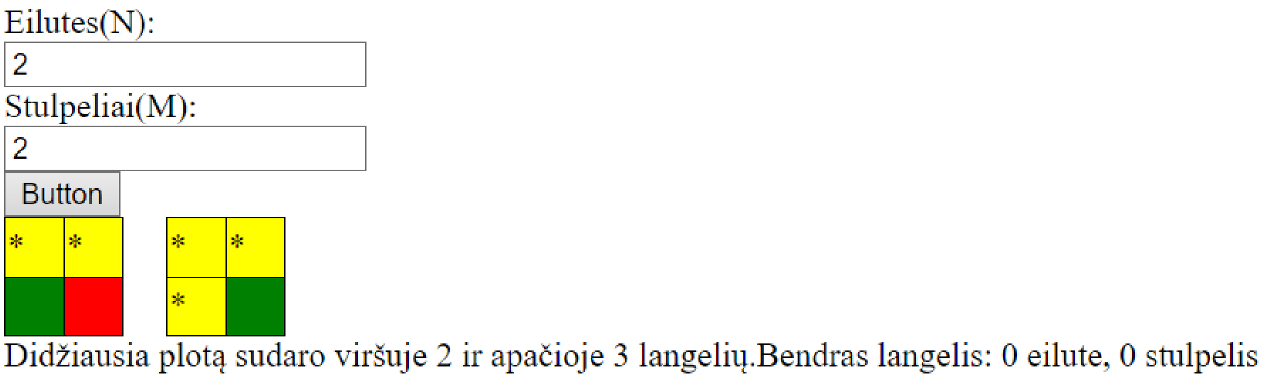
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm.aspx.cs"
Inherits="L1.WebForm" %>
```

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style>
        .tables-container {
            display: flex;
            gap: 20px;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
        <div style="height: 500px">
            <asp:ValidationSummary ID="ValidationSummary1" runat="server"/>
            <asp:Label ID="Label1" runat="server" Text="Eilutes(N):"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
            <asp:RequiredFieldValidator id="RequiredFieldValidator1"
runat="server" ControlToValidate="TextBox1" ForeColor="Red" ErrorMessage="Do not
leave empty spaces" Text="*"></asp:RequiredFieldValidator>
            <asp:CustomValidator ID="CustomValidator1" runat="server"
ControlToValidate="TextBox1" ForeColor="Red" Text="*"
OnServerValidate="CustomValidator1_ServerValidate" ErrorMessage="Digit, 1<=N<=20"
></asp:CustomValidator>
            <br />
            <asp:Label ID="Label2" runat="server"
Text="Stulpeliai(M):"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
            <asp:RequiredFieldValidator id="RequiredFieldValidator2"
runat="server" ControlToValidate="TextBox2" ForeColor="Red" ErrorMessage="Do not
leave empty spaces" Text="*"></asp:RequiredFieldValidator>
            <asp:CustomValidator ID="CustomValidator2" runat="server"
ControlToValidate="TextBox2" ForeColor="Red" Text="*"
OnServerValidate="CustomValidator2_ServerValidate" ErrorMessage="Digit,
1<=M<=30"></asp:CustomValidator>
            <br />
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="Button" />
            <br />

            <div class="tables-container">
                <asp:Table ID="Table1" runat="server" BorderColor="Black"
BorderStyle="Solid" BorderWidth="1px" GridLines="Both"></asp:Table>
                <asp:Table ID="Table2" runat="server" BorderColor="Black"
BorderStyle="Solid" BorderWidth="1px" GridLines="Both"></asp:Table>
            </div>

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

        </div>
    </form>
</body>
</html>
```

```
namespace L1
{
    public partial class WebForm : System.Web.UI.Page
    {
        /// <summary>
        /// Validate the textbox to check if the user typed in numbers and if they fit
the desired range
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator1_ServerValidate(object source,
ServerValidateEventArgs args)
        {
            int N;
            args.IsValid = int.TryParse(TextBox1.Text, out N) && N >= 1 && N <= 20;
        }

        /// <summary>
        /// Validate the textbox to check if the user typed in numbers and if they fit
the desired range
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator2_ServerValidate(object source,
ServerValidateEventArgs args)
        {
            int M;
            args.IsValid = int.TryParse(TextBox2.Text, out M) && M >= 1 && M <= 30;
        }
        /// <summary>
        /// Generates the graphical output in the website
        /// </summary>
        /// <param name="map"></param>
        /// <param name="N"></param>
        /// <param name="M"></param>
        /// <param name="Table"></param>
        public static void GenerateTable(Square[,] map, int N, int M, Table Table)
        {
            Table.Rows.Clear();

            for (int i = 0; i < N; i++)
            {
                TableRow row = new TableRow();
                for (int j = 0; j < M; j++)
                {
                    TableCell cell = new TableCell();
                    int colorCode = map[i, j].Colour;
                    switch (colorCode)
                    {
                        case 0:
                            cell.BackColor = System.Drawing.Color.Green;
                            break;
                        case 1:
                            cell.BackColor = System.Drawing.Color.Red;
                            break;
                        case 2:
                            cell.BackColor = System.Drawing.Color.Yellow;
                            break;
                    }
                    if (map[i, j].Answer)
                    {
                        cell.Text = "*";
                    }
                    else
                    {
                        cell.Text = " ";
                    }

                    cell.Style["width"] = "25px";
                    cell.Style["height"] = "25px";
```

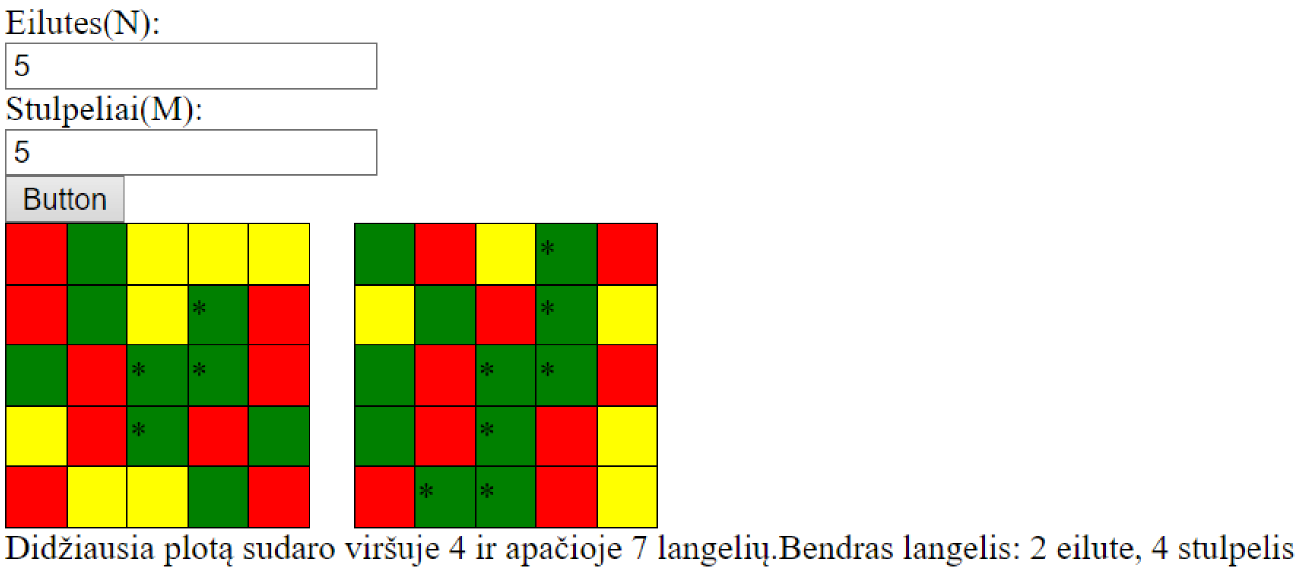
```
                        row.Cells.Add(cell);

                    }
                    Table.Rows.Add(row);
                }
            }
        }
    }

namespace L1
{
    public partial class WebForm : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            Page.Validate();
            if (Page.IsValid)
            {
                string FilePath = Server.MapPath(@"App_Data/Results.txt");
                File.Delete(FilePath);

                int N = InOutUtils.ReadBox(TextBox1); int M =
InOutUtils.ReadBox(TextBox2);
                int sizeOne = 0; int sizeTwo = 0; int colourOne = -1; int colourTwo = -
1;

                Square[,] filledMapOne = TaskUtils.GenerateMap(N, M);
Thread.Sleep(250);
                Square[,] filledMapTwo = TaskUtils.GenerateMap(N, M);

                TaskUtils.FindLargestConnectedArea(filledMapOne, N, M, ref sizeOne, ref
colourOne);
                TaskUtils.FindLargestConnectedArea(filledMapTwo, N, M, ref sizeTwo, ref
colourTwo);
                if (sizeOne > sizeTwo)
                {
                    TaskUtils.RemoveAnswerTag(filledMapTwo, N, M);
TaskUtils.FindLargestConnectedAreaColour(filledMapTwo, N, M, ref sizeTwo, ref
colourOne);
                }
                else if (sizeTwo > sizeOne)
                {
                    TaskUtils.RemoveAnswerTag(filledMapOne, N, M);
TaskUtils.FindLargestConnectedAreaColour(filledMapOne, N, M, ref sizeOne, ref
colourTwo);
                }

                int sharedX = -1, sharedY = -1;
                TaskUtils.SharedPoint(filledMapOne, filledMapTwo, N, M, ref sharedX,
ref sharedY);

                GenerateTable(filledMapOne, N, M, Table1);
                GenerateTable(filledMapTwo, N, M, Table2);
                string resultText = $"Didžiausia plotą sudaro viršuje {sizeOne} ir
apačioje {sizeTwo} langelių.Bendras langelis: {sharedX + 1} eilute, {sharedY + 1}
stulpelis";
                Label3.Text = resultText;

                InOutUtils.PrintToTxt(filledMapOne, N, M, FilePath);
                InOutUtils.PrintToTxt(filledMapTwo, N, M, FilePath);
                InOutUtils.PrintHeaderToTxt(resultText, FilePath);
            }

        }
    }
}
```

1.7. Pradiniai duomenys ir rezultatai



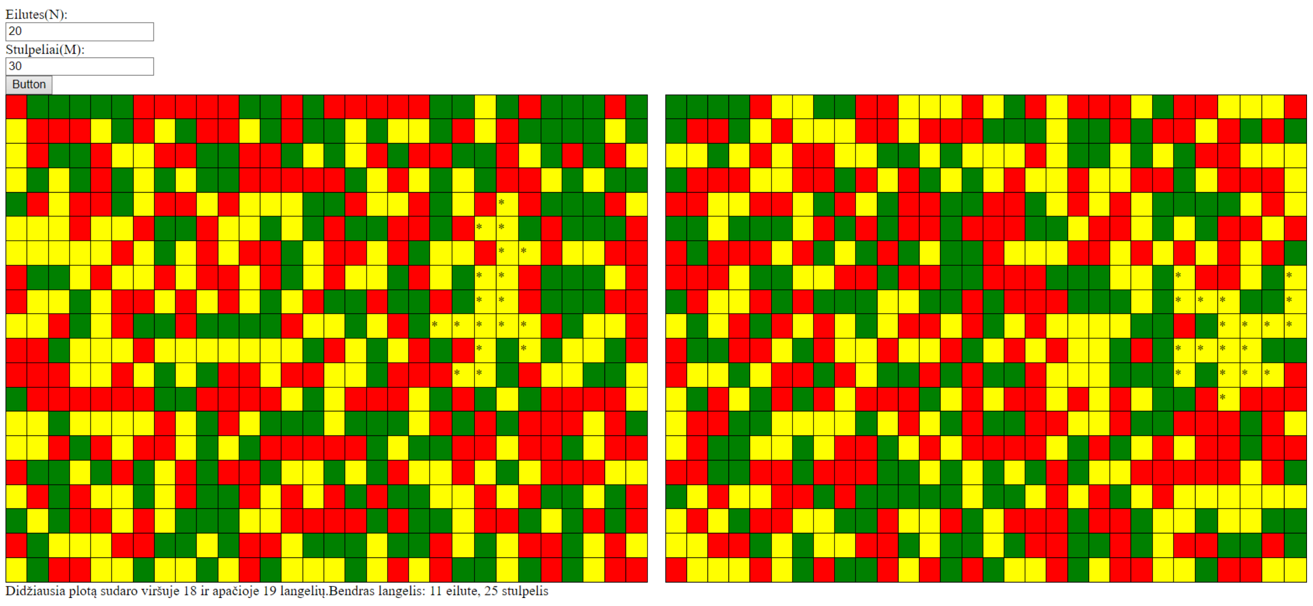
Eilutes(N):
2
Stulpeliai(M):
2

Didžiausia plotą sudaro viršuje 2 ir apačioje 3 langelių.Bendras langelis: 0 eilute, 0 stulpelis



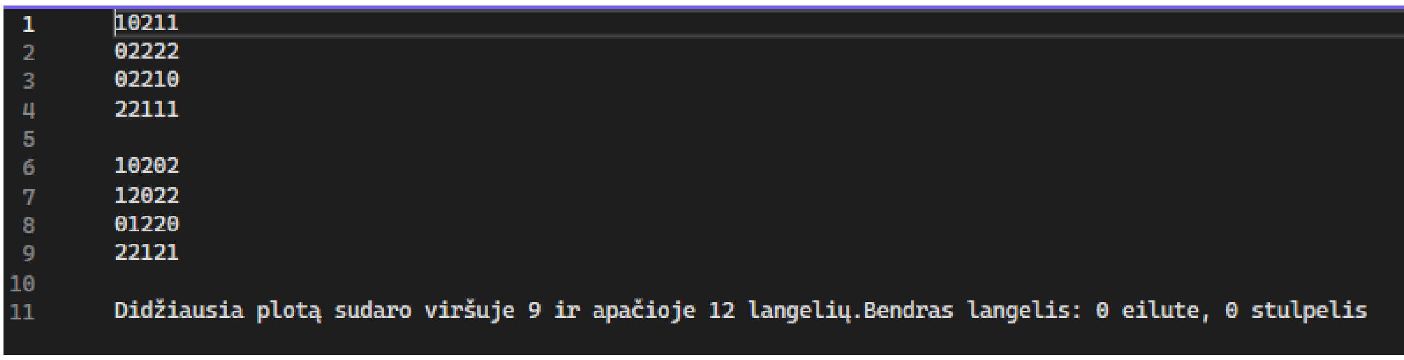
Eilutes(N):
5
Stulpeliai(M):
5

Didžiausia plotą sudaro viršuje 4 ir apačioje 7 langelių.Bendras langelis: 2 eilute, 4 stulpelis



Eilutes(N):
20
Stulpeliai(M):
30

Didžiausia plotą sudaro viršuje 18 ir apačioje 19 langelių.Bendras langelis: 11 eilute, 25 stulpelis


```
10211
02222
02210
22111

10202
12022
01220
22121

Didžiausia plotą sudaro viršuje 9 ir apačioje 12 langelių.Bendras langelis: 0 eilute, 0 stulpelis
```

1.8. Dėstytojo pastabos

Nespausdina rezultatų į .txt failą - Ištaisyta.

2. Dinaminis atminties valdymas (L2)

2.1. Darbo užduotis

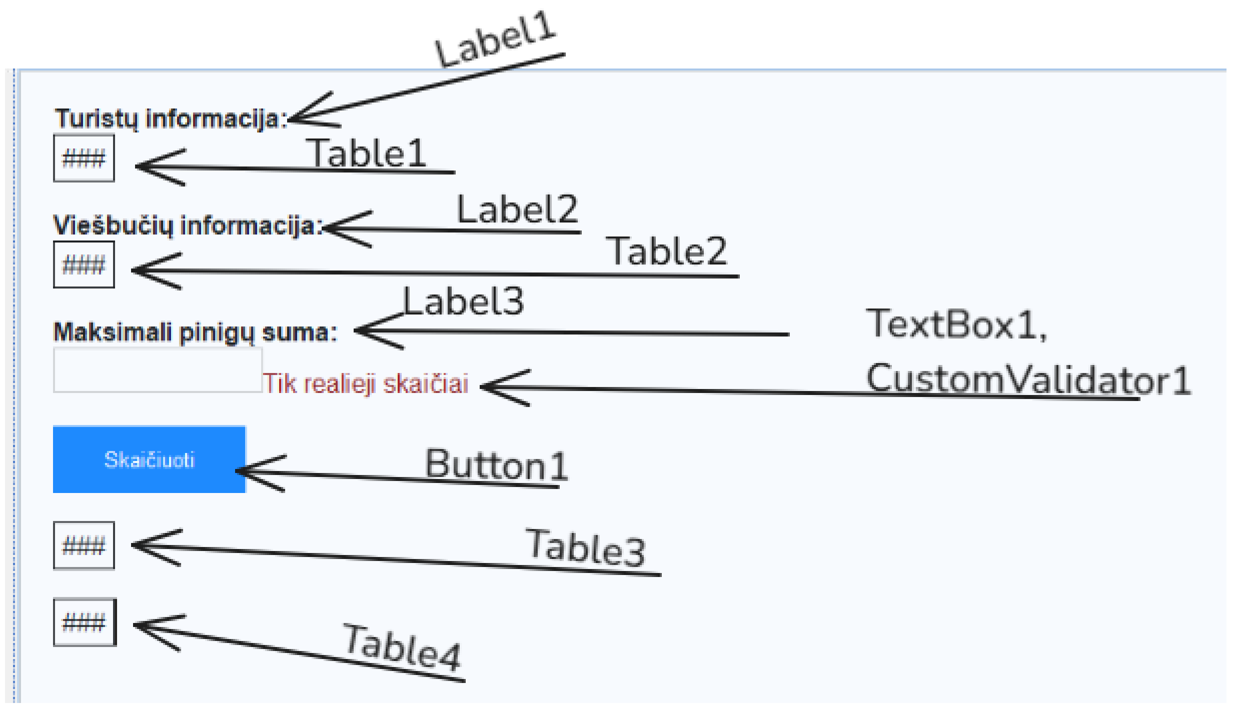
LD_17. **Viešbučiai**. Prieš vasaros keliones keliautojai renkasi viešbučius. Sudarykite keliautojų pasirinktų viešbučių sąrašą. Sudarykite atskirą nepasirinktų viešbučių sąrašą. Sudarykite keliautojų, kurie viešbučiuose nakvos daugiausiai naktų, sąrašą. Sąrašas turi būti surikiuotas pagal keliautojų pavardes ir vardus abėcėlės tvarka.

Duomenys:

- tekstiniame faile U17a.txt yra informacija apie keliautojus: keliautojo pavardė ir vardas, pasirinkto viešbučio pavadinimas, kambario tipas, planuojamas nakvynių skaičius;
- tekstiniame faile U17b.txt yra informacija apie viešbučius: viešbučio pavadinimas, kambario tipas, paros kaina.

Sudarykite keliautojų, kurie už viešbučius sumokėjo pinigų sumą, ne didesnę už nurodytą (įvedama klaviatūra), sąrašą (keliautojo pavardė ir vardas, suma). Sąrašas turi būti surikiuotas pagal keliautojų pavardes ir vardus abėcėlės tvarka.

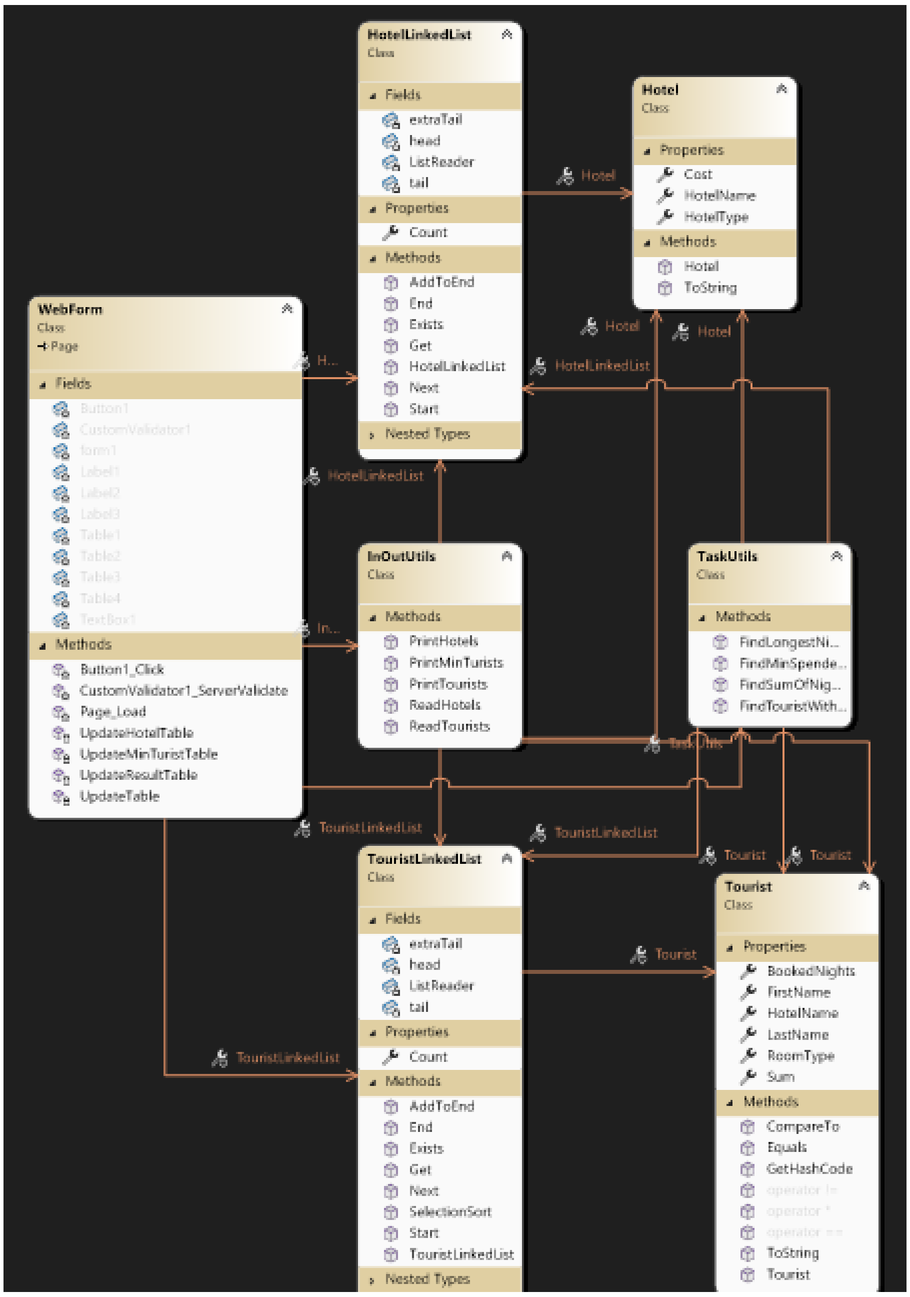
2.2. Grafinės naudotojo sąsajos schema



2.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Label(Bendri pakeitimai)	CssClass	label
Label(Bendri pakeitimai)	Font-weight	bold
Table(Bendri pakeitimai)	CssClass	table
Table(Bendri pakeitimai)	Backround-color	White
Table(Bendri pakeitimai)	Border-color	Black
Table(Bendri pakeitimai)	Border-style	Solid
Table(Bendri pakeitimai)	Border-width	1px
Table(Bendri pakeitimai)	Color	Black
Table(Bendri pakeitimai)	Border-collapse	Collapse
Label1	Text	Turistų informacija:
Label2	Text	Viešbučių informacija:
Label3	Text	Maksimali pinigų suma:
CustomValidator1	Text	Tik Realieji skaičiai
CustomValidator1	ControlToValidate	TextBox1
CustomValidator1	Color	Red
TextBox1	Padding	5px
TextBox1	Border	1px solid
Button1	Padding	10px 20px
Button1	Backround-color	#007bff
Button1	Color	White
Button1	Border	None
Button1	Cursor	Pointer
Button1:Hover	Backround-color	#0056b3

2.4. Klasių diagrama



2.5. Programos naudotojo vadovas

Vartotojas turi sukurti duomenų failus viešbučiam „U17b.txt “ir turistam „U17a.txt“ ir įdėti juos į App_Data aplankalą. Viešbučių duomenų faile duomenys išdėstomi kiekvienoje naujoje eilutėje tokia eiga: Pavardė;Vardas;Viešbučio,kuriame apsistojo pavadinimas;norimas kambario tipas;kiek nakčių praleido(skaičius). Viešbučių informacija tokia eiga: Pavadinimas;Kambarių tipas;nakties kaina(skaičius). Programa yra paleidžiama, atsidariusiame lange įvedama išlaidų limitas, rezultatai matomi lange ir yra atspausdinami į „Rezultatai.txt“

2.6. Programos tekstas

```
namespace L2
{
    /// <summary>
    /// Base class for the hotel object
    /// </summary>
    class Hotel
    {
        public string HotelName { get; set; }
        public string HotelType { get; set; }
        public int Cost { get; set; }
        /// <summary>
        /// Constructor for the hotel object
        /// </summary>
        /// <param name="hotelName"></param>
        /// <param name="hotelType"></param>
        /// <param name="cost"></param>
        public Hotel(string hotelName, string hotelType, int cost)
        {
            HotelName = hotelName;
            HotelType = hotelType;
            Cost = cost;
        }
        /// <summary>
        /// Overriden ToString method for the hotel object
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            string line = $"{HotelName,-15} | {HotelType,-15} | {Cost,5}|";
            return line;
        }
    }
}

namespace L2
{
    /// <summary>
    /// Class for the hotel linked list
    /// </summary>
    class HotelLinkedList
    {
        /// <summary>
        /// Class for the hotel node
        /// </summary>
        private sealed class HotelNode
        {
            public Hotel Data { get; set; }
            public HotelNode Link { get; set; }

            /// <summary>
            /// Constructor for the hotel node
            /// </summary>
            /// <param name="data"></param>
            /// <param name="link"></param>
            public HotelNode(Hotel data, HotelNode link)
            {
                Data = data;
                Link = link;
            }
        }
        public int Count { get; private set; } = 0;
        private HotelNode head; //start
        private HotelNode tail; //end
        private HotelNode extraTail; //end (extra)
        private HotelNode ListReader; // list linker
        /// <summary>
        /// Constructor for the hotel linked list
        /// </summary>
        public HotelLinkedList()
        {
            this.tail = new HotelNode(null, null);
            this.head = new HotelNode(null, this.tail);
            this.extraTail = head;
            this.ListReader = null;
        }
        /// <summary>
        /// Method to add a hotel to the end of the linked list
        /// </summary>
        /// <param name="data"></param>
        public void AddToEnd(Hotel data)
        {
            extraTail.Link = new HotelNode(data, null);
            extraTail = extraTail.Link;
            Count++;
        }
        /// <summary>
        /// Method to get the hotel from the linked list
        /// </summary>
        /// <returns></returns>
        public Hotel Get()
        {
            return ListReader.Data;
        }
        /// <summary>
        /// Method to move to the next hotel in the linked list
        /// </summary>
        public void Next()
        {
            ListReader = ListReader.Link;
        }
        /// <summary>
        /// Method to check if the hotel exists in the linked list
        /// </summary>
        /// <returns></returns>
        public bool Exists()
        {
            return ListReader != null && ListReader.Data != null;
        }
        /// <summary>
        /// Method for the start of the linked list
        /// </summary>
        public void Start()
        {
            ListReader = head.Link;
        }
        /// <summary>
        /// Method for the end of the linked list
        /// </summary>
        public void End()
        {
            ListReader = tail.Link;
        }
    }
}
namespace L2
{
    class InOutUtils
    {
        /// <summary>
        /// Reads tourists from the file
        /// </summary>
        /// <param name="file"></param>
        /// <param name="tourists"></param>
        public static void ReadTourists(string file, TouristLinkedList tourists)
        {
            string[] lines = File.ReadAllLines(file);

            foreach (string line in lines)
            {
                string[] data = line.Split(';');
                string LastName = data[0];
                string FirstName = data[1];
                string Hotel = data[2];
                string RoomType = data[3];
                int Days = int.Parse(data[4]);

                Tourist tourist = new Tourist(LastName, FirstName, Hotel, RoomType, Days);
                tourists.AddToEnd(tourist);
            }
        }
        /// <summary>
        /// Reads hotels from the file
        /// </summary>
        /// <param name="file"></param>
        /// <param name="hotels"></param>
        public static void ReadHotels(string file, HotelLinkedList hotels)
        {
            string[] lines = File.ReadAllLines(file);

            foreach (string line in lines)
            {
                string[] data = line.Split(';');
                string Name = data[0];
                string RoomType = data[1];
                int Price = int.Parse(data[2]);

                Hotel hotel = new Hotel(Name, RoomType, Price);
                hotels.AddToEnd(hotel);
            }
        }


        /// <summary>
        /// Prints tourists to the file
        /// </summary>
        /// <param name="file"></param>
        /// <param name="tourists"></param>
        /// <param name="header"></param>
        /// <param name="answer"></param>
        public static void PrintTourists(string file, TouristLinkedList tourists, string
header, bool answer)
        {
            using (StreamWriter writer = new StreamWriter(file,true))
            {
                string line = new string('-', header.Length);
                if (answer)
                {
                    writer.WriteLine("Turistai, kurie praleido daugiausia nakčių:");
                }
                else
                {
                    writer.WriteLine("Pradiniai turistų duomenys:");
                }
                writer.WriteLine(line);
                writer.WriteLine(header);
                writer.WriteLine(line);
                if (tourists == null)
                {
                    writer.WriteLine("Turistu nėra");
                }
                else
                {
                    for (tourists.Start(); tourists.Exists(); tourists.Next())
                    {
                        Tourist tourist = tourists.Get();
                        writer.WriteLine(tourist.ToString());
                    }
                }
                writer.WriteLine(line);
                writer.WriteLine();
            }
        }
        /// <summary>
        /// Prints hotels to the file
        /// </summary>
        /// <param name="file"></param>
        /// <param name="hotels"></param>
        /// <param name="header"></param>
        public static void PrintHotels(string file, HotelLinkedList hotels, string header)
        {
            using (StreamWriter writer = new StreamWriter(file,true))
            {
                string line = new string('-', header.Length);
                writer.WriteLine("Pradiniai viesbuciu duomenys:");
                writer.WriteLine(line);
                writer.WriteLine(header);
                writer.WriteLine(line);
                for (hotels.Start(); hotels.Exists(); hotels.Next())
                {
                Hotel hotel = hotels.Get();
                writer.WriteLine(hotel.ToString());
                }
                writer.WriteLine(line);
                writer.WriteLine();
            }
        }
        public static void PrintHotelsAnswer(string file, HotelLinkedList hotels, string
    header, bool first)
        {
            using (StreamWriter writer = new StreamWriter(file, true))
            {
                string line = new string('-', header.Length);
                if (first)
                {
                    writer.WriteLine("Pasirinkti viesbuciai:");
                }
                else
                {
                    writer.WriteLine("Nepasirinkti viesbuciai:");
                }

                if (hotels.Count == 0)
                {
                    if (first)
                    {
                        writer.WriteLine("Pasirinktu viesbuciu nera\n");
                    }
                    else
                    {
                        writer.WriteLine("Nepasirinkti viesbuciu nera\n");
                    }
                }
                else
                {
                    writer.WriteLine(line);
                    writer.WriteLine(header);
                    writer.WriteLine(line);
                    for (hotels.Start(); hotels.Exists(); hotels.Next())
                    {
                        Hotel hotel = hotels.Get();
                        writer.WriteLine(hotel.ToString());
                    }
                    writer.WriteLine(line);
                    writer.WriteLine();
                }
            }
        }
        /// <summary>
        /// Prints tourists who paid less than the limit
        /// </summary>
        /// <param name="file"></param>
        /// <param name="tourists"></param>
        /// <param name="header"></param>
        /// <param name="limit"></param>
        public static void PrintMinTurists(string file, TouristLinkedList tourists, string
header, int limit)
        {
            using (StreamWriter writer = new StreamWriter(file, true))
            {
                if (tourists.Count != 0)
                {
                    string line = new string('-', header.Length);
                    writer.WriteLine($"Turistai, kurie už kambarius sumokėjo mažiau negu
{limit}:");
                    writer.WriteLine(line);
                    writer.WriteLine(header);
                    writer.WriteLine(line);
                    if (tourists == null)
                    {
                        writer.WriteLine("Turistu nėra");
                    }
                    else
                    {
                        for (tourists.Start(); tourists.Exists(); tourists.Next())
                        {
                            Tourist tourist = tourists.Get();
                            string format = $"{tourist.LastName,-15} | {tourist.FirstName,-15}
| {tourist.Sum,10}|";
                            writer.WriteLine(format);
                        }
                    }
                    writer.WriteLine(line);
                    writer.WriteLine();
                }
                else
                {
                    writer.WriteLine("Turistu, kurie už kambarius sumokėjo mažiau negu
nurodyta, nėra");
                }
            }
        }
    }
}
namespace L2
{
    class TaskUtils
    {
        /// <summary>
        /// Finds the sum of nights in hotels for each tourist
        /// </summary>
        /// <param name="tourists"></param>
        /// <param name="hotels"></param>
        public static void FindSumOfNightsInHotels(TouristLinkedList tourists, HotelLinkedList
hotels)
        {
            for (tourists.Start(); tourists.Exists(); tourists.Next())
            {
                Tourist tourist = tourists.Get();
                for (hotels.Start(); hotels.Exists(); hotels.Next())
                {
                    Hotel hotel = hotels.Get();
                    if (hotel == tourist)
                    {
                        tourist.Sum = hotel*tourist;
                        break;
                    }
                }
            }
        }
        /// <summary>
        /// Finds the longest night in the hotels
        /// </summary>
        /// <param name="tourists"></param>
        /// <returns></returns>
        public static int FindLongestNight(TouristLinkedList tourists)
        {
            int longestStay = 0;
            for (tourists.Start(); tourists.Exists(); tourists.Next())
            {
                Tourist tourist = tourists.Get();
                if (tourist.BookedNights > longestStay)
                {
                    longestStay = tourist.BookedNights;
                }
            }
            return longestStay;
        }
        /// <summary>
        /// Finds the tourist with the longest stay
        /// </summary>
        /// <param name="BaseTourists"></param>
        /// <param name="asnwerList"></param>
        /// <param name="longestStay"></param>
        public static void FindTouristWithLongestStay(TouristLinkedList BaseTourists,
TouristLinkedList asnwerList, int longestStay)
        {
            for (BaseTourists.Start(); BaseTourists.Exists(); BaseTourists.Next())
            {
                Tourist tourist = BaseTourists.Get();
                if (tourist.BookedNights == longestStay)
                {
                    asnwerList.AddToEnd(tourist);
                }
            }
        }
        /// <summary>
        /// Finds the tourist with the smallest sum
        /// </summary>
        /// <param name="BaseTourists"></param>
        /// <param name="asnwerList"></param>
        /// <param name="Limit"></param>
        public static void FindMinSpenders(TouristLinkedList BaseTourists, TouristLinkedList
asnwerList, int Limit)
        {
            for (BaseTourists.Start(); BaseTourists.Exists(); BaseTourists.Next())
            {
                Tourist tourist = BaseTourists.Get();
                if (tourist.Sum < Limit && tourist.Sum != 0)
                {
                    asnwerList.AddToEnd(tourist);
                }
            }
        }

        public static void FindUsedUnused(TouristLinkedList BaseTourists, HotelLinkedList
original, HotelLinkedList used, HotelLinkedList unused)
        {
            for (original.Start(); original.Exists(); original.Next())
            {
                Hotel hotel = original.Get();
                bool isUsed = false;

                for (BaseTourists.Start(); BaseTourists.Exists(); BaseTourists.Next())
                {
                    Tourist tourist = BaseTourists.Get();
                    if (hotel == tourist)
                    {
                        isUsed = true;
                        break;
                    }
                }

                if (isUsed)
                {
                    used.AddToEnd(hotel);
                }
                else
                {
                    unused.AddToEnd(hotel);
                }
            }
        }
    }
}
namespace L2
{
    /// <summary>
    /// Class for the tourist object
    /// </summary>
    class Tourist
    {
        public string LastName { get; set; }
        public string FirstName { get; set; }
        public string HotelName { get; set; }
        public string RoomType { get; set; }
        public int BookedNights { get; set; }

        public int Sum { get; set; } = 0;
        /// <summary>
        /// Constructor for the tourist object
        /// </summary>
        /// <param name="lastName"></param>
        /// <param name="firstName"></param>
        /// <param name="hotelName"></param>
        /// <param name="roomType"></param>
        /// <param name="bookedNights"></param>
        public Tourist(string lastName, string firstName, string hotelName, string roomType,
int bookedNights)
        {
            LastName = lastName;
            FirstName = firstName;
            HotelName = hotelName;
            RoomType = roomType;
            BookedNights = bookedNights;
        }
        /// <summary>
        /// Overriden ToString method for the tourist object
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            string line = $"{LastName,-15} | {FirstName,-15} | {HotelName,-10} | {RoomType,-
15} | {BookedNights,22}|";
            return line;
        }
        /// <summary>
        /// Compares two tourist objects by their last name and first name
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public int CompareTo(Tourist other)
        {
            if(this.LastName.CompareTo(other.LastName) == 0)
            {
                return this.FirstName.CompareTo(other.FirstName);
            }
            return this.LastName.CompareTo(other.LastName);
        }
        /// <summary>
        /// Compares a hotel and tourist object by their hotel name and room type
        /// </summary>
        /// <param name="hotel"></param>
        /// <param name="tourist"></param>
        /// <returns></returns>
        public static bool operator ==(Hotel hotel, Tourist tourist)
        {
            return hotel.HotelName == tourist.HotelName && hotel.HotelType ==
tourist.RoomType;
        }
        /// <summary>
        /// Compares a hotel and tourist object by their hotel name and room type
        /// </summary>
        /// <param name="hotel"></param>
        /// <param name="tourist"></param>
        /// <returns></returns>
        public static bool operator !=(Hotel hotel, Tourist tourist)
        {
            return !(hotel == tourist);
        }

        /// <summary>
        /// Operator to calculate the total cost of the tourist's stay
        /// </summary>
        /// <param name="hotel"></param>
        /// <param name="tourist"></param>
        /// <returns></returns>
        public static int operator *(Hotel hotel, Tourist tourist)
        {
            return tourist.BookedNights * hotel.Cost;
        }
        /// <summary>
        /// Equals override
        /// </summary>
        /// <param name="obj"></param>
        /// <returns></returns>
        public override bool Equals(object obj)
        {
            return obj is Tourist tourist &&
                   LastName == tourist.LastName &&
                   FirstName == tourist.FirstName &&
                   HotelName == tourist.HotelName &&
                   RoomType == tourist.RoomType &&
                   BookedNights == tourist.BookedNights &&
                   Sum == tourist.Sum;
        }
        /// <summary>
        /// GetHashCode override
        /// </summary>
        /// <returns></returns>
        public override int GetHashCode()
        {
            int hashCode = 1304080926;
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(LastName);
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(FirstName);
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(HotelName);
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(RoomType);
            hashCode = hashCode * -1521134295 + BookedNights.GetHashCode();
            hashCode = hashCode * -1521134295 + Sum.GetHashCode();
            return hashCode;
        }
    }
}
```

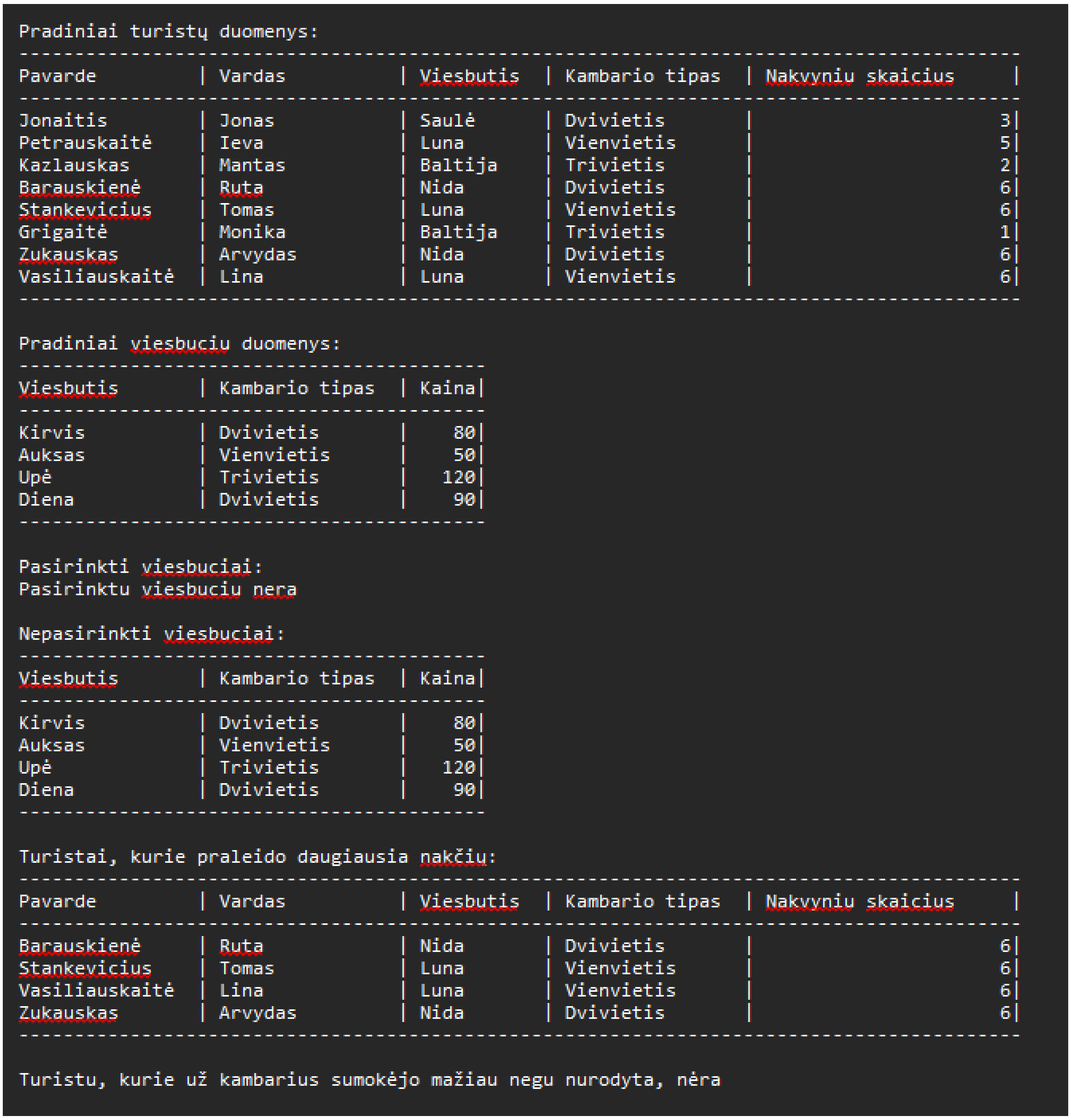
```
namespace L2
{
    /// <summary>
    /// Class for the tourist linked list
    /// </summary>
    class TouristLinkedList
    {
        /// <summary>
        /// Class for the tourist node
        /// </summary>
        private sealed class TouristNode
        {
            public Tourist Data { get; set; }
            public TouristNode Link { get; set; }
            /// <summary>
            /// Constructor for the tourist node
            /// </summary>
            /// <param name="data"></param>
            /// <param name="link"></param>
            public TouristNode(Tourist data, TouristNode link)
            {
                Data = data;
                Link = link;
            }
        }
        public int Count { get; private set; } = 0;

        private TouristNode head; //start
        private TouristNode tail; //end
        private TouristNode extraTail; //end (extra)
        private TouristNode ListReader; // list linker
        /// <summary>
        /// Constructor for the tourist linked list
        /// </summary>
        public TouristLinkedList()
        {
            this.tail = new TouristNode(null, null);
            this.head = new TouristNode(null, this.tail);
            this.extraTail = head;
            this.ListReader = null;
        }

        /// <summary>
        /// Method to add a tourist to the end of the linked list
        /// </summary>
        /// <param name="data"></param>
        public void AddToEnd(Tourist data)
        {
            extraTail.Link = new TouristNode(data, null);
            extraTail = extraTail.Link;
            Count++;
        }
        /// <summary>
        /// Method to get the tourist from the linked list
        /// </summary>
        /// <returns></returns>
        public Tourist Get()
        {
            return ListReader.Data;
        }
        /// <summary>
        /// Method for the next tourist in the linked list
        /// </summary>
        public void Next()
        {
            ListReader = ListReader.Link;
        }
        /// <summary>
        /// Method to check if the linked list exists
        /// </summary>
        /// <returns></returns>
        public bool Exists()
        {
            return ListReader != null && ListReader.Data != null;
        }

        /// <summary>
        /// Method for the start of the linked list
        /// </summary>
        public void Start()
        {
            ListReader = head.Link;
        }
        /// <summary>
        /// Method for the end of the linked list
        /// </summary>
        public void End()
        {
            ListReader = tail.Link;
        }
        /// <summary>
        /// Bubble sort for the linked list using the operator override
        /// </summary>
        public void BubbleSort()
        {
            if (head.Link == null || head.Link.Link == null)
                return;

            bool flag = true;
            while (flag)
            {
                flag = false;
                TouristNode d = head.Link;
                TouristNode prev = null;

                while (d.Link != null)
                {
                    if (d.Data.CompareTo(d.Link.Data) > 0)
                    {
                        Tourist temp = d.Data;
                        d.Data = d.Link.Data;
                        d.Link.Data = temp;

                        flag = true;
                    }
                    prev = d;
                    d = d.Link;
                }
            }
        }
    }
}

body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
}

.container {
    width: 80%;
    margin: 20px auto;
    padding: 20px;
    background-color: white;
    border: 1px solid #ccc;
    height: 1000px;
```

```
}

.label {
    font-weight: bold;
}

.table {
    background-color: white;
    border-color: black;
    border-style: solid;
    border-width: 1px;
    color: black;
    border-collapse: collapse;
}

    .table td, .table th {
        border: 1px solid black;
        padding: 5px;
    }

.textbox {
    padding: 5px;
    border: 1px solid #ccc;
}

.button {
    padding: 10px 20px;
    background-color: #007bff;
    color: white;
    border: none;
    cursor: pointer;
}

    .button:hover {
        background-color: #0056b3;
    }

.custom-validator {
    color: red;
}
namespace L2
{
    public partial class WebForm : System.Web.UI.Page
    {
        /// <summary>
        /// Method to validate text box input for a positive integer
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator1_ServerValidate(object source, ServerValidateEventArgs
args)
        {
            int N;
            args.IsValid = int.TryParse(TextBox1.Text, out N) && N > 0 &&
!String.IsNullOrEmpty(TextBox1.Text);

        }
        /// <summary>
        /// updates the table with the tourist data
        /// </summary>
        /// <param name="tourists"></param>
        private void UpdateTable(TouristLinkedList tourists)
        {

            Table1.Rows.Clear();


            TableHeaderRow headerRow = new TableHeaderRow();
            headerRow.Cells.Add(new TableHeaderCell { Text = "Pavarde" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Vardas" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Nakvynių skaičius" });
            Table1.Rows.Add(headerRow);


            for (tourists.Start(); tourists.Exists(); tourists.Next())
            {
                Tourist tourist = tourists.Get();
                TableRow row = new TableRow();
                row.Cells.Add(new TableCell { Text = tourist.LastName });
                row.Cells.Add(new TableCell { Text = tourist.FirstName });
                row.Cells.Add(new TableCell { Text = tourist.HotelName });
                row.Cells.Add(new TableCell { Text = tourist.RoomType });
                row.Cells.Add(new TableCell { Text = tourist.BookedNights.ToString() });
                Table1.Rows.Add(row);
            }
        }
        /// <summary>
        /// Method to update the result table with the tourist data
        /// </summary>
        /// <param name="tourists"></param>
        private void UpdateResultTable(TouristLinkedList tourists)
        {

            Table3.Rows.Clear();


            TableHeaderRow headerRow = new TableHeaderRow();
            headerRow.Cells.Add(new TableHeaderCell { Text = "Pavarde" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Vardas" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Nakvynių skaičius" });
            Table3.Rows.Add(headerRow);


            for (tourists.Start(); tourists.Exists(); tourists.Next())
            {
                Tourist tourist = tourists.Get();
                TableRow row = new TableRow();
                row.Cells.Add(new TableCell { Text = tourist.LastName });
                row.Cells.Add(new TableCell { Text = tourist.FirstName });
                row.Cells.Add(new TableCell { Text = tourist.HotelName });
                row.Cells.Add(new TableCell { Text = tourist.RoomType });
                row.Cells.Add(new TableCell { Text = tourist.BookedNights.ToString() });
                Table3.Rows.Add(row);
            }
        }
        /// <summary>
        /// Method to update the hotel table with the hotel data
        /// </summary>
        /// <param name="hotels"></param>
        private void UpdateHotelTable(HotelLinkedList hotels)
        {

            Table2.Rows.Clear();


            TableHeaderRow headerRow = new TableHeaderRow();
            headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Kaina" });
            Table2.Rows.Add(headerRow);
```

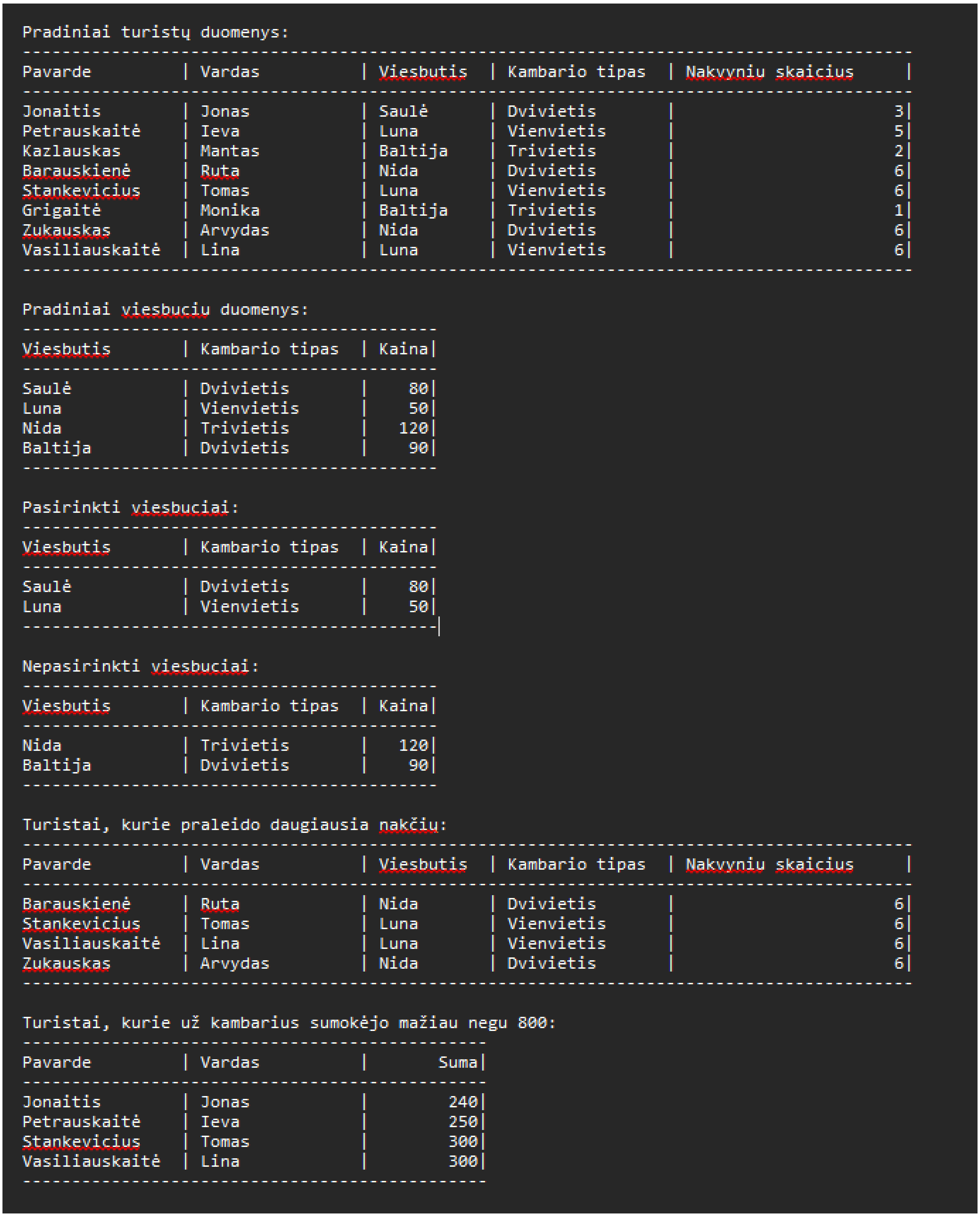
```
            for (hotels.Start(); hotels.Exists(); hotels.Next())
            {
                Hotel hotel = hotels.Get();
                TableRow row = new TableRow();
                row.Cells.Add(new TableCell { Text = hotel.HotelName });
                row.Cells.Add(new TableCell { Text = hotel.HotelType });
                row.Cells.Add(new TableCell { Text = hotel.Cost.ToString() });
                Table2.Rows.Add(row);
            }
        }
        /// <summary>
        /// Method to update the table with the tourists who spent less than the specified 
amount
        /// </summary>
        /// <param name="tourists"></param>
        private void UpdateMinTuristTable(TouristLinkedList tourists)
        {

            Table4.Rows.Clear();

            if (tourists.Count != 0 )
            {
                TableHeaderRow headerRow = new TableHeaderRow();
                headerRow.Cells.Add(new TableHeaderCell { Text = "Pavarde" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Vardas" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Suma" });
                Table4.Rows.Add(headerRow);


                for (tourists.Start(); tourists.Exists(); tourists.Next())
                {
                    Tourist tourist = tourists.Get();
                    TableRow row = new TableRow();
                    row.Cells.Add(new TableCell { Text = tourist.LastName });
                    row.Cells.Add(new TableCell { Text = tourist.FirstName });
                    row.Cells.Add(new TableCell { Text = tourist.Sum.ToString() });
                    Table4.Rows.Add(row);
                }
            }
            else
            {
                TableRow row = new TableRow();
                row.Cells.Add(new TableCell { Text = "Turistų, kurie išleido mažiau negu 
nurodyta suma nėra" });
                Table4.Rows.Add(row);
            }
        }
        private void UpdateHotelTableTwo(HotelLinkedList hotels)
        {

            if (hotels.Count != 0)
            {
                Table5.Rows.Clear();


                TableHeaderRow headerRow = new TableHeaderRow();
                headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Kaina" });
                Table5.Rows.Add(headerRow);


                for (hotels.Start(); hotels.Exists(); hotels.Next())
                {
                    Hotel hotel = hotels.Get();
                    TableRow row = new TableRow();
                    row.Cells.Add(new TableCell { Text = hotel.HotelName });
                    row.Cells.Add(new TableCell { Text = hotel.HotelType });
                    row.Cells.Add(new TableCell { Text = hotel.Cost.ToString() });
                    Table5.Rows.Add(row);
                }
            }

            else
            {
                TableRow row = new TableRow();
                row.Cells.Add(new TableCell { Text = "Nera" });
                Table5.Rows.Add(row);
            }
        }
        private void UpdateHotelTableThree(HotelLinkedList hotels)
        {
            if (hotels.Count != 0)
            {
                Table6.Rows.Clear();


                TableHeaderRow headerRow = new TableHeaderRow();
                headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Kaina" });
                Table6.Rows.Add(headerRow);


                for (hotels.Start(); hotels.Exists(); hotels.Next())
                {
                    Hotel hotel = hotels.Get();
                    TableRow row = new TableRow();
                    row.Cells.Add(new TableCell { Text = hotel.HotelName });
                    row.Cells.Add(new TableCell { Text = hotel.HotelType });
                    row.Cells.Add(new TableCell { Text = hotel.Cost.ToString() });
                    Table6.Rows.Add(row);
                }
            }

            else
            {
                TableRow row = new TableRow();
                row.Cells.Add(new TableCell { Text = "nera" });
                Table6.Rows.Add(row);
            }
        }
    }
}
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm.aspx.cs"
Inherits="L2.WebForm" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
    <form id="form1" runat="server">
        <div class="container">
            <asp:Label ID="Label2" runat="server" Text="Turistų informacija:"
CssClass="label"></asp:Label>
            <br />
            <asp:Table ID="Table1" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label3" runat="server" Text="Viešbučių informacija:"
CssClass="label"></asp:Label>
            <br />
            <asp:Table ID="Table2" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label1" runat="server" Text="Maksimali pinigų suma:" 
CssClass="label"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox1" runat="server" CssClass="textbox"></asp:TextBox>
            <asp:CustomValidator ID="CustomValidator1" runat="server" 
ControlToValidate="TextBox1" ErrorMessage="Tik realieji skaičiai" 
OnServerValidate="CustomValidator1_ServerValidate" 
ValidateEmptyText="True"></asp:CustomValidator>
            <br />
            <br />
            <asp:Button ID="Button1" runat="server" Text="Skaičiuoti" CssClass="button" 
OnClick="Button1_Click" />
            <br />
            <br />
            <asp:Table ID="Table5" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Table ID="Table6" runat="server" CssClass="table"></asp:Table>
            <br />
            <br />
            <asp:Table ID="Table3" runat="server" CssClass="table"></asp:Table>
            <br />
            <br />
            <asp:Table ID="Table4" runat="server" CssClass="table"></asp:Table>
        </div>
    </form>
</body>
</html>
```

```
namespace L2
{
    public partial class WebForm : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        /// <summary>
        /// Method for the button click event
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        protected void Button1_Click(object sender, EventArgs e)
        {
            if(Page.IsValid)
            {
                string TouristFile = Server.MapPath("App_Data/U17a.txt");
                string HotelFile = Server.MapPath("App_Data/U17b.txt");
                string Result = Server.MapPath("App_Data/Rezultatai.txt");
                File.Delete(Result);
                int Limit = int.Parse(TextBox1.Text);

                TouristLinkedList Tourists = new TouristLinkedList();
                HotelLinkedList Hotels = new HotelLinkedList();

                InOutUtils.ReadTourists(TouristFile, Tourists);
                InOutUtils.ReadHotels(HotelFile, Hotels);

                string headerOne = $"{"Pavarde",-15} | {"Vardas",-15} | {"Viesbutis",-10} | 
{"Kambario tipas",-15} | {"Nakvyniu skaicius",-22}|";
                string headerTwo = $"{"Viesbutis",-15} | {"Kambario tipas",-15} | {"Kaina",-
5}|";

                InOutUtils.PrintTourists(Result,Tourists, headerOne,false);
                InOutUtils.PrintHotels(Result,Hotels, headerTwo);


                HotelLinkedList usedHotel = new HotelLinkedList();
                HotelLinkedList unusedHotel = new HotelLinkedList();
                TaskUtils.FindUsedUnused(Tourists, Hotels, usedHotel, unusedHotel);


                int longestStay = TaskUtils.FindLongestNight(Tourists);

                InOutUtils.PrintHotelsAnswer(Result, usedHotel, headerTwo, true);
                InOutUtils.PrintHotelsAnswer(Result, unusedHotel, headerTwo, false);

                TaskUtils.FindSumOfNightsInHotels(Tourists, Hotels);
                TouristLinkedList TouristWithLongestStay = new TouristLinkedList();
                TaskUtils.FindTouristWithLongestStay(Tourists, TouristWithLongestStay,
longestStay);

                TouristWithLongestStay.BubbleSort();
                InOutUtils.PrintTourists(Result, TouristWithLongestStay, headerOne,true);

                string HeaderThree = $"{"Pavarde",-15} | {"Vardas",-15} | {"Suma",10}|";
                TouristLinkedList MinSpenders = new TouristLinkedList();
                TaskUtils.FindMinSpenders(Tourists, MinSpenders, Limit);
                MinSpenders.BubbleSort();
                InOutUtils.PrintMinTurists(Result, MinSpenders, HeaderThree, Limit);

                UpdateTable(Tourists);
                UpdateHotelTable(Hotels);
                UpdateHotelTableTwo(usedHotel);
                UpdateHotelTableThree(unusedHotel);
                UpdateResultTable(TouristWithLongestStay);
                UpdateMinTuristTable(MinSpenders);

            }
        }
    }
}
```

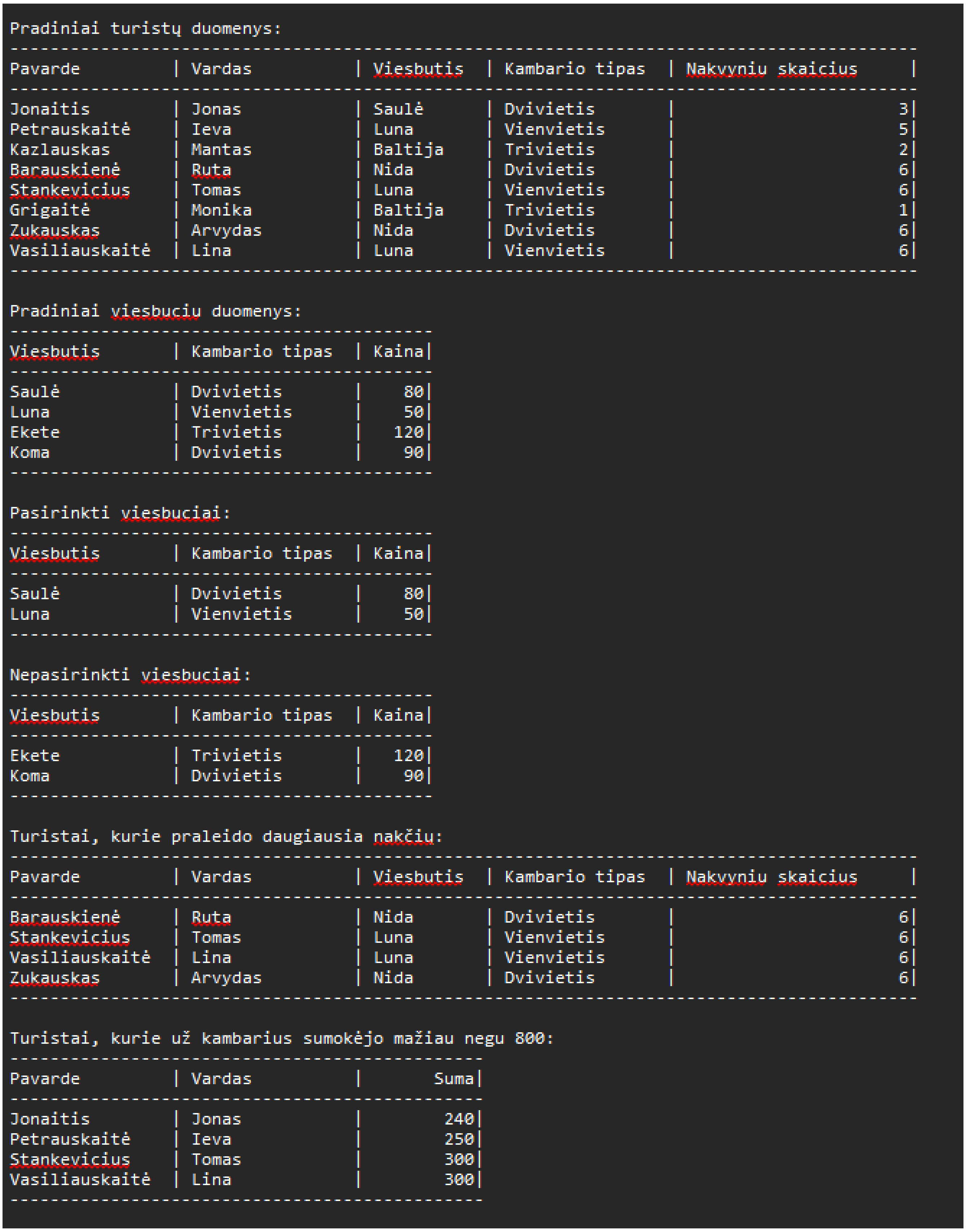
2.7. Pradiniai duomenys ir rezultatai



```
Pradiniai turistų duomenys:
------------------------------------------------------------------------------------------
Pavarde         | Vardas           | Viesbutis  | Kambario tipas   | Nakvyniu skaicius     |
------------------------------------------------------------------------------------------
Jonaitis        | Jonas           | Saulė      | Dvivietis       |                      3|
Petrauskaitė    | Ieva            | Luna       | Vienvietis      |                      5|
Kazlauskas      | Mantas          | Baltija    | Trivietis       |                      2|
Barauskienė     | Ruta            | Nida       | Dvivietis       |                      6|
Stankevicius    | Tomas           | Luna       | Vienvietis      |                      6|
Grigaitė        | Monika          | Baltija    | Trivietis       |                      1|
Zukauskas       | Arvydas         | Nida       | Dvivietis       |                      6|
Vasiliauskaitė  | Lina            | Luna       | Vienvietis      |                      6|
------------------------------------------------------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis       | Kambario tipas   | Kaina|
------------------------------------------
Kirvis          | Dvivietis       |    80|
Auksas          | Vienvietis      |    50|
Upė             | Trivietis       |   120|
Diena           | Dvivietis       |    90|
------------------------------------------

Pasirinkti viesbuciai:
Pasirinktu viesbuciu nera

Nepasirinkti viesbuciai:
------------------------------------------
Viesbutis       | Kambario tipas   | Kaina|
------------------------------------------
Kirvis          | Dvivietis       |    80|
Auksas          | Vienvietis      |    50|
Upė             | Trivietis       |   120|
Diena           | Dvivietis       |    90|
------------------------------------------

Turistai, kurie praleido daugiausia nakčių:
------------------------------------------------------------------------------------------
Pavarde         | Vardas           | Viesbutis  | Kambario tipas   | Nakvyniu skaicius     |
------------------------------------------------------------------------------------------
Barauskienė     | Ruta            | Nida       | Dvivietis       |                      6|
Stankevicius    | Tomas           | Luna       | Vienvietis      |                      6|
Vasiliauskaitė  | Lina            | Luna       | Vienvietis      |                      6|
Zukauskas       | Arvydas         | Nida       | Dvivietis       |                      6|
------------------------------------------------------------------------------------------

Turistu, kurie už kambarius sumokėjo mažiau negu nurodyta, nėra
```



```
Pradiniai turistų duomenys:
-----------------------------------------------------------------------------------------
Pavarde          | Vardas           | Viesbutis  | Kambario tipas   | Nakvyniu skaicius     |
-----------------------------------------------------------------------------------------
Jonaitis         | Jonas            | Saulė      | Dvivietis        |                      3|
Petrauskaitė     | Ieva             | Luna       | Vienvietis       |                      5|
Kazlauskas       | Mantas           | Baltija    | Trivietis        |                      2|
Barauskienė      | Ruta             | Nida       | Dvivietis        |                      6|
Stankevicius     | Tomas            | Luna       | Vienvietis       |                      6|
Grigaitė         | Monika           | Baltija    | Trivietis        |                      1|
Zukauskas        | Arvydas          | Nida       | Dvivietis        |                      6|
Vasiliauskaitė   | Lina             | Luna       | Vienvietis       |                      6|
-----------------------------------------------------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis        | Kambario tipas   | Kaina|
------------------------------------------
Saulė            | Dvivietis        |    80|
Luna             | Vienvietis       |    50|
Nida             | Trivietis        |   120|
Baltija          | Dvivietis        |    90|
------------------------------------------

Pasirinkti viesbuciai:
------------------------------------------
Viesbutis        | Kambario tipas   | Kaina|
------------------------------------------
Saulė            | Dvivietis        |    80|
Luna             | Vienvietis       |    50|
------------------------------------------|

Nepasirinkti viesbuciai:
------------------------------------------
Viesbutis        | Kambario tipas   | Kaina|
------------------------------------------
Nida             | Trivietis        |   120|
Baltija          | Dvivietis        |    90|
------------------------------------------

Turistai, kurie praleido daugiausia nakčių:
-----------------------------------------------------------------------------------------
Pavarde          | Vardas           | Viesbutis  | Kambario tipas   | Nakvyniu skaicius     |
-----------------------------------------------------------------------------------------
Barauskienė      | Ruta             | Nida       | Dvivietis        |                      6|
Stankevicius     | Tomas            | Luna       | Vienvietis       |                      6|
Vasiliauskaitė   | Lina             | Luna       | Vienvietis       |                      6|
Zukauskas        | Arvydas          | Nida       | Dvivietis        |                      6|
-----------------------------------------------------------------------------------------

Turistai, kurie už kambarius sumokėjo mažiau negu 800:
-----------------------------------------------
Pavarde          | Vardas           |      Suma|
-----------------------------------------------
Jonaitis         | Jonas            |       240|
Petrauskaitė     | Ieva             |       250|
Stankevicius     | Tomas            |       300|
Vasiliauskaitė   | Lina             |       300|
-----------------------------------------------
```



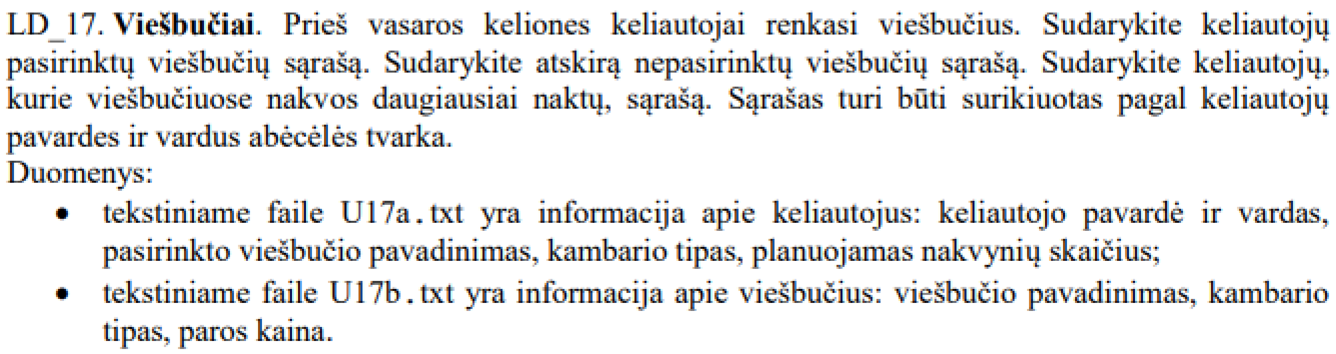
```
Pradiniai turistų duomenys:
------------------------------------------------------------------------------------------
Pavarde          | Vardas           | Viesbutis  | Kambario tipas   | Nakvyniu skaicius      |
------------------------------------------------------------------------------------------
Jonaitis         | Jonas            | Saulė      | Dvivietis        |                       3|
Petrauskaitė     | Ieva             | Luna       | Vienvietis       |                       5|
Kazlauskas       | Mantas           | Baltija    | Trivietis        |                       2|
Barauskienė      | Ruta             | Nida       | Dvivietis        |                       6|
Stankevicius     | Tomas            | Luna       | Vienvietis       |                       6|
Grigaitė         | Monika           | Baltija    | Trivietis        |                       1|
Zukauskas        | Arvydas          | Nida       | Dvivietis        |                       6|
Vasiliauskaitė   | Lina             | Luna       | Vienvietis       |                       6|
------------------------------------------------------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis        | Kambario tipas   | Kaina|
------------------------------------------
Saulė            | Dvivietis        |    80|
Luna             | Vienvietis       |    50|
Ekete            | Trivietis        |   120|
Koma             | Dvivietis        |    90|
------------------------------------------

Pasirinkti viesbuciai:
------------------------------------------
Viesbutis        | Kambario tipas   | Kaina|
------------------------------------------
Saulė            | Dvivietis        |    80|
Luna             | Vienvietis       |    50|
------------------------------------------

Nepasirinkti viesbuciai:
------------------------------------------
Viesbutis        | Kambario tipas   | Kaina|
------------------------------------------
Ekete            | Trivietis        |   120|
Koma             | Dvivietis        |    90|
------------------------------------------

Turistai, kurie praleido daugiausia nakčių:
------------------------------------------------------------------------------------------
Pavarde          | Vardas           | Viesbutis  | Kambario tipas   | Nakvyniu skaicius      |
------------------------------------------------------------------------------------------
Barauskienė      | Ruta             | Nida       | Dvivietis        |                       6|
Stankevicius     | Tomas            | Luna       | Vienvietis       |                       6|
Vasiliauskaitė   | Lina             | Luna       | Vienvietis       |                       6|
Zukauskas        | Arvydas          | Nida       | Dvivietis        |                       6|
------------------------------------------------------------------------------------------

Turistai, kurie už kambarius sumokėjo mažiau negu 800:
-----------------------------------------------
Pavarde          | Vardas           |      Suma|
-----------------------------------------------
Jonaitis         | Jonas            |       240|
Petrauskaitė     | Ieva             |       250|
Stankevicius     | Tomas            |       300|
Vasiliauskaitė   | Lina             |       300|
-----------------------------------------------
```

2.8. Dėstytojo pastabos

Truko dviejų sarašų uždavinyje, pubo ištaisyta ir pridėta.

3. Bendrinės klasės ir testavimas (L3)

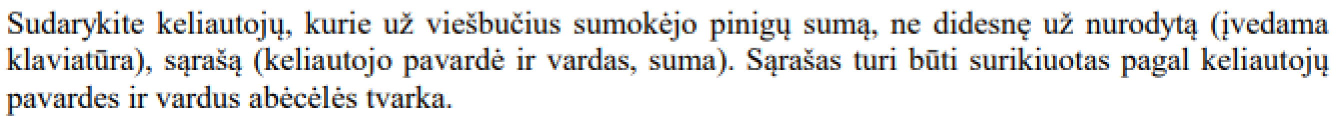
3.1. Darbo užduotis



LD_17. **Viešbučiai**. Prieš vasaros keliones keliautojai renkasi viešbučius. Sudarykite keliautojų pasirinktų viešbučių sąrašą. Sudarykite atskirą nepasirinktų viešbučių sąrašą. Sudarykite keliautojų, kurie viešbučiuose nakvos daugiausiai naktų, sąrašą. Sąrašas turi būti surikiuotas pagal keliautojų pavardes ir vardus abėcėlės tvarka.

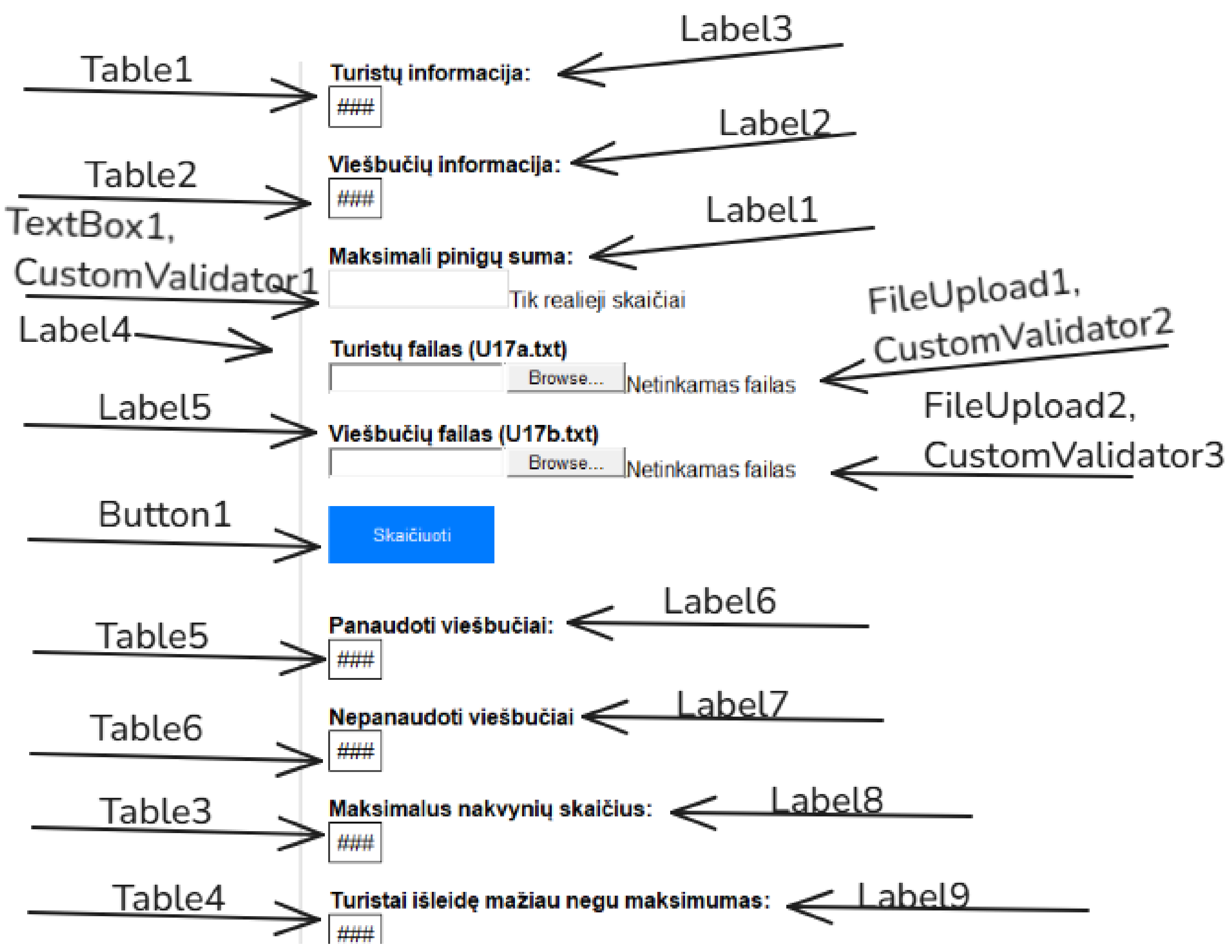
Duomenys:

- tekstiniame faile U17a.txt yra informacija apie keliautojus: keliautojo pavardė ir vardas, pasirinkto viešbučio pavadinimas, kambario tipas, planuojamas nakvynių skaičius;
- tekstiniame faile U17b.txt yra informacija apie viešbučius: viešbučio pavadinimas, kambario tipas, paros kaina.



Sudarykite keliautojų, kurie už viešbučius sumokėjo pinigų sumą, ne didesnę už nurodytą (įvedama klaviatūra), sąrašą (keliautojo pavardė ir vardas, suma). Sąrašas turi būti surikiuotas pagal keliautojų pavardes ir vardus abėcėlės tvarka.

3.2. Grafinės naudotojo sąsajos schema

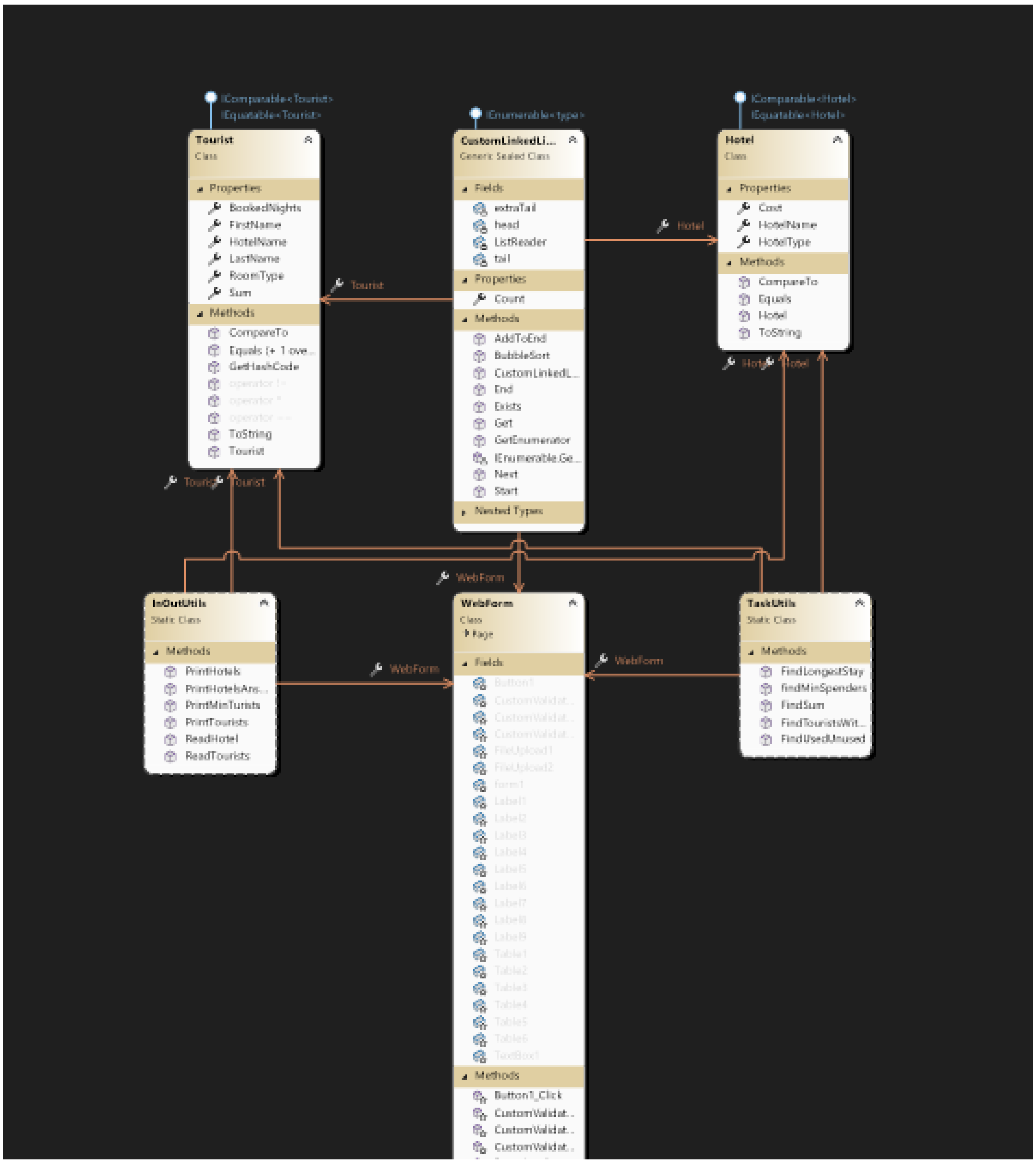


3.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Label(Bendri pakeitimai)	CssClass	label
Label(Bendri pakeitimai)	Font-weight	bold
Table(Bendri pakeitimai)	CssClass	table
Table(Bendri pakeitimai)	Backround-color	White
Table(Bendri pakeitimai)	Border-color	Black
Table(Bendri pakeitimai)	Border-style	Solid
Table(Bendri pakeitimai)	Border-width	1px
Table(Bendri pakeitimai)	Color	Black
Table(Bendri pakeitimai)	Border-collapse	Collapse
Label1	Text	Maksimali pinigų suma:
Label2	Text	Turistų informacija:
Label3	Text	Viešbučių informacija:
Label4	Text	Turistų failas (U17a.txt)
Label5	Text	Viešbučių failas (U17b.txt)
Label6	Text	Panaudoti viešbučiai:
Label7	Text	Nepanaudoti viešbučiai
Label8	Text	Maksimalus nakvynių skaičius:
Label9	Text	Turistai išleidę mažiau negu maksimumas:
CustomValidator1	Text	Tik Realieji skaičiai
CustomValidator1	ControlToValidate	TextBox1
CustomValidator1	Color	Red
CustomValidator1	ValidateEmptyText	True
CustomValidator2	Text	Netinkamas failas
CustomValidator2	ControlToValidate	FileUpload1
CustomValidator2	Color	Red
CustomValidator2	ValidateEmptyText	True
CustomValidator3	Text	Netinkamas failas
CustomValidator3	ControlToValidate	FileUpload2
CustomValidator3	Color	Red
CustomValidator3	ValidateEmptyText	True
FileUpload1		
FileUpload2		
TextBox1	Padding	5px
TextBox1	Border	1px solid
Button1	Padding	10px 20px
Button1	Backround-color	#007bff
Button1	Color	White
Button1	Border	None

Button1	Cursor	Pointer
Button1:Hover	Backround-color	#0056b3

3.4. Klasių diagrama



3.5. Programos naudotojo vadovas

Vartotojas turi sukurti duomenų failus viešbučiam „U17b.txt “ir turistam „U17a.txt“. Viešbučių duomenų faile duomenys išdėstomi kiekvienoje naujoje eilutėje tokia eiga: Pavardė;Vardas;Viešbučio,kuriame apsistojo pavadinimas;norimas kambario tipas;kiek nakčių praleido(skaičius). Viešbučių informacija tokia eiga: Pavadinimas;Kambarių tipas;nakties kaina(skaičius). Programa yra paleidžiama, duomenys yra įkeliami puslapyje, pasirinkus teisingas įkėlimo vietas, teksto įvedimo lange įvedamas išlaidų limitas, rezultatai matomi lange ir yra atspausdinami į App_Data/Rezultatai.txt“

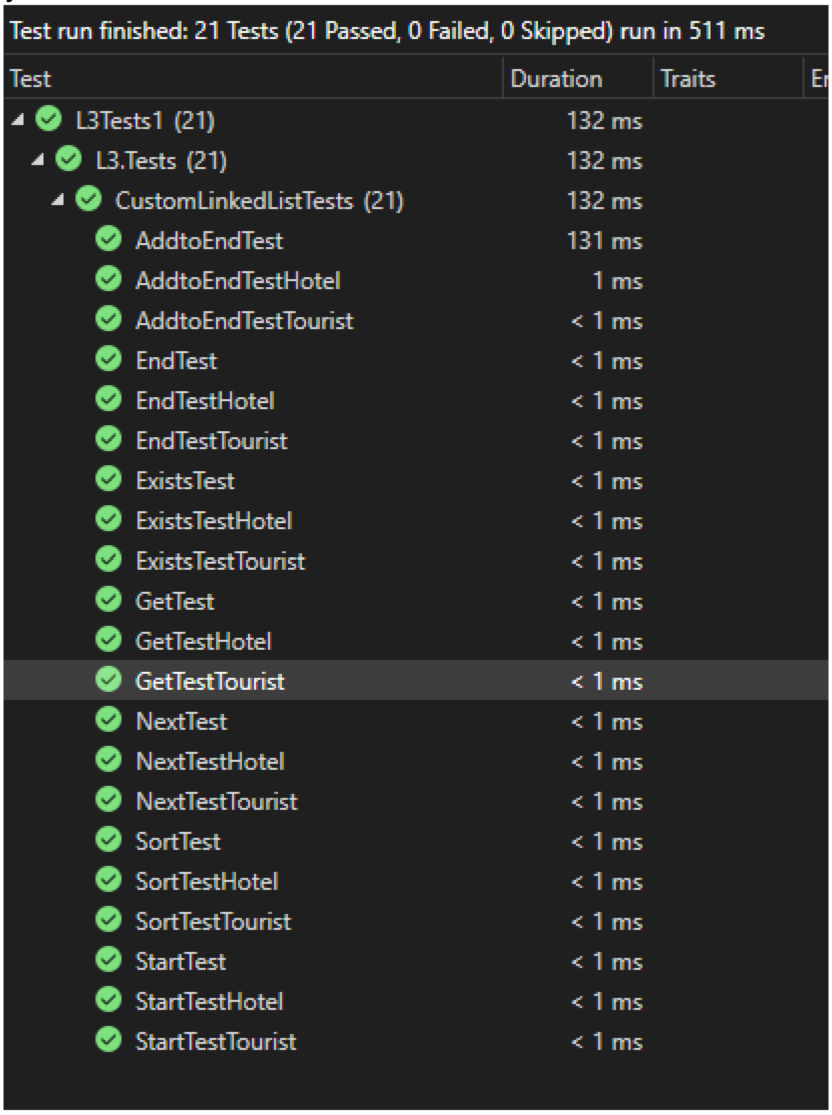
3.6. Programos tekstas

```
namespace L3
{
    /// <summary>
    /// Base class for the hotel object
    /// </summary>
    public class Hotel : IComparable<Hotel>, IEquatable<Hotel>
    {
        public string HotelName { get; set; }
        public string HotelType { get; set; }
        public int Cost { get; set; }
        /// <summary>
        /// Constructor for the hotel object
        /// </summary>
        /// <param name="hotelName"></param>
        /// <param name="hotelType"></param>
        /// <param name="cost"></param>
        public Hotel(string hotelName, string hotelType, int cost)
        {
            HotelName = hotelName;
            HotelType = hotelType;
            Cost = cost;
        }
        /// <summary>
        /// Overriden ToString method for the hotel object
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            string line = $"{HotelName,-15} | {HotelType,-15} | {Cost,5}|";
            return line;
        }

        /// <summary>
        /// Implementation of IEquatable<Hotel>.Equals
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public bool Equals(Hotel other)
        {
            if (other == null) return false;
            return HotelName == other.HotelName && HotelType == other.HotelType && Cost ==
other.Cost;
        }

        /// <summary>
        /// Implementation of IComparable<Hotel>.CompareTo
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public int CompareTo(Hotel other)
        {
            if(HotelName.CompareTo(other.HotelName)==0)
            {
                if (HotelType.CompareTo(other.HotelType) == 0)
                {
                    return Cost.CompareTo(other.Cost);
                }
                return HotelType.CompareTo(other.HotelType);
            }
            return HotelName.CompareTo(other.HotelName);
        }
        /// <summary>
        /// Makes sure the objects are equal by comparing their properties
        /// </summary>
        /// <param name="obj"></param>
        /// <returns></returns>
        public override bool Equals(object obj)
        {
            return obj is Hotel hotel &&
                   HotelName == hotel.HotelName &&
                   HotelType == hotel.HotelType &&
                   Cost == hotel.Cost;
        }
        /// <summary>
        /// Converts the hotel object to a table row
        /// </summary>
        /// <returns></returns>
        public TableRow ToTableRow()
        {
            TableRow row = new TableRow();
            row.Cells.Add(new TableCell { Text = HotelName });
            row.Cells.Add(new TableCell { Text = HotelType });
            row.Cells.Add(new TableCell { Text = Cost.ToString() });
            return row;
        }
    }
}

namespace L3
{
      public static class InOutUtils
    {
        public static WebForm WebForm
        {
            get => default;
            set
            {
            }
        }

        public static Tourist Tourist
        {
            get => default;
            set
            {
            }
        }

        public static Hotel Hotel
        {
            get => default;
            set
            {
            }
        }
        /// <summary>
        /// Reads tourist data from the file
        /// </summary>
        /// <param name="fileName"></param>
        /// <param name="Tourists"></param>
        public static void ReadTourists(string fileName, CustomLinkedList<Tourist> Tourists)
             {
                    string[] lines = File.ReadAllLines(fileName);

                    foreach (string line in lines)
                    {
                          string[] data = line.Split(';');

                          Tourists.AddToEnd(new Tourist(data[0], data[1], data[2], data[3],
int.Parse(data[4])));
                    }
             }
        /// <summary>
        /// Reads hotel data from the file
        /// </summary>
        /// <param name="fileName"></param>
        /// <param name="Hotels"></param>
        public static void ReadHotel(string fileName, CustomLinkedList<Hotel> Hotels)
        {
            string[] lines = File.ReadAllLines(fileName);

            foreach (string line in lines)
            {
                string[] data = line.Split(';');

                Hotels.AddToEnd(new Hotel(data[0], data[1], int.Parse(data[2])));
            }
        }


        /// <summary>
        /// Method that prints everything to the file as needed
        /// </summary>
        /// <typeparam name="T"></typeparam>
        /// <param name="file"></param>
        /// <param name="header"></param>
        /// <param name="values"></param>
        /// <param name="answer"></param>
        /// <param name="data"></param>
        public static void Print<T>(string file, string header, string values, bool answer,
CustomLinkedList<T> data) where T : IComparable<T>, IEquatable<T>
        {
            using (StreamWriter writer = new StreamWriter(file, true))
            {
                if (data.Count != 0)
                {
                    writer.WriteLine(values);
                    string line = new string('-', header.Length);
                    writer.WriteLine(header);
                    writer.WriteLine(line);
                    foreach (T item in data)
                    {
                        if (answer && item is Tourist tourist)
                        {
                            string format = $"{tourist.LastName,-15} | {tourist.FirstName,-15}
| {tourist.Sum,10}|";
                            writer.WriteLine(format);
                        }
                        else
                        {
                            writer.WriteLine(item.ToString());
                        }
                    }
                    writer.WriteLine(line);
                    writer.WriteLine();
                }
                else
                {
                    writer.WriteLine(values);
                    writer.WriteLine("Tokio tipo duomenų nėra");
                }
            }
        }
    }
}

namespace L3
{
    public sealed class CustomLinkedList<type> : IEnumerable<type> where type :
IComparable<type>, IEquatable<type>
    {

        private sealed class Node<type>
        {
            public type Data { get; set; }
            public Node<type> Link { get; set; }

            public Node(type data, Node<type> link)
            {
                Data = data;
                Link = link;
            }
        }

        public int Count { get; private set; } = 0;

        public WebForm WebForm
        {
            get => default;
            set
            {
            }
        }

        public Tourist Tourist
        {
            get => default;
            set
            {
            }
        }

        public Hotel Hotel
        {
            get => default;
            set
            {
            }
        }

        private Node<type> head; //start
        private Node<type> tail; //end
        private Node<type> extraTail; //end (extra)
        private Node<type> ListReader; // list linker
        /// <summary>
        /// Constructor for the linked list
        /// </summary>
        public CustomLinkedList()
        {
            this.tail = new Node<type>(default, null);
            this.head = new Node<type>(default, this.tail);
            this.extraTail = head;
            this.ListReader = null;
        }
        /// <summary>
        /// Method to add data to the end of the linked list
        /// </summary>
        /// <param name="data"></param>
        public void AddToEnd(type data)
        {
            extraTail.Link = new Node<type>(data, null);
            extraTail = extraTail.Link;
            Count++;
        }
        /// <summary>
        /// Method to get data from the linked list
        /// </summary>
        /// <returns></returns>
        public type Get()
        {
            return ListReader.Data;
        }
        /// <summary>
        /// Method to move to the next node in the linked list
        /// </summary>
        public void Next()
        {
            ListReader = ListReader.Link;
        }
        /// <summary>
        /// Method to check if the data exists in the linked list
        /// </summary>
        /// <returns></returns>
        public bool Exists()
        {
            return ListReader != null && ListReader.Data != null;
        }
        /// <summary>
        /// Method for the start of the linked list
        /// </summary>
        public void Start()
        {
            ListReader = head.Link;
        }
        /// <summary>
        /// Method for the end of the linked list
        /// </summary>
        public void End()
        {
            ListReader = tail.Link;
        }
        /// <summary>
        /// returns data in the foreach loop
        /// </summary>
        /// <returns></returns>
        public IEnumerator<type> GetEnumerator()
        {
            for (Start(); Exists(); Next())
            {
                yield return ListReader.Data;
            }
        }
        /// <summary>
        /// Method to allow the linked list to be used in a foreach loop
        /// </summary>
        /// <returns></returns>

        IEnumerator IEnumerable.GetEnumerator()
        {
            return GetEnumerator();
        }
        /// <summary>
        /// Method to sort the linked list
        /// </summary>
        public void BubbleSort()
        {
            if (head.Link == null || head.Link.Link == null)
                return;

            bool flag = true;
            while (flag)
            {
                flag = false;
                Node<type> d = head.Link;
```

```
                Node<type> prev = null;

                while (d.Link != null)
                {
                    if (d.Data.CompareTo(d.Link.Data) < 0)
                    {
                        type temp = d.Data;
                        d.Data = d.Link.Data;
                        d.Link.Data = temp;

                        flag = true;
                    }
                    prev = d;
                    d = d.Link;
                }
            }
        }
    }
}

namespace L3
{
    public partial class WebForm : System.Web.UI.Page
    {
        /// <summary>
        /// validates the textbox input
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator1_ServerValidate(object source, ServerValidateEventArgs
args)
        {
            int N;
            args.IsValid = int.TryParse(TextBox1.Text, out N) && N > 0 &&
!String.IsNullOrEmpty(TextBox1.Text);

        }
        /// <summary>
        /// validates the file upload
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator2_ServerValidate(object source, ServerValidateEventArgs
args)
        {
            if (FileUpload1.HasFile && FileUpload1.FileName == "U17a.txt")
            {
                args.IsValid = true;
            }
            else
            {
                args.IsValid = false;
            }

        }
        /// <summary>
        /// validates the file upload
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator3_ServerValidate(object source, ServerValidateEventArgs
args)
        {
            if (FileUpload2.HasFile && FileUpload2.FileName == "U17b.txt")
            {
                args.IsValid = true;
            }
            else
            {
                args.IsValid = false;
            }

        }
        /// <summary>
        /// Method that updates the tables with the data from the linked list
        /// </summary>
        /// <typeparam name="T"></typeparam>
        /// <param name="list"></param>
        /// <param name="table"></param>
        /// <param name="IsHotel"></param>
        /// <param name="IsAnswer"></param>
        private void UpdateTables<T>(CustomLinkedList<T> list, Table table, bool IsHotel, bool
IsAnswer) where T : IComparable<T>, IEquatable<T>
        {
            if (list.Count != 0)
            {
                table.Rows.Clear();
                TableHeaderRow headerRow = new TableHeaderRow();
                if (IsAnswer)
                {
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Pavarde" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Vardas" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Suma" });
                }
                else if (IsHotel)
                {
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Kaina" });
                }
                else
                {
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Pavarde" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Vardas" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Viešbutis" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Kambario tipas" });
                    headerRow.Cells.Add(new TableHeaderCell { Text = "Nakvynių skaičius" });
                }
                table.Rows.Add(headerRow);
                foreach (T item in list)
                {
                    TableRow row = new TableRow();
                    if (IsAnswer && item is Tourist tourist)
                    {
                        row = tourist.ToTableRow(true);
                    }
                    else if (item is Tourist tourist2)
                    {
                        row = tourist2.ToTableRow();
                    }
                    else if (item is Hotel hotel)
                    {
                        row = hotel.ToTableRow();
                    }
                    table.Rows.Add(row);
                }
            }
            else
            {
                table.Rows.Clear();
                TableHeaderRow headerRow = new TableHeaderRow();
                headerRow.Cells.Add(new TableHeaderCell { Text = "Nera duomenu" });
                table.Rows.Add(headerRow);
            }
        }
```

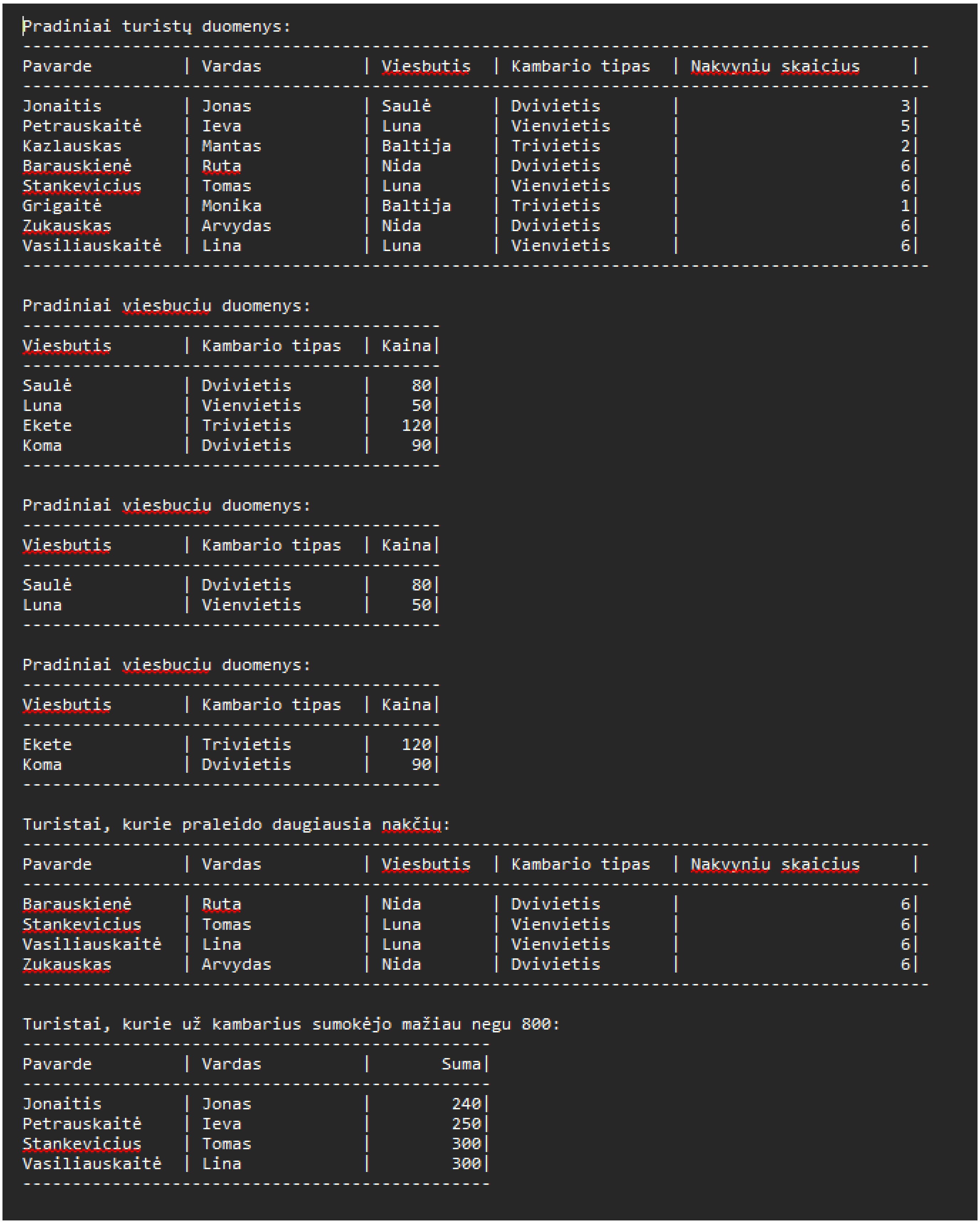
```
    }
}

body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
}

.container {
    width: 80%;
    margin: 20px auto;
    padding: 20px;
    background-color: white;
    border: 1px solid #ccc;
    height: 1000px;
}

.label {
    font-weight: bold;
}

.table {
    background-color: white;
    border-color: black;
    border-style: solid;
    border-width: 1px;
    color: black;
    border-collapse: collapse;
}

    .table td, .table th {
        border: 1px solid black;
        padding: 5px;
    }

.textbox {
    padding: 5px;
    border: 1px solid #ccc;
}

.button {
    padding: 10px 20px;
    background-color: #007bff;
    color: white;
    border: none;
    cursor: pointer;
}

    .button:hover {
        background-color: #0056b3;
    }

.custom-validator {
    color: red;
}

namespace L3
{
      public static class TaskUtils
    {
        public static WebForm WebForm
        {
            get => default;
            set
            {
            }
        }
        public static Hotel Hotel
        {
            get => default;
            set
            {
            }
        }

        public static Tourist Tourist
        {
            get => default;
            set
            {
            }
        }
        /// <summary>
        /// Finds the hotels that were used and unused
        /// </summary>
        /// <param name="BaseTourists"></param>
        /// <param name="original"></param>
        /// <param name="used"></param>
        /// <param name="unused"></param>
        public static void FindUsedUnused(CustomLinkedList<Tourist> BaseTourists,
CustomLinkedList<Hotel> original, CustomLinkedList<Hotel> used, CustomLinkedList<Hotel>
unused)
        {
            foreach(Hotel hotel in original)
            {

                bool isUsed = false;

                foreach (Tourist tourist in BaseTourists)
                {
                    if (hotel == tourist)
                    {
                        isUsed = true;
                        break;
                    }
                }

                if (isUsed)
                {
                    used.AddToEnd(hotel);
                }
                else
                {
                    unused.AddToEnd(hotel);
                }
            }
        }
        /// <summary>
        /// Finds the sum of each of the tourists
        /// </summary>
        /// <param name="Tourists"></param>
        /// <param name="Hotels"></param>
        public static void FindSum(CustomLinkedList<Tourist> Tourists, CustomLinkedList<Hotel>
Hotels)
        {
            foreach (Tourist tourist in Tourists)
            {
                foreach (Hotel hotel in Hotels)
                {
                    if (hotel == tourist)
                    {
                        tourist.Sum = hotel * tourist;
                    }
                }
            }
        }
        /// <summary>
        /// Finds the longest stay
        /// </summary>
        /// <param name="Tourists"></param>
        /// <returns></returns>
        public static int FindLongestStay(CustomLinkedList<Tourist> Tourists)
        {
            int longest = 0;
            foreach (Tourist tourist in Tourists)
            {
                if (tourist.BookedNights > longest)
                {
                    longest = tourist.BookedNights;
                }
            }
            return longest;
        }
        /// <summary>
        /// Finds tourists who stayed the longest
        /// </summary>
        /// <param name="Tourists"></param>
        /// <param name="longestStay"></param>
        /// <param name="LongestStayTourists"></param>
        public static void FindTouristsWithLongestStay(CustomLinkedList<Tourist> Tourists, int
longestStay, CustomLinkedList<Tourist> LongestStayTourists)
        {
            foreach (Tourist tourist in Tourists)
            {
                if (tourist.BookedNights == longestStay)
                {
                    LongestStayTourists.AddToEnd(tourist);
                }
            }
        }
        /// <summary>
        /// Finds tourists who spent less than the maximum amount
        /// </summary>
        /// <param name="Tourists"></param>
        /// <param name="maximum"></param>
        /// <param name="MinSpenders"></param>
        public static void findMinSpenders(CustomLinkedList<Tourist> Tourists, int maximum,
CustomLinkedList<Tourist> MinSpenders)
        {
            foreach (Tourist tourist in Tourists)
            {
                if (tourist.Sum < maximum && tourist.Sum !=0)
                {
                    MinSpenders.AddToEnd(tourist);
                }
            }
        }
    }
}

namespace L3
{
    public class Tourist : IComparable<Tourist>, IEquatable<Tourist>
    {
        public string LastName { get; set; }
        public string FirstName { get; set; }
        public string HotelName { get; set; }
        public string RoomType { get; set; }
        public int BookedNights { get; set; }

        public int Sum { get; set; } = 0;
        /// <summary>
        /// Constructor for the tourist object
        /// </summary>
        /// <param name="lastName"></param>
        /// <param name="firstName"></param>
        /// <param name="hotelName"></param>
        /// <param name="roomType"></param>
        /// <param name="bookedNights"></param>
        public Tourist(string lastName, string firstName, string hotelName, string roomType, 
int bookedNights)
        {
            LastName = lastName;
            FirstName = firstName;
            HotelName = hotelName;
            RoomType = roomType;
            BookedNights = bookedNights;
        }
        /// <summary>
        /// Overriden ToString method for the tourist object
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            string line = $"{LastName,-15} | {FirstName,-15} | {HotelName,-10} | {RoomType,-
15} | {BookedNights,22}|";
            return line;
        }
        /// <summary>
        /// Compares two tourist objects by their last name and first name
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public int CompareTo(Tourist other)
        {
            if (this.LastName.CompareTo(other.LastName) == 0)
            {
                return this.FirstName.CompareTo(other.FirstName);
            }
            return this.LastName.CompareTo(other.LastName);
        }
        /// <summary>
        /// Makes sure the objects are equal by comparing their properties
        /// </summary>
        /// <param name="obj"></param>
        /// <returns></returns>
        public override bool Equals(object obj)
        {
            return obj is Tourist tourist &&
                   LastName == tourist.LastName &&
                   FirstName == tourist.FirstName &&
                   HotelName == tourist.HotelName &&
                   RoomType == tourist.RoomType &&
                   BookedNights == tourist.BookedNights;
        }
        /// <summary>
        /// makes sure the tourists are equal by comparing their properties
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public bool Equals(Tourist other)
        {
            return other != null &&
                   LastName == other.LastName &&
                   FirstName == other.FirstName &&
                   HotelName == other.HotelName &&
                   RoomType == other.RoomType &&
                   BookedNights == other.BookedNights;
        }
        /// <summary>
        /// Gets the hashcode of the tourist object
        /// </summary>
        /// <returns></returns>
        public override int GetHashCode()
        {
            int hashCode = 499324390;
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(LastName);
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(FirstName);
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(HotelName);
            hashCode = hashCode * -1521134295 +
EqualityComparer<string>.Default.GetHashCode(RoomType);
            hashCode = hashCode * -1521134295 + BookedNights.GetHashCode();
            return hashCode;
        }

        /// <summary>
        /// Compares a hotel and tourist object by their hotel name and room type
        /// </summary>
        /// <param name="hotel"></param>
        /// <param name="tourist"></param>
        /// <returns></returns>
        public static bool operator ==(Hotel hotel, Tourist tourist)
        {
            return hotel.HotelName == tourist.HotelName && hotel.HotelType ==
tourist.RoomType;
        }
        /// <summary>
        /// Compares a hotel and tourist object by their hotel name and room type
        /// </summary>
        /// <param name="hotel"></param>
        /// <param name="tourist"></param>
        /// <returns></returns>
        public static bool operator !=(Hotel hotel, Tourist tourist)
        {
            return !(hotel == tourist);
        }

        /// <summary>
        /// Operator to calculate the total cost of the tourist's stay
        /// </summary>
        /// <param name="hotel"></param>
        /// <param name="tourist"></param>
        /// <returns></returns>
        public static int operator *(Hotel hotel, Tourist tourist)
        {
            return tourist.BookedNights * hotel.Cost;
        }

        public TableRow ToTableRow(bool isAnswer = false)
        {
            TableRow row = new TableRow();
            if (isAnswer)
            {
                row.Cells.Add(new TableCell { Text = LastName });
                row.Cells.Add(new TableCell { Text = FirstName });
                row.Cells.Add(new TableCell { Text = Sum.ToString() });
            }
            else
            {
                row.Cells.Add(new TableCell { Text = LastName });
                row.Cells.Add(new TableCell { Text = FirstName });
                row.Cells.Add(new TableCell { Text = HotelName });
                row.Cells.Add(new TableCell { Text = RoomType });
                row.Cells.Add(new TableCell { Text = BookedNights.ToString() });
            }
            return row;
        }
    }
}

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

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
    <form id="form1" runat="server">
        <div class="container">
            <asp:Label ID="Label2" runat="server" Text="Turistų informacija:" CssClass="label"
Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table1" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label3" runat="server" Text="Viešbučių informacija:"
CssClass="label" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table2" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label1" runat="server" Text="Maksimali pinigų suma:"
CssClass="label"></asp:Label>
            <br />
            <asp:TextBox ID="TextBox1" runat="server" CssClass="textbox"></asp:TextBox>
            <asp:CustomValidator ID="CustomValidator1" runat="server"
ControlToValidate="TextBox1" ErrorMessage="Tik realieji skaičiai"
OnServerValidate="CustomValidator1_ServerValidate"
ValidateEmptyText="True"></asp:CustomValidator>
            <br />
            <br />
            <asp:Label ID="Label4" runat="server" Text="Turistų failas (U17a.txt)"
CssClass="label"></asp:Label>
            <br />
            <asp:FileUpload ID="FileUpload1" runat="server" />
            <asp:CustomValidator ID="CustomValidator2" runat="server"
ControlToValidate="FileUpload1" ErrorMessage="Netinkamas failas"
OnServerValidate="CustomValidator1_ServerValidate"
ValidateEmptyText="True"></asp:CustomValidator>
            <br />
            <br />
            <asp:Label ID="Label5" runat="server" Text="Viešbučių failas (U17b.txt)"
CssClass="label"></asp:Label>
            <br />
            <asp:FileUpload ID="FileUpload2" runat="server" />
            <asp:CustomValidator ID="CustomValidator3" runat="server"
ControlToValidate="FileUpload2" ErrorMessage="Netinkamas failas"
OnServerValidate="CustomValidator1_ServerValidate"
ValidateEmptyText="True"></asp:CustomValidator>
            <br />
            <br />
            <asp:Button ID="Button1" runat="server" Text="Skaičiuoti" CssClass="button"
OnClick="Button1_Click" />
            <br />
            <br />
            <br />
            <asp:Label ID="Label6" runat="server" Text="Panaudoti viešbučiai:"
CssClass="label" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table5" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label7" runat="server" Text="Nepanaudoti viešbučiai"
CssClass="label" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table6" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label8" runat="server" Text="Maksimalus nakvynių skaičius:"
CssClass="label" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table3" runat="server" CssClass="table"></asp:Table>
            <br />
            <asp:Label ID="Label9" runat="server" Text="Turistai išleidę mažiau negu
maksimumas:" CssClass="label" Visible="False"></asp:Label>
            <br />
            <asp:Table ID="Table4" runat="server" CssClass="table"></asp:Table>
        </div>
    </form>
</body>
</html>

namespace L3
{
      public partial class WebForm : System.Web.UI.Page
      {
             protected void Page_Load(object sender, EventArgs e)
             {

             }

        protected void Button1_Click(object sender, EventArgs e)
        {
            if (Page.IsValid)
            {



                string inputHotels = Server.MapPath($"App_Data/U17b.txt");
                string inputTourists = Server.MapPath($"App_Data/U17a.txt");
                string result = Server.MapPath($"App_Data/Rezultatai.txt");

                string headerOne = $"{"Pavarde",-15} | {"Vardas",-15} | {"Viesbutis",-10} |
{"Kambario tipas",-15} | {"Nakvyniu skaicius",-22}| ";
                string headerTwo = $"{"Viesbutis",-15} | {"Kambario tipas",-15} |
{"Kaina",5}|";
                string HeaderThree = $"{"Pavarde",-15} | {"Vardas",-15} | {"Suma",10}|";

                if (File.Exists(inputHotels)) File.Delete(inputHotels);
                if (File.Exists(inputTourists)) File.Delete(inputTourists);
                if (File.Exists(result)) File.Delete(result);

                FileUpload1.SaveAs(inputTourists);
                FileUpload2.SaveAs(inputHotels);

                CustomLinkedList<Tourist> Tourists = new CustomLinkedList<Tourist>();
                CustomLinkedList<Hotel> Hotels = new CustomLinkedList<Hotel>();

                InOutUtils.ReadTourists(inputTourists, Tourists);
                InOutUtils.ReadHotel(inputHotels, Hotels);
                UpdateTables(Tourists, Table1, false, false);
                UpdateTables(Hotels, Table2, true, false);

                //UpdateTableTourist(Tourists, Table1);
                //UpdateTableHotel(Hotels, Table2);


                //InOutUtils.PrintTourists(result, Tourists, headerOne, false);
                //InOutUtils.PrintHotels(result, Hotels, headerTwo);
                InOutUtils.Print(result, headerOne, "Pradiniai turistų duomenys:", false,
Tourists);
                InOutUtils.Print(result, headerTwo, "Pradiniai viesbuciu duomenys:", false,
Hotels);

                CustomLinkedList<Hotel> usedHotels = new CustomLinkedList<Hotel>();
                CustomLinkedList<Hotel> unsedHotels = new CustomLinkedList<Hotel>();

                TaskUtils.FindUsedUnused(Tourists, Hotels, usedHotels, unsedHotels);

                UpdateTables(usedHotels, Table5, true, false);
                UpdateTables(unsedHotels, Table6, true, false);
                InOutUtils.Print(result, headerTwo, "Panaudoti viešbučiai:", false,
usedHotels);
                InOutUtils.Print(result, headerTwo, "Nepanaudoti viešbučiai:", false,
unsedHotels);

                int maximum = int.Parse(TextBox1.Text);
                int longestStay = TaskUtils.FindLongestStay(Tourists);


                CustomLinkedList<Tourist> LongestStayTourists = new
CustomLinkedList<Tourist>();
                CustomLinkedList<Tourist> MinimumSpenderTourists = new
CustomLinkedList<Tourist>();

                TaskUtils.FindTouristsWithLongestStay(Tourists, longestStay,
LongestStayTourists);
                LongestStayTourists.BubbleSort();


                UpdateTables(LongestStayTourists, Table3, false, false);
                InOutUtils.Print(result, headerOne, "Ilgiausiai prabuve turistai:", false,
LongestStayTourists);

                TaskUtils.FindSum(Tourists, Hotels);
                TaskUtils.findMinSpenders(Tourists, maximum, MinimumSpenderTourists);
                MinimumSpenderTourists.BubbleSort();
                UpdateTables(MinimumSpenderTourists,Table4,false,true);
                InOutUtils.Print(result, headerOne, "Mažiausiai išleidę turistai:", true,
MinimumSpenderTourists);
                Label6.Visible = true;
                Label7.Visible = true;
                Label8.Visible = true;
                Label9.Visible = true;
            }
        }
    }
}

namespace L3.Tests
{
    [TestClass()]
    public class CustomLinkedListTests
    {
        private CustomLinkedList<int> list;
        private CustomLinkedList<Tourist> list2;
        private CustomLinkedList<Hotel> list3;

        [TestInitialize]
        public void Setup()
        {
            list = new CustomLinkedList<int>();
            list.AddToEnd(1);
            list.AddToEnd(2);
            list.AddToEnd(3);
            list2 = new CustomLinkedList<Tourist>();
            list2.AddToEnd(new Tourist("Alastname", "Afirstname", "Hotel1", "Room1", 5));
            list2.AddToEnd(new Tourist("Blastname", "Bfirstname", "Hotel2", "Room2", 3));
            list2.AddToEnd(new Tourist("Clastname", "Cfirstname", "Hotel3", "Room3", 2));
            list3 = new CustomLinkedList<Hotel>();
            list3.AddToEnd(new Hotel("AHotel", "Room1", 5));
            list3.AddToEnd(new Hotel("BHotel", "Room2", 3));
            list3.AddToEnd(new Hotel("CHotel", "Room3", 2));
        }

        [TestMethod]
        public void AddtoEndTest()
        {
            var enumerator = list.GetEnumerator();
            enumerator.MoveNext();
            Assert.AreEqual(1, enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(2, enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(3, enumerator.Current);
        }

        [TestMethod]
        public void AddtoEndTestTourist()
        {
            var enumerator = list2.GetEnumerator();
            enumerator.MoveNext();
            Assert.AreEqual(new Tourist("Alastname", "Afirstname", "Hotel1", "Room1", 5),
enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Tourist("Blastname", "Bfirstname", "Hotel2", "Room2", 3),
enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Tourist("Clastname", "Cfirstname", "Hotel3", "Room3", 2),
enumerator.Current);
        }

        [TestMethod]
        public void AddtoEndTestHotel()
        {
            var enumerator = list3.GetEnumerator();
            enumerator.MoveNext();
            Assert.AreEqual(new Hotel("AHotel", "Room1", 5), enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Hotel("BHotel", "Room2", 3), enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Hotel("CHotel", "Room3", 2), enumerator.Current);
        }

        [TestMethod]
        public void GetTest()
        {
            list.Start();
            Assert.AreEqual(1, list.Get());
        }

        [TestMethod]
        public void GetTestTourist()
        {
            list2.Start();
            Assert.AreEqual(new Tourist("Alastname", "Afirstname", "Hotel1", "Room1", 5),
list2.Get());
        }

        [TestMethod]
        public void GetTestHotel()
        {
            list3.Start();
            Assert.AreEqual(new Hotel("AHotel", "Room1", 5), list3.Get());
        }

        [TestMethod]
        public void NextTest()
        {
            list.Start();
            list.Next();
            Assert.AreEqual(2, list.Get());
        }

        [TestMethod]
        public void NextTestTourist()
        {
            list2.Start();
            list2.Next();
            Assert.AreEqual(new Tourist("Blastname", "Bfirstname", "Hotel2", "Room2", 3),
list2.Get());
        }

        [TestMethod]
        public void NextTestHotel()
        {
            list3.Start();
            list3.Next();
            Assert.AreEqual(new Hotel("BHotel", "Room2", 3), list3.Get());
        }

        [TestMethod]
        public void ExistsTest()
        {
            list.Start();
            Assert.IsTrue(list.Exists());
            list.Next();
            Assert.IsTrue(list.Exists());
            list.Next();
            Assert.IsTrue(list.Exists());
            list.Next();
            Assert.IsFalse(list.Exists());
        }

        [TestMethod]
        public void ExistsTestTourist()
        {
            list2.Start();
            Assert.IsTrue(list2.Exists());
            list2.Next();
            Assert.IsTrue(list2.Exists());
            list2.Next();
            Assert.IsTrue(list2.Exists());
            list2.Next();
            Assert.IsFalse(list2.Exists());
        }

        [TestMethod]
        public void ExistsTestHotel()
        {
            list3.Start();
            Assert.IsTrue(list3.Exists());
            list3.Next();
            Assert.IsTrue(list3.Exists());
            list3.Next();
            Assert.IsTrue(list3.Exists());
            list3.Next();
            Assert.IsFalse(list3.Exists());
        }

        [TestMethod]
        public void StartTest()
        {
            list.Start();
            Assert.AreEqual(1, list.Get());
        }

        [TestMethod]
        public void StartTestTourist()
        {
            list2.Start();
            Assert.AreEqual(new Tourist("Alastname", "Afirstname", "Hotel1", "Room1", 5),
list2.Get());
        }

        [TestMethod]
        public void StartTestHotel()
        {
            list3.Start();
            Assert.AreEqual(new Hotel("AHotel", "Room1", 5), list3.Get());
        }

        [TestMethod]
        public void EndTest()
        {
            list.End();
            Assert.IsFalse(list.Exists());
        }

        [TestMethod]
        public void EndTestTourist()
        {
            list2.End();
            Assert.IsFalse(list2.Exists());
        }

        [TestMethod]
        public void EndTestHotel()
        {
            list3.End();
            Assert.IsFalse(list3.Exists());
        }

        [TestMethod]
        public void SortTest()
        {
            list.BubbleSort();
            var enumerator = list.GetEnumerator();
            enumerator.MoveNext();
            Assert.AreEqual(3, enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(2, enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(1, enumerator.Current);
        }

        [TestMethod]
        public void SortTestTourist()
        {
            list2.BubbleSort();
            var enumerator = list2.GetEnumerator();
            enumerator.MoveNext();
            Assert.AreEqual(new Tourist("Clastname", "Cfirstname", "Hotel3", "Room3", 2),
enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Tourist("Blastname", "Bfirstname", "Hotel2", "Room2", 3),
enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Tourist("Alastname", "Afirstname", "Hotel1", "Room1", 5),
enumerator.Current);
```

```
        }

        [TestMethod]
        public void SortTestHotel()
        {
            list3.BubbleSort();
            var enumerator = list3.GetEnumerator();
            enumerator.MoveNext();
            Assert.AreEqual(new Hotel("CHotel", "Room3", 2), enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Hotel("BHotel", "Room2", 3), enumerator.Current);
            enumerator.MoveNext();
            Assert.AreEqual(new Hotel("AHotel", "Room1", 5), enumerator.Current);
        }
    }
}
```

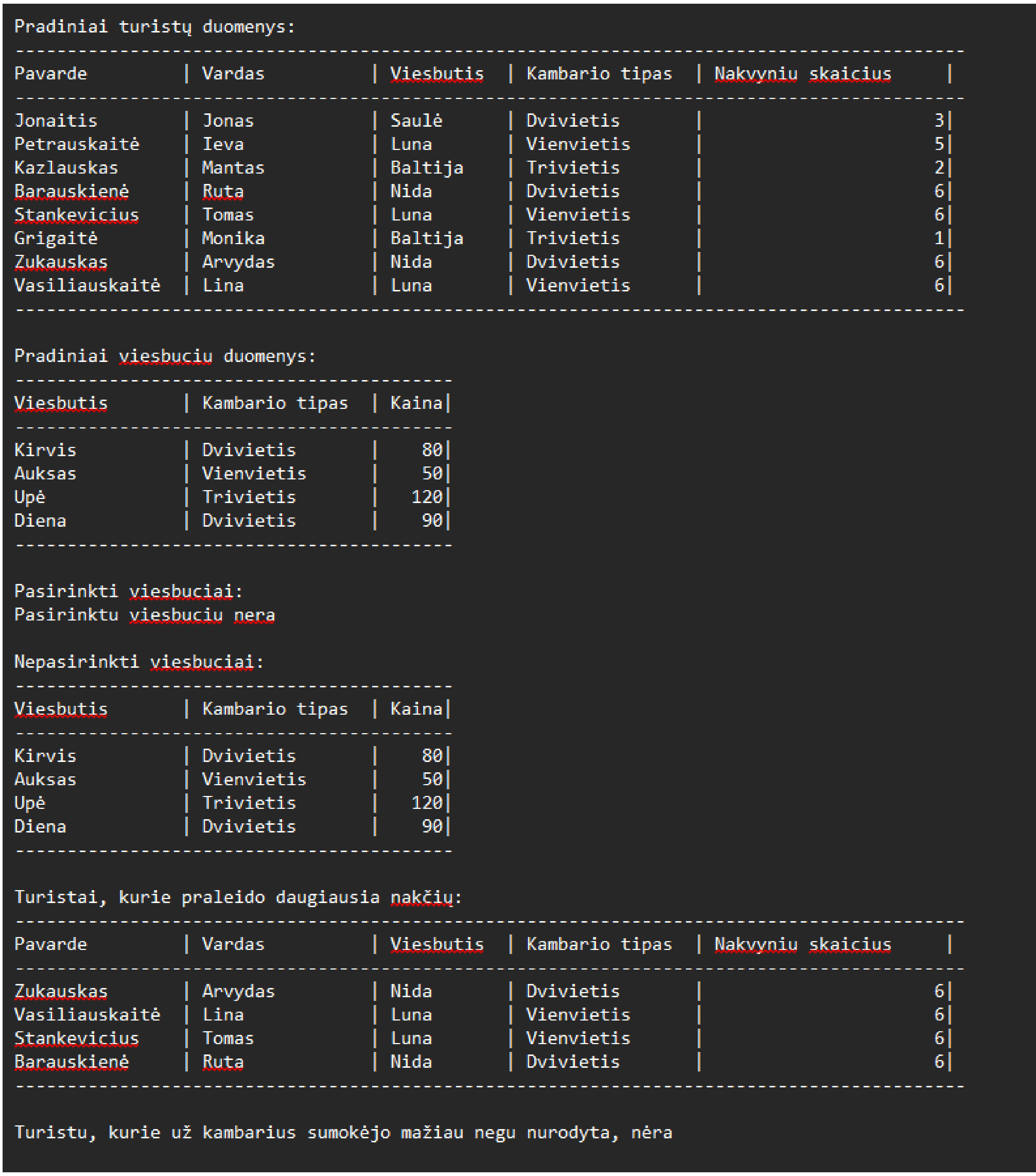


```
Test run finished: 21 Tests (21 Passed, 0 Failed, 0 Skipped) run in 511 ms
Test                                   Duration    Traits    Er
◢ ✓ L3Tests1 (21)                        132 ms
  ◢ ✓ L3.Tests (21)                      132 ms
    ◢ ✓ CustomLinkedListTests (21)       132 ms
        ✓ AddtoEndTest                   131 ms
        ✓ AddtoEndTestHotel                1 ms
        ✓ AddtoEndTestTourist            < 1 ms
        ✓ EndTest                        < 1 ms
        ✓ EndTestHotel                   < 1 ms
        ✓ EndTestTourist                 < 1 ms
        ✓ ExistsTest                     < 1 ms
        ✓ ExistsTestHotel                < 1 ms
        ✓ ExistsTestTourist              < 1 ms
        ✓ GetTest                        < 1 ms
        ✓ GetTestHotel                   < 1 ms
        ✓ GetTestTourist                 < 1 ms
        ✓ NextTest                       < 1 ms
        ✓ NextTestHotel                  < 1 ms
        ✓ NextTestTourist                < 1 ms
        ✓ SortTest                       < 1 ms
        ✓ SortTestHotel                  < 1 ms
        ✓ SortTestTourist                < 1 ms
        ✓ StartTest                      < 1 ms
        ✓ StartTestHotel                 < 1 ms
        ✓ StartTestTourist               < 1 ms
```

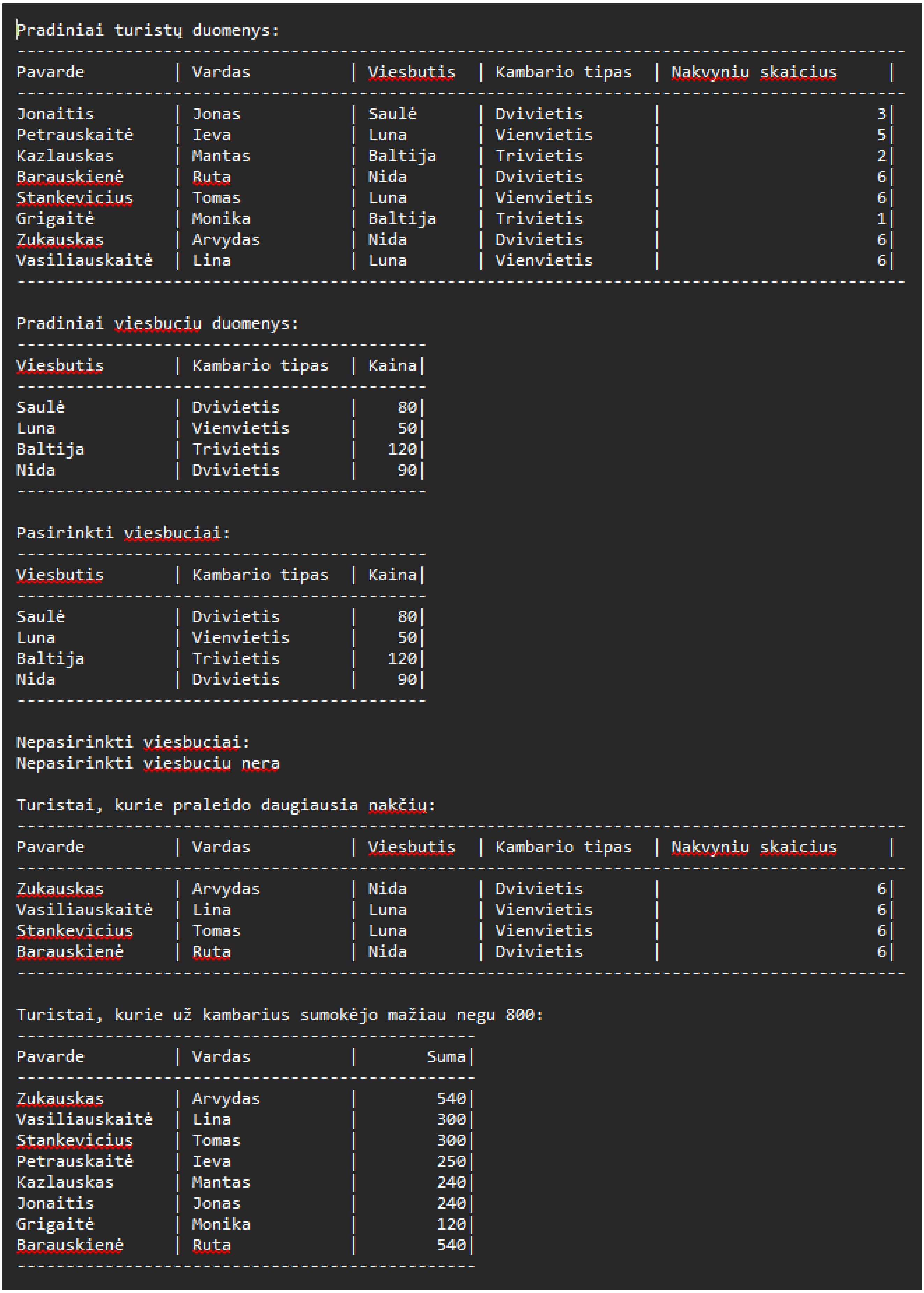
3.7. Pradiniai duomenys ir rezultatai



```
Pradiniai turistų duomenys:
-------------------------------------------------------------------------------------------
Pavarde         | Vardas          | Viesbutis  | Kambario tipas  | Nakvyniu skaicius     |
-------------------------------------------------------------------------------------------
Jonaitis        | Jonas           | Saulė      | Dvivietis       |                      3|
Petrauskaitė    | Ieva            | Luna       | Vienvietis      |                      5|
Kazlauskas      | Mantas          | Baltija    | Trivietis       |                      2|
Barauskienė     | Ruta            | Nida       | Dvivietis       |                      6|
Stankevicius    | Tomas           | Luna       | Vienvietis      |                      6|
Grigaitė        | Monika          | Baltija    | Trivietis       |                      1|
Zukauskas       | Arvydas         | Nida       | Dvivietis       |                      6|
Vasiliauskaitė  | Lina            | Luna       | Vienvietis      |                      6|
-------------------------------------------------------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis       | Kambario tipas  | Kaina|
------------------------------------------
Saulė           | Dvivietis       |    80|
Luna            | Vienvietis      |    50|
Ekete           | Trivietis       |   120|
Koma            | Dvivietis       |    90|
------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis       | Kambario tipas  | Kaina|
------------------------------------------
Saulė           | Dvivietis       |    80|
Luna            | Vienvietis      |    50|
------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis       | Kambario tipas  | Kaina|
------------------------------------------
Ekete           | Trivietis       |   120|
Koma            | Dvivietis       |    90|
------------------------------------------

Turistai, kurie praleido daugiausia nakčių:
-------------------------------------------------------------------------------------------
Pavarde         | Vardas          | Viesbutis  | Kambario tipas  | Nakvyniu skaicius     |
-------------------------------------------------------------------------------------------
Barauskienė     | Ruta            | Nida       | Dvivietis       |                      6|
Stankevicius    | Tomas           | Luna       | Vienvietis      |                      6|
Vasiliauskaitė  | Lina            | Luna       | Vienvietis      |                      6|
Zukauskas       | Arvydas         | Nida       | Dvivietis       |                      6|
-------------------------------------------------------------------------------------------

Turistai, kurie už kambarius sumokėjo mažiau negu 800:
-----------------------------------------------
Pavarde         | Vardas          |       Suma|
-----------------------------------------------
Jonaitis        | Jonas           |        240|
Petrauskaitė    | Ieva            |        250|
Stankevicius    | Tomas           |        300|
Vasiliauskaitė  | Lina            |        300|
-----------------------------------------------
```



```
Pradiniai turistų duomenys:
-------------------------------------------------------------------------------------------
Pavarde          | Vardas           | Viesbutis  | Kambario tipas  | Nakvyniu skaicius     |
-------------------------------------------------------------------------------------------
Jonaitis         | Jonas            | Saulė      | Dvivietis       |                      3|
Petrauskaitė     | Ieva             | Luna       | Vienvietis      |                      5|
Kazlauskas       | Mantas           | Baltija    | Trivietis       |                      2|
Barauskienė      | Ruta             | Nida       | Dvivietis       |                      6|
Stankevicius     | Tomas            | Luna       | Vienvietis      |                      6|
Grigaitė         | Monika           | Baltija    | Trivietis       |                      1|
Zukauskas        | Arvydas          | Nida       | Dvivietis       |                      6|
Vasiliauskaitė   | Lina             | Luna       | Vienvietis      |                      6|
-------------------------------------------------------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis        | Kambario tipas  | Kaina|
------------------------------------------
Kirvis           | Dvivietis        |   80|
Auksas           | Vienvietis       |   50|
Upė              | Trivietis        |  120|
Diena            | Dvivietis        |   90|
------------------------------------------

Pasirinkti viesbuciai:
Pasirinktu viesbuciu nera

Nepasirinkti viesbuciai:
------------------------------------------
Viesbutis        | Kambario tipas  | Kaina|
------------------------------------------
Kirvis           | Dvivietis        |   80|
Auksas           | Vienvietis       |   50|
Upė              | Trivietis        |  120|
Diena            | Dvivietis        |   90|
------------------------------------------

Turistai, kurie praleido daugiausia nakčių:
-------------------------------------------------------------------------------------------
Pavarde          | Vardas           | Viesbutis  | Kambario tipas  | Nakvyniu skaicius     |
-------------------------------------------------------------------------------------------
Zukauskas        | Arvydas          | Nida       | Dvivietis       |                      6|
Vasiliauskaitė   | Lina             | Luna       | Vienvietis      |                      6|
Stankevicius     | Tomas            | Luna       | Vienvietis      |                      6|
Barauskienė      | Ruta             | Nida       | Dvivietis       |                      6|
-------------------------------------------------------------------------------------------

Turistu, kurie už kambarius sumokėjo mažiau negu nurodyta, nėra
```



```
Pradiniai turistų duomenys:
-------------------------------------------------------------------------------------------
Pavarde         | Vardas          | Viesbutis  | Kambario tipas  | Nakvyniu skaicius     |
-------------------------------------------------------------------------------------------
Jonaitis        | Jonas           | Saulė      | Dvivietis       |                      3|
Petrauskaitė    | Ieva            | Luna       | Vienvietis      |                      5|
Kazlauskas      | Mantas          | Baltija    | Trivietis       |                      2|
Barauskienė     | Ruta            | Nida       | Dvivietis       |                      6|
Stankevicius    | Tomas           | Luna       | Vienvietis      |                      6|
Grigaitė        | Monika          | Baltija    | Trivietis       |                      1|
Zukauskas       | Arvydas         | Nida       | Dvivietis       |                      6|
Vasiliauskaitė  | Lina            | Luna       | Vienvietis      |                      6|
-------------------------------------------------------------------------------------------

Pradiniai viesbuciu duomenys:
------------------------------------------
Viesbutis       | Kambario tipas  | Kaina|
------------------------------------------
Saulė           | Dvivietis       |    80|
Luna            | Vienvietis      |    50|
Baltija         | Trivietis       |   120|
Nida            | Dvivietis       |    90|
------------------------------------------

Pasirinkti viesbuciai:
------------------------------------------
Viesbutis       | Kambario tipas  | Kaina|
------------------------------------------
Saulė           | Dvivietis       |    80|
Luna            | Vienvietis      |    50|
Baltija         | Trivietis       |   120|
Nida            | Dvivietis       |    90|
------------------------------------------

Nepasirinkti viesbuciai:
Nepasirinkti viesbuciu nera

Turistai, kurie praleido daugiausia nakčių:
-------------------------------------------------------------------------------------------
Pavarde         | Vardas          | Viesbutis  | Kambario tipas  | Nakvyniu skaicius     |
-------------------------------------------------------------------------------------------
Zukauskas       | Arvydas         | Nida       | Dvivietis       |                      6|
Vasiliauskaitė  | Lina            | Luna       | Vienvietis      |                      6|
Stankevicius    | Tomas           | Luna       | Vienvietis      |                      6|
Barauskienė     | Ruta            | Nida       | Dvivietis       |                      6|
-------------------------------------------------------------------------------------------

Turistai, kurie už kambarius sumokėjo mažiau negu 800:
-----------------------------------------------
Pavarde         | Vardas          |       Suma|
-----------------------------------------------
Zukauskas       | Arvydas         |        540|
Vasiliauskaitė  | Lina            |        300|
Stankevicius    | Tomas           |        300|
Petrauskaitė    | Ieva            |        250|
Kazlauskas      | Mantas          |        240|
Jonaitis        | Jonas           |        240|
Grigaitė        | Monika          |        120|
Barauskienė     | Ruta            |        540|
-----------------------------------------------
```

3.8. Dėstytojo pastabos

Pastabų nėra

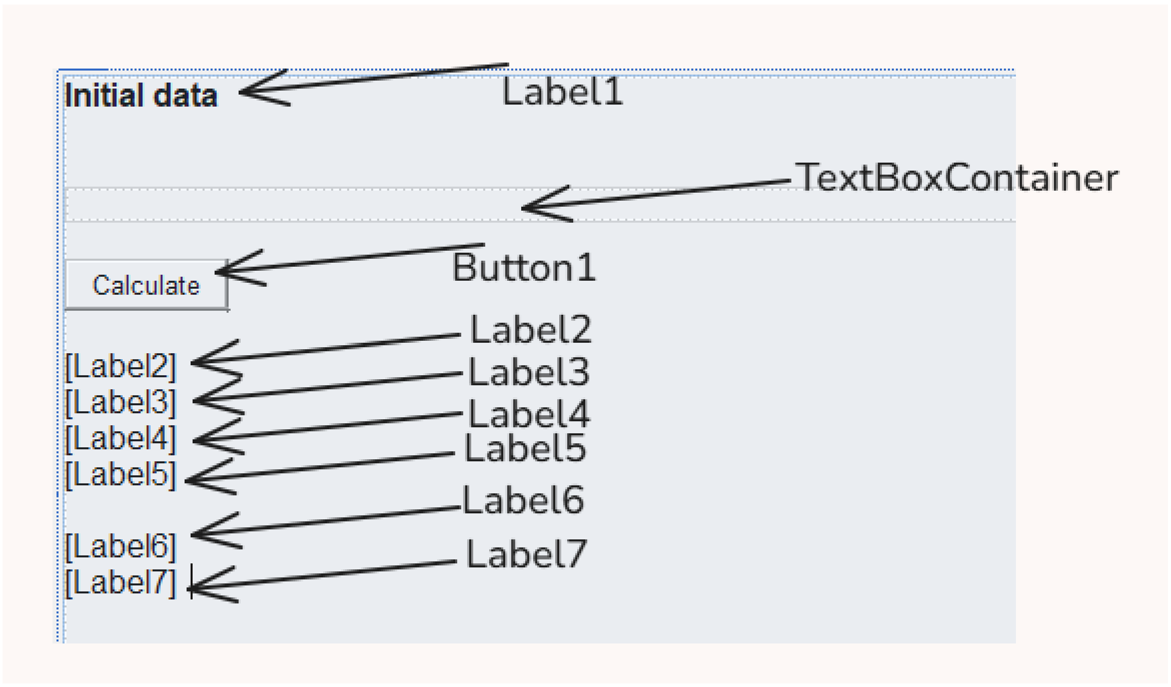
4. Polimorfizmas ir išimčių valdymas (L4)

4.1. Darbo užduotis

U4_17. Protų mūšis. Protų mūšius organizuojančios studentų atstovybės (>=3) nusprendė susivienyti ir sudaryti bendrą klausimų bazę. Pirmoje eilutėje nurodytas studentų atstovybės pavadinimas. Toliau yra klausimai. Protų mūšio klausimai gali būti tik dviejų rūšių: su galimais atsakymų variantais ir muzikiniai. Sukurkite abstrakčiąją klasę „Question“ (savybės – tema, sudėtingumas, klausimo autorius, klausimo tekstas, teisingas atsakymas, balai), kurią paveldės klasės “TestQuestion” (savybės – atsakymo variantai) ir “MusicQuestion” (savybė – failo vardas).

- Raskite, kiek yra I, II ir III sudėtingumo lygio klausimų (visų tipų), rezultatus atspausdinkite ekrane.
- Raskite, kas sukūrė daugiausiai klausimų kiekvienoje atstovybėje (bendrai paėmus), autoriaus vardą bei klausimų kiekį atspausdinkite ekrane.
- Sudarykite sudėtingiausių muzikinių klausimų sąrašą. Įrašykite į failą „SudėtingiMuzikiniai.csv“. Sudarykite bendrai sudėtingiausių klausimų sąrašą. Įrašykite į failą „SudėtingiBendrai.csv“.
- Sudarykite sąrašą klausimų iš temos „Linksmasis“. Įrašykite į failą „Linksmieji.csv“ ir išrikiuokite testo varianto klausimus pagal temą ir sudėtingumą, o muzikinius – pagal failo pavadinimą.

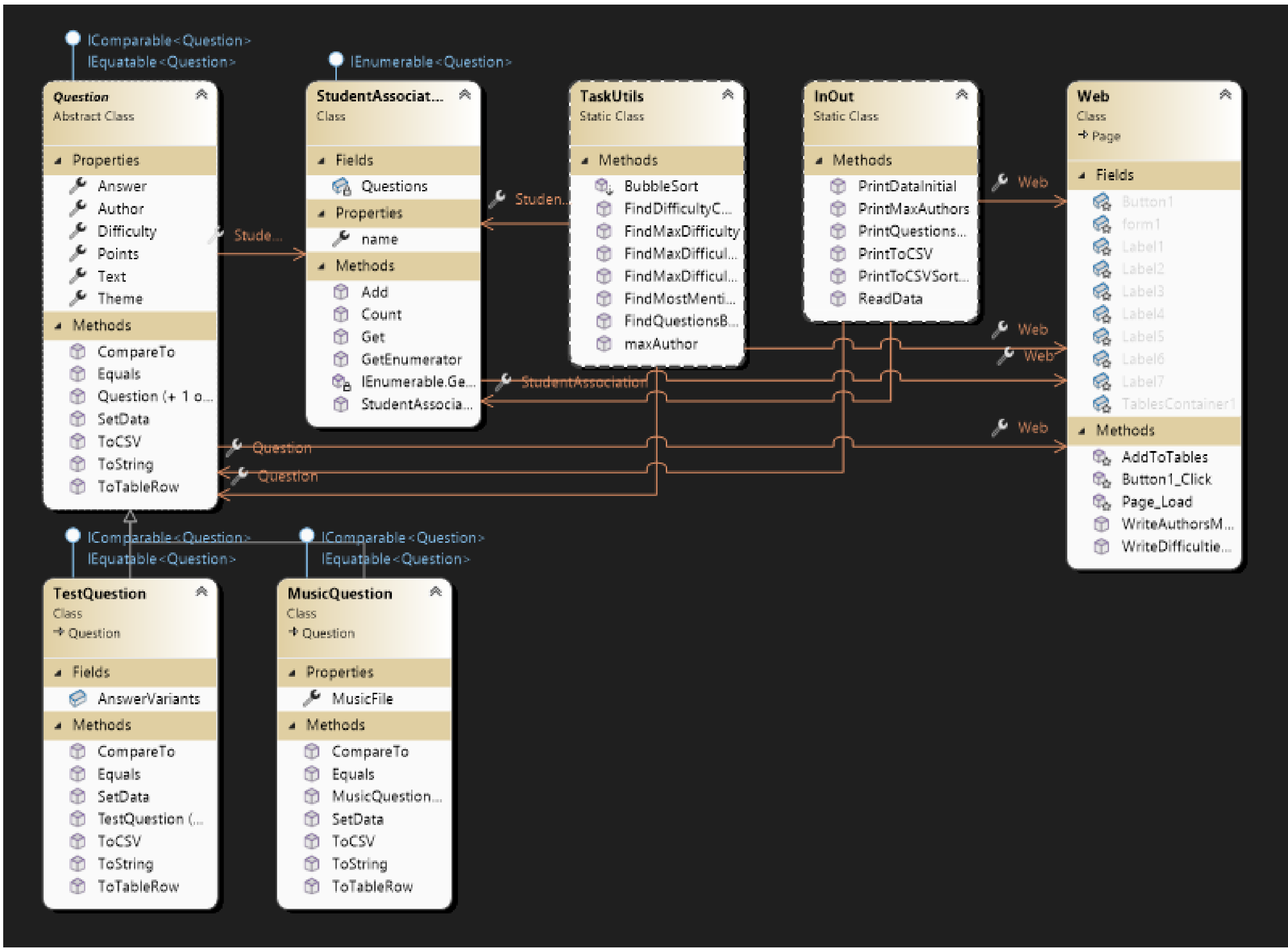
4.2. Grafinės naudotojo sąsajos schema



4.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	**Savybė**	**Reikšmė**
Label(Bendri pakeitimai)	CssClass	label
Table(Bendri pakeitimai)	CssClass	table
Label1	Text	Initial Data
Label2	Text	(Tuščia, užpildoma kode vėliau)
Label3	Text	(Tuščia, užpildoma kode vėliau)
Label4	Text	(Tuščia, užpildoma kode vėliau)
Label5	Text	(Tuščia, užpildoma kode vėliau)
Label6	Text	(Tuščia, užpildoma kode vėliau)
Label7	Text	(Tuščia, užpildoma kode vėliau)
Button1	Text	Skaičiuoti

4.4. Klasių diagrama



4.5. Programos naudotojo vadovas

Vartotojas turi sukurti duomenų failus kiekvienai studentų atstovybei ir įkelti juos į App_Data/Input failų katalogą. Kiekvieno failo pirmoje eilutėje turi būti įrašytas studentų atstovybės pavadinimas. Toliau kiekvienoje naujoje eilutėje turi būti pateikiami protų mūšio klausimai. Klausimai gali būti dviejų rūšių: su galimais atsakymų variantais arba muzikiniai. Klausimo eilutės duomenys pateikiami tokia seka: klausimo tipas (Test arba Music); tema; sudėtingumas; autorius; klausimo tekstas; teisingas atsakymas; balai; papildoma informacija (Test klausimams – atsakymų variantai atskirti „|“ ženklu, Music klausimams – muzikinio failo vardas). Programa paleidžiama ir paspaudžiamas mygtukas – skaičiuoti.

4.6. Programos tekstas

```
namespace L4.App_Code
{
    public static class InOut
    {

        /// <summary>
        /// read data from file
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <returns></returns>
        public static List<StudentAssociation> ReadData(string FileFolder, ref string errors)
        {
            List<StudentAssociation> result = new List<StudentAssociation>();
            StringBuilder errorBuilder = new StringBuilder(errors); // Start with existing
errors

            foreach (string filePath in Directory.GetFiles(FileFolder, "*.txt"))
            {
                string[] lines = File.ReadAllLines(filePath);
                StudentAssociation SA = new StudentAssociation(lines[0]);

                foreach (string line in lines.Skip(1))
                {
                    try
                    {
                        int count = Regex.Split(line, "; ").Length;

                        switch (count)
                        {
                            case 7:
                                SA.Add(new MusicQuestion(line));
                                break;
                            case 10:
                                SA.Add(new TestQuestion(line));
                                break;
                            default:
                                throw new FormatException("Incorrect initial data format");
                        }
                    }
                    catch (FormatException ex)
                    {
                        errorBuilder.AppendLine($"Error in file {Path.GetFileName(filePath)}:
{ex.Message} - Line: {line}<br/>");
                    }
                }
                result.Add(SA);
            }
            errors = errorBuilder.ToString(); // Update the ref parameter
            return result;
        }



        /// <summary>
        /// print initial data to file
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="header"></param>
        /// <param name="List"></param>
        public static void PrintDataInitial(string FileFolder, string header, 
List<StudentAssociation> List)
        {
            using (StreamWriter writer = new StreamWriter(FileFolder + @"\Output.txt", true))
            {
                writer.WriteLine(header);

                foreach (StudentAssociation data in List)
                {
                    writer.WriteLine(data.name);
                    writer.WriteLine(new string('-', 348));
                    writer.WriteLine($"{"Theme",-10} | {"Difficulty",10} | {"Author",-30} | 
{"Text",-75} | {"Answer",-30} | {"Points",6}| {"Music Filename OR 4 different answer 
possibilities",-168} |");
                    writer.WriteLine(new string('-', 348));
                    foreach (Question question in data)
                    {
                        writer.WriteLine(question.ToString());
                        writer.WriteLine(new string('-', 348));
                    }
                    writer.WriteLine();
                }
            }
        }

        /// <summary>
        /// prints questions by difficulty
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="One"></param>
        /// <param name="Two"></param>
        /// <param name="Three"></param>

        public static void PrintQuestionsByDifficulty(string FileFolder, int One, int Two, int 
Three)
        {
            using (StreamWriter writer = new StreamWriter(FileFolder + @"\Output.txt", true))
            {
                writer.WriteLine("Questions by difficulty");
                if (One > 0)
                {
                    writer.WriteLine("Difficulty 1: " + One.ToString());
                }
                else
                {
                    writer.WriteLine("Difficulty 1: No questions");
                }

                if (Two > 0)
                {
                    writer.WriteLine("Difficulty 2: " + One.ToString());
                }
                else
                {
                    writer.WriteLine("Difficulty 2: No questions");
                }

                if (Three > 0)
                {
                    writer.WriteLine("Difficulty 3: " + One.ToString());
                }
                else
                {
                    writer.WriteLine("Difficulty 3: No questions");
                }
                writer.WriteLine();
            }
        }

        /// <summary>
        /// prints the number of difficult questions written by each author
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="List"></param>
        /// <param name="number"></param>

        public static void PrintMaxAuthors(string FileFolder, List<StudentAssociation> List,
int number)
        {
            using (StreamWriter writer = new StreamWriter(FileFolder + @"\Output.txt", true))
            {
                writer.WriteLine("Authors name and the number of difficult questions they
wrote");

                foreach (StudentAssociation SA in List)
                {
                    string authorNames = string.Empty;
                    foreach (Question question in SA)
                    {
                        authorNames += question.Author + ", ";

                    }
                    if (authorNames != string.Empty)
                    {
                        writer.WriteLine(SA.name + ": " + authorNames.TrimEnd(',') +
number.ToString());
                    }
                    else
                    {
                        writer.WriteLine(SA.name + ": No questions");
                    }
                }
                writer.WriteLine();
            }
        }
        /// <summary>
        /// prints to CSV file
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="List"></param>
        /// <param name="FileName"></param>
        public static void PrintToCSV(string FileFolder, List<StudentAssociation> List, string
FileName, string header)
        {
            using (StreamWriter writer = new StreamWriter(Path.Combine(FileFolder, FileName),
true))
            {
                writer.WriteLine(header);

                if (List == null || !List.Any() || List.All(data => data == null ||
!data.Any()))
                {
                    writer.WriteLine("No Questions fitting the criteria");
                    return;
                }
                else
                {
                    writer.WriteLine($"Theme; Difficulty; Author; Text; Answer; Points; Music
Filename OR 4 different answer possibilities");
                }

                foreach (StudentAssociation data in List)
                {
                    if (data != null && data.Any())
                    {
                        foreach (Question question in data)
                        {
                            writer.WriteLine(question.ToCSV());
                        }
                    }
                }
            }
        }


        /// <summary>
        /// prints to CSV file sorted by theme and difficulty
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="List"></param>
        /// <param name="FileName"></param>
        public static void PrintToCSVSorted(string FileFolder, List<Question> List, string
FileName, string header)
        {
            using (StreamWriter writer = new StreamWriter(Path.Combine(FileFolder, FileName),
true))
            {
                writer.WriteLine(header);
                if (List == null || !List.Any())
                {
                    writer.WriteLine("No Questions fitting the criteria");
                    return;
                }
                else
                {
                    writer.WriteLine($"Theme; Difficulty; Author; Text; Answer; Points; Music
Filename OR 4 different answer possibilities");
                }
                foreach (Question question in List)
                {
                    if (question != null)
                    {
                        writer.WriteLine(question.ToCSV());
                    }
                }
            }
        }
    }
}

namespace L4.App_Code
{
       public abstract class Question : IComparable<Question>, IEquatable<Question>
       {
              public string Theme { get; set; }
              public int Difficulty { get; set; }
              public string Author { get; set; }
              public string Text { get; set; }

              public string Answer { get; set; }
              public int Points { get; set; }

        public Question() { }

        public Question(string theme, int difficulty, string author, string text, string
answer, int points)
        {
            Theme = theme;
            Difficulty = difficulty;
            Author = author;
            Text = text;
            Answer = answer;
            Points = points;
        }

        /// <summary>
        /// compare two Question objects by Theme and Difficulty
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public virtual int CompareTo(Question other)
        {
            if (this.Theme.CompareTo(other.Theme) == 0)
            {
                return this.Difficulty.CompareTo(other.Difficulty);
            }
            return this.Theme.CompareTo(other.Theme);
        }
        /// <summary>
        /// compare two Question objects by Author
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public virtual bool Equals(Question other)
        {
            return this.Author == other.Author;
        }

        /// <summary>
        /// Set data for Question object
        /// </summary>
        /// <param name="line"></param>
        /// <exception cref="FormatException"></exception>
        /// <exception cref="IndexOutOfRangeException"></exception>
        public virtual void SetData(string line)
        {
            string[] data = Regex.Split(line, "; ");

            try
            {
                Theme = data[0];
                Difficulty = int.Parse(data[1]);
                Author = data[2];
                Text = data[3];
                Answer = data[4];
                Points = int.Parse(data[5]);
            }
            catch (FormatException)
            {
                throw new FormatException("Invalid data format");
            }
            catch (IndexOutOfRangeException)
            {
                throw new IndexOutOfRangeException("Data is missing");
            }
        }
        /// <summary>
        /// Convert Question object to CSV format
        /// </summary>
        /// <returns></returns>
        public abstract string ToCSV();

        /// <summary>
        /// Convert Question object to string format
```

```
        /// </summary>
        /// <returns></returns>
        public virtual string ToString()
        {
            return $"{Theme, -10} | {Difficulty, 10} | {Author, -30} | {Text, -75} |
{Answer, -30} | {Points, 6}|";
        }

        /// <summary>
        /// Convert Question object to TableRow format
        /// </summary>
        /// <returns></returns>
        public virtual TableRow ToTableRow()
        {
            TableRow row = new TableRow();

            row.Cells.Add(new TableCell { Text = Theme });
            row.Cells.Add(new TableCell { Text = Difficulty.ToString()});
            row.Cells.Add(new TableCell { Text = Author });
            row.Cells.Add(new TableCell { Text = Text });
            row.Cells.Add(new TableCell { Text = Answer });
            row.Cells.Add(new TableCell { Text = Points.ToString() });

            return row;
        }
    }
}

namespace L4.App_Code
{
      public class MusicQuestion : Question , IComparable<Question>, IEquatable<Question>
    {

        public string MusicFile { get; set; }

        public MusicQuestion(string line)
        {
            SetData(line);
        }

        public MusicQuestion(string theme, int difficulty, string author, string text,
string answer, int points, string MusicFile) : base(theme, difficulty, author, text,
answer, points)
        {
            this.MusicFile = MusicFile;
        }
        /// <summary>
        /// compare two MusicQuestion objects by MusicFile
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public override int CompareTo(Question other)
        {
            if (other is MusicQuestion otherMusicQuestion)
            {
                return this.MusicFile.CompareTo(otherMusicQuestion.MusicFile);
            }
            return base.CompareTo(other);
        }
        /// <summary>
        /// compare two MusicQuestion objects by Author
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public override bool Equals(Question other)
        {
            return base.Equals(other);
        }

        /// <summary>
        /// Set data for MusicQuestion object
        /// </summary>
        /// <param name="line"></param>
        /// <exception cref="FormatException"></exception>
        public override void SetData(string line)
        {
            base.SetData(line);
            string[] data = Regex.Split(line, "; ");
            try
            {
                MusicFile = data[6];
            }
            catch (FormatException)
            {
                throw new FormatException("Invalid data format");
            }
        }
        /// <summary>
        /// returns a string representation of the MusicQuestion object
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            return base.ToString() + $"{MusicFile, -40} | {"-", -40} | {"-",-40} | {"-",-
40} |";
        }

        /// <summary>
        /// returns a TableRow object for the MusicQuestion object
        /// </summary>
        /// <returns></returns>
        public override TableRow ToTableRow()
        {
            TableRow row = base.ToTableRow();

            row.Cells.Add(new TableCell { Text = MusicFile });
            row.Cells.Add(new TableCell { Text = "-" });
            row.Cells.Add(new TableCell { Text = "-" });
            row.Cells.Add(new TableCell { Text = "-" });


            return row;
        }
        /// <summary>
        /// returns a CSV representation of the MusicQuestion object
        /// </summary>
        /// <returns></returns>
        public override string ToCSV()
        {
            return $"{Theme}; {Difficulty}; {Author}; {Text}; {Answer};
{Points};{MusicFile}";
        }
    }
}
namespace L4.App_Code
{
    public class TestQuestion : Question, IComparable<Question>, IEquatable<Question>
    {
        public string[] AnswerVariants = new string[4];

        public TestQuestion(string line)
        {
            SetData(line);
        }

        public TestQuestion(string theme, int difficulty, string author, string text,
string answer, int points, string[] AnswerVariants) : base(theme, difficulty, author,
text, answer, points)
        {
            this.AnswerVariants = AnswerVariants;
        }
        /// <summary>
        /// compare two TestQuestion objects by Theme and Difficulty
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public override int CompareTo(Question other)
        {
            if (other is TestQuestion testQuestion)
            {
                if (this.Theme.CompareTo(other.Theme) == 0)
                {
                    return this.Difficulty.CompareTo(other.Difficulty);
                }
                return this.Theme.CompareTo(other.Theme);
            }
            return base.CompareTo(other);
        }
        /// <summary>
        /// compare two TestQuestion objects by Author
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public override bool Equals(Question other)
        {
            return base.Equals(other);
        }
        /// <summary>
        /// Set data for TestQuestion object
        /// </summary>
        /// <param name="line"></param>
        /// <exception cref="IndexOutOfRangeException"></exception>
        public override void SetData(string line)
        {
            base.SetData(line);
            string[] data = Regex.Split(line, "; ");

            for (int i = 0; i < 4; i++)
            {
                try
                {
                    AnswerVariants[i] = data[i + 6];
                }
                catch (IndexOutOfRangeException)
                {
                    throw new IndexOutOfRangeException("Not enough answer variants
provided");
                }
            }
        }
        /// <summary>
        /// returns a string representation of the TestQuestion object
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            return base.ToString() + $"{AnswerVariants[0],-40} | {AnswerVariants[1],-40}
| {AnswerVariants[2],-40} | {AnswerVariants[3],-40} |";
        }
        /// <summary>
        /// returns a TableRow representation of the TestQuestion object
        /// </summary>
        /// <returns></returns>
        public override TableRow ToTableRow()
        {
            TableRow row = base.ToTableRow();

            row.Cells.Add(new TableCell { Text = AnswerVariants[0] });
            row.Cells.Add(new TableCell { Text = AnswerVariants[1] });
            row.Cells.Add(new TableCell { Text = AnswerVariants[2] });
            row.Cells.Add(new TableCell { Text = AnswerVariants[3] });

            return row;
        }

        /// <summary>
        /// returns a CSV representation of the TestQuestion object
        /// </summary>
        /// <returns></returns>
        public override string ToCSV()
        {
            return $"{Theme}; {Difficulty}; {Author}; {Text}; {Answer}; {Points};
{AnswerVariants[0]}; {AnswerVariants[1]}; {AnswerVariants[2]}; {AnswerVariants[3]}";
        }
    }
}

namespace L4.App_Code
{
      public class StudentAssociation : IEnumerable<Question>
    {

            public string name { get; set; }


        private List<Question> Questions;

        public StudentAssociation()
        {
            Questions = new List<Question>();
        }

        public StudentAssociation(string name)
        {
            this.name = name;
            Questions = new List<Question>();
        }

        /// <summary>
        /// Add a question to the list of questions
        /// </summary>
        /// <param name="question"></param>
        public void Add(Question question)
        {
            Questions.Add(question);
        }
        /// <summary>
        /// Count the number of questions in the list
        /// </summary>
        /// <returns></returns>
        public int Count()
        {
            return Questions.Count();
        }

        /// <summary>
        /// Get the list of questions
        /// </summary>
        /// <param name="index"></param>
        /// <returns></returns>
        public Question Get(int index)
        {
            try
            {
                return Questions[index];
            }
            catch
            {
                return null;
            }
        }



        public IEnumerator<Question> GetEnumerator()
        {
            return Questions.GetEnumerator();
        }

        IEnumerator IEnumerable.GetEnumerator()
        {
            return GetEnumerator();
        }
    }
}

public static class TaskUtils
{
    public static void FindDifficultyCount(ref int levelOneHardness, ref int levelTwoHardness,
ref int levelThreeHardness, List<StudentAssociation> Data, ref string errors)
    {
        StringBuilder errorBuilder = new StringBuilder(errors);
        try
        {
            foreach (var SA in Data)
            {
                foreach (Question question in SA)
                {
                    try
                    {
                        if (question.Difficulty == 1)
                        {
                            levelOneHardness++;
                        }
                        else if (question.Difficulty == 2)
                        {
                            levelTwoHardness++;
                        }
                        else if (question.Difficulty == 3)
                        {
                            levelThreeHardness++;
                        }
                        else
                        {
                            throw new FormatException("Invalid difficulty level");
                        }
                    }
                    catch (FormatException ex)
                    {
                        errorBuilder.AppendLine($"Error in StudentAssociation {SA.name}:
{ex.Message}");
                    }
                }
            }
        }
        catch (Exception ex)
        {
            errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
        }
        errors = errorBuilder.ToString();
    }

    public static int FindMaxDifficulty(List<StudentAssociation> Data, ref string errors)
    {
        StringBuilder errorBuilder = new StringBuilder(errors);
        int maxDifficulty = 0;
        try
        {
            foreach (var SA in Data)
            {
                foreach (Question question in SA)
                {
                    try
                    {
                        if (question.Difficulty > maxDifficulty)
                        {
                            maxDifficulty = question.Difficulty;
                        }
                    }
                    catch (Exception ex)
                    {
                        errorBuilder.AppendLine($"Error in StudentAssociation {SA.name}:
{ex.Message}");
                    }
                }
            }
        }
        catch (Exception ex)
        {
            errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
        }
        errors = errorBuilder.ToString();
        return maxDifficulty;
    }

    public static List<StudentAssociation> FindMaxDifficultyList(List<StudentAssociation>
Data, int maxDifficulty, ref string errors)
    {
        StringBuilder errorBuilder = new StringBuilder(errors);
        List<StudentAssociation> result = new List<StudentAssociation>();
        try
        {
            foreach (var SA in Data)
            {
                StudentAssociation temp = new StudentAssociation(SA.name);
                foreach (Question question in SA)
                {
                    try
                    {
                        if (question.Difficulty == maxDifficulty)
                        {
                            temp.Add(question);
                        }
                    }
                    catch (Exception ex)
                    {
                        errorBuilder.AppendLine($"Error in StudentAssociation {SA.name}:
{ex.Message}");
                    }
                }
                result.Add(temp);
            }
        }
        catch (Exception ex)
        {
            errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
        }
        errors = errorBuilder.ToString(); // Update the ref parameter
        return result;
    }

        public static List<StudentAssociation>
FindMaxDifficultyMusical(List<StudentAssociation> Data, ref string errors)
        {
            StringBuilder errorBuilder = new StringBuilder(errors);
            List<StudentAssociation> result = new List<StudentAssociation>();
            try
            {
                foreach (var SA in Data)
                {
                    StudentAssociation temp = new StudentAssociation(SA.name);
                    foreach (Question question in SA)
                    {
                        try
                        {
                            if (question is MusicQuestion)
                            {
                                temp.Add(question);
                            }
                        }
                        catch (Exception ex)
                        {
                            errorBuilder.AppendLine($"Error in StudentAssociation {SA.name}:
{ex.Message}");
                        }
                    }
                    result.Add(temp);
                }
            }
            catch (Exception ex)
            {
                errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
            }
            errors = errorBuilder.ToString(); // Update the ref parameter
            return result;
        }

        public static List<Question> FindQuestionsByTheme(List<StudentAssociation> Data,
string theme, ref string errors)
        {
            StringBuilder errorBuilder = new StringBuilder(errors);
            List<Question> result = new List<Question>();
            try
            {
                foreach (var SA in Data)
                {
                    foreach (Question question in SA)
                    {
                        try
                        {
                            if (question.Theme == theme)
                            {
                                result.Add(question);
                            }
                        }
                        catch (Exception ex)
                        {
                            errorBuilder.AppendLine($"Error in StudentAssociation {SA.name}:
{ex.Message}");
                        }
                    }
                }
            }
            catch (Exception ex)
            {
                errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
            }
            errors = errorBuilder.ToString();
            return result;
        }

        public static Dictionary<string, int>
FindMostMentionedAuthors(List<StudentAssociation> Data, ref string errors)
        {
            StringBuilder errorBuilder = new StringBuilder(errors);
            Dictionary<string, int> authorCount = new Dictionary<string,
int>(StringComparer.OrdinalIgnoreCase);
            try
            {
                foreach (StudentAssociation list in Data)
                {
                    if (list == null) continue;

                    foreach (Question question in list)
                    {
                        try
                        {
                            if (question?.Author == null) continue;

                            string author = question.Author.Trim();
                            if (!authorCount.ContainsKey(author))
                            {
                                authorCount.Add(author, 1);
                            }
                            else
                            {
                                authorCount[author] += 1;
                            }
                        }
                        catch (Exception ex)
                        {
                            errorBuilder.AppendLine($"Error in StudentAssociation {list.name}:
{ex.Message}");
                        }
                    }
                }
            }
            catch (Exception ex)
            {
                errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
            }
            errors = errorBuilder.ToString();
            return authorCount;
        }

        public static Dictionary<string, int> maxAuthor(Dictionary<string, int> data, ref
string errors)
        {
            StringBuilder errorBuilder = new StringBuilder(errors);
            Dictionary<string, int> answer = new Dictionary<string, int>();
            try
            {
                int max = data.Values.Max();

                foreach (var item in data)
                {
                    try
                    {
                        if (item.Value == max)
                        {
                            answer.Add(item.Key, item.Value);
                        }
                    }
                    catch (Exception ex)
                    {
                        errorBuilder.AppendLine($"Error processing author {item.Key}:
{ex.Message}");
                    }
                }
            }
            catch (Exception ex)
            {
                errorBuilder.AppendLine($"Unexpected error: {ex.Message}");
            }
            errors = errorBuilder.ToString();
            return answer;
        }


        /// <summary >
        /// Sort the list of questions by theme and difficulty using bubble sort
        /// </summary >
        /// <param name="list" > </param >
        public static void BubbleSort(this List<Question> list)
        {
            for (int i = 0; i < list.Count - 1; i++)

            {
                for (int j = 0; j < list.Count - 1 - i; j++)

                {
                    if (list[j].CompareTo(list[j + 1]) < 0)

                    {
                        Question temp = list[j];
                        list[j] = list[j + 1];
                        list[j + 1] = temp;
                    }

                }
            }
        }
    }
}
```

```
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
```

```css
}

.container {
    width: 80%;
    margin: 20px auto;
    padding: 20px;
    background-color: white;
    border: 1px solid #ccc;
    height: 1000px;
}

.label {
    font-weight: bold;
}

table {
    width: 90%;
    max-width: 1800px;
    table-layout: fixed;
    border-collapse: collapse;
    margin: 0 auto 10px auto;
    overflow-x: auto;
}

    table th, table td {
        border: 1px solid #ddd;
        text-align: center;
        padding: 8px;
        word-wrap: break-word;
        white-space: normal;
    }

    table th {
        background-color: #f2f2f2;
        font-weight: bold;
    }

    table + table {
        margin-top: 10px;
    }

.textbox {
    padding: 5px;
    border: 1px solid #ccc;
}

.button {
    padding: 10px 20px;
    background-color: #007bff;
    color: white;
    border: none;
    cursor: pointer;
}

    .button:hover {
        background-color: #0056b3;
    }

.custom-validator {
    color: red;
}
```

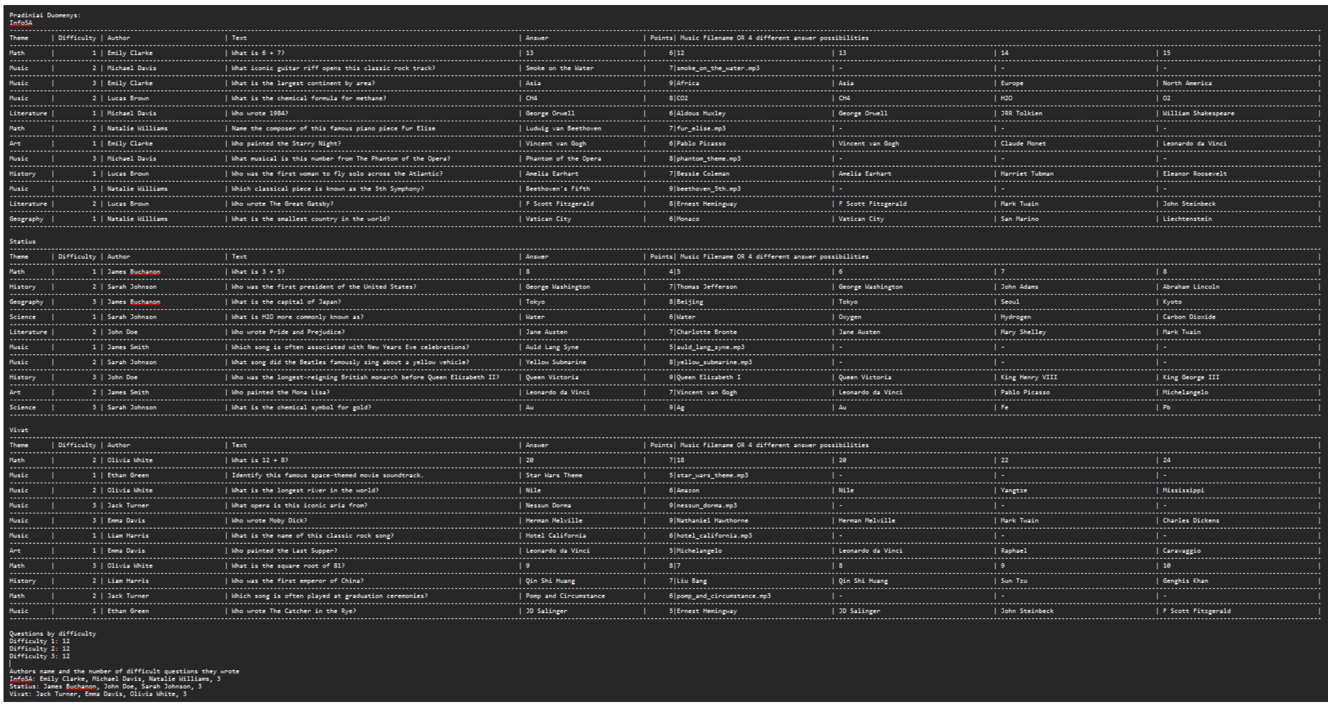
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Web.aspx.cs" Inherits="L4.Web"
%>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link rel="stylesheet" type="text/css" href="StyleSheet.css" />
</head>
<body>
    <form id="form1" runat="server">
        <div style="height: 600px">

            <asp:Label ID="Label1" runat="server" Text="Initial data"
CssClass="label"></asp:Label>
            <br />
            <br />
            <br />
            <asp:Panel ID="TablesContainer1" runat="server"></asp:Panel>
            <br />

            <asp:Button ID="Button1" runat="server" Text="Calculate"
OnClick="Button1_Click" />

            <br />
            <br />
            <asp:Label ID="Label2" runat="server" Text=""></asp:Label>
            <br />
            <asp:Label ID="Label3" runat="server" Text=""></asp:Label>
            <br />
            <asp:Label ID="Label4" runat="server" Text=""></asp:Label>
            <br />
            <asp:Label ID="Label5" runat="server" Text=""></asp:Label>


            <br />
            <br />
            <asp:Label ID="Label6" runat="server" Text=""></asp:Label>
            <br />
            <asp:Label ID="Label7" runat="server" Text=""></asp:Label>
            <br />
        </div>
    </form>
</body>
</html>

namespace L4
{
    public partial class Web : System.Web.UI.Page
    {
        /// <summary>
        /// generic add to tables method
        /// </summary>
        /// <param name="Data"></param>
        /// <param name="TablesContainer"></param>
        protected void AddToTables(List<StudentAssociation> Data, ref Panel TablesContainer)
        {
            foreach (var studentAssociation in Data)
            {
                Table table = new Table
                {
                    CssClass = "table"
                };
                Label label = new Label
                {
                    Text = studentAssociation.name,
                    CssClass = "table-header"
                };

                TableHeaderRow headerRow = new TableHeaderRow();
                headerRow.Cells.Add(new TableHeaderCell { Text = "Theme" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Difficulty" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Author" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Text" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Answer" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Points" });
                TableHeaderCell mergedHeaderCell = new TableHeaderCell
                {
                    Text = "Music Filename OR 4 different answer possibilities",
                    ColumnSpan = 4,
                    HorizontalAlign = HorizontalAlign.Center
                };
                headerRow.Cells.Add(mergedHeaderCell);
                table.Rows.Add(headerRow);

                foreach (var question in studentAssociation)
                {

                    TableRow row = question.ToTableRow();

                    table.Rows.Add(row);
                }

                TablesContainer.Controls.Add(label);
                TablesContainer.Controls.Add(table);
            }
        }

        /// <summary>
        /// Writes the number of questions by difficulty into the labels
        /// </summary>
        /// <param name="label2"></param>
        /// <param name="label3"></param>
        /// <param name="label4"></param>
        /// <param name="label5"></param>
        /// <param name="levelOneHardness"></param>
        /// <param name="levelTwoHardness"></param>
        /// <param name="levelThreeHardness"></param>
        public static void WriteDifficultiesIntoLabels(ref Label label2, ref Label label3, ref
    Label label4, ref Label label5, int levelOneHardness, int levelTwoHardness, int
    levelThreeHardness)
        {
            label2.Text = "Questions by difficulty";
            if (levelOneHardness > 0)
            {
                label3.Text = "Difficulty 1: " + levelOneHardness.ToString();
            }
            else
            {
                label3.Text = "Difficulty 1: No questions";
            }

            if (levelTwoHardness > 0)
            {
                label4.Text = "Difficulty 2: " + levelTwoHardness.ToString();
            }
            else
            {
                label4.Text = "Difficulty 2: No questions";
            }
            if (levelThreeHardness > 0)
            {
                label5.Text = "Difficulty 3: " + levelThreeHardness.ToString();
            }
            else
            {
                label5.Text = "Difficulty 3: No questions";
            }

        }

        /// <summary>
        /// Writes the authors and the number of difficult questions they wrote into the
labels
        /// </summary>
        /// <param name="label6"></param>
        /// <param name="label7"></param>
        /// <param name="data"></param>
        public static void WriteAuthorsMaxCount(ref Label label6, ref Label label7,
Dictionary<string, int> data)
        {
            label6.Text = "Authors name and the number of difficult questions they wrote";
            string answer = "";
            foreach (var item in data)
            {
                answer += $"{item.Key} {item.Value} <br />";
            }
            label7.Text = answer;

        }


        /// <summary>
        /// checks if any error is triggered and then clears the labels and the table
        /// </summary>
        /// <param name="ErrorLabel"></param>
        /// <param name="Label1"></param>
        /// <param name="Label2"></param>
        /// <param name="Label3"></param>
        /// <param name="Label4"></param>
        /// <param name="Label5"></param>
        /// <param name="Label6"></param>
        /// <param name="Label7"></param>
        /// <param name="TablesContainer1"></param>
        private static void Checking(Label ErrorLabel, Label Label1, Label Label2, Label
Label3, Label Label4, Label Label5, Label Label6, Label Label7, Panel TablesContainer1)
        {
            if (ErrorLabel.Text != string.Empty)
            {
                // Clear all labels
                Label1.Text = string.Empty;
                Label2.Text = string.Empty;
                Label3.Text = string.Empty;
                Label4.Text = string.Empty;
                Label5.Text = string.Empty;
                Label6.Text = string.Empty;
                Label7.Text = string.Empty;

                // Clear and hide the table panel
                TablesContainer1.Controls.Clear();
                TablesContainer1.Visible = false;
            }
        }
    }
}

namespace L4
```

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

        }

        protected void Button1_Click(object sender, EventArgs e)
        {

            string CIN = Server.MapPath(@"App_Data/Var2");
            string COUT = Server.MapPath(@"App_Data");
            string log = Server.MapPath(@"App_Data");
            File.Delete(COUT + @"\Output.txt");
            File.Delete(COUT + @"\SudėtingiMuzikiniai.csv");
            File.Delete(COUT + @"\SudėtingiBendrai.csv");

            string errors = string.Empty;


            List<StudentAssociation> MainDataList = InOut.ReadData(CIN, ref errors);
            int levelOneHardness = 0;
            int levelTwoHardness = 0;
            int levelThreeHardness = 0;

            InOut.PrintDataInitial(COUT, "Pradiniai Duomenys:", MainDataList);

            TaskUtils.FindDifficultyCount(ref levelOneHardness, ref levelTwoHardness, ref
levelThreeHardness, MainDataList, ref errors);

            AddToTables(MainDataList, ref TablesContainer1);

            int MaxDifficulty = 3;

            Dictionary<string, int> MentionedAuthorCount =
TaskUtils.FindMostMentionedAuthors(MainDataList, ref errors);
            Dictionary<string, int> MostMentionedAuthors =
TaskUtils.maxAuthor(MentionedAuthorCount, ref errors);

            List<StudentAssociation> MaxDifficultyList =
TaskUtils.FindMaxDifficultyList(MainDataList, MaxDifficulty, ref errors);



            WriteDifficultiesIntoLabels(ref Label2, ref Label3, ref Label4, ref Label5,
levelOneHardness, levelTwoHardness, levelThreeHardness);
            WriteAuthorsMaxCount(ref Label6, ref Label7, MostMentionedAuthors);

            List<StudentAssociation> MaxDifficultyListMusical =
TaskUtils.FindMaxDifficultyMusical(MaxDifficultyList, ref errors);

            string SpecifiedTheme = "Music";
            File.Delete(COUT + $@"\{SpecifiedTheme}.csv");

            List<Question> ThemeList = TaskUtils.FindQuestionsByTheme(MainDataList,
SpecifiedTheme, ref errors);

            ThemeList.BubbleSort();

            InOut.PrintQuestionsByDifficulty(COUT, levelOneHardness, levelTwoHardness,
levelThreeHardness);

            InOut.PrintMaxAuthors(COUT, MaxDifficultyList, MaxDifficulty);

            InOut.PrintToCSV(COUT, MaxDifficultyListMusical, "SudėtingiMuzikiniai.csv",
"Sudetingi muzikiniai");
```

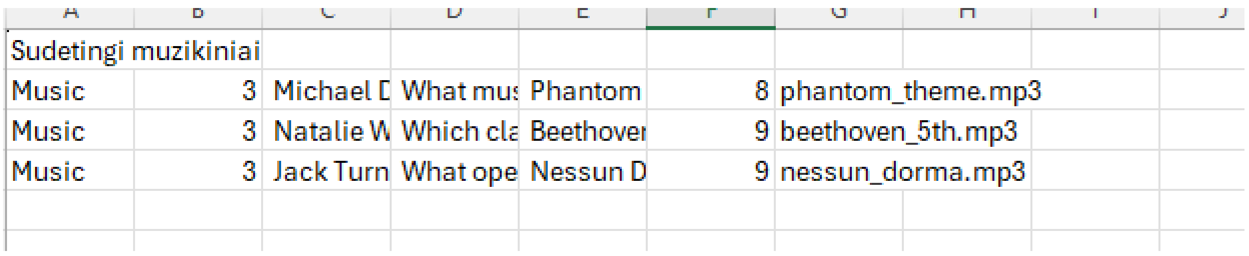
```
            InOut.PrintToCSV(COUT, MaxDifficultyList, "SudėtingiBendrai.csv", "Sudetingi
Bendrai");

            InOut.PrintToCSVSorted(COUT, ThemeList, $"{SpecifiedTheme}.csv", "Surikiuoti");

            ErrorLabel.Text = errors;

            Checking(ErrorLabel, Label1, Label2, Label3, Label4, Label5, Label6, Label7,
TablesContainer1);



        }
    }
}
```

4.7. Pradiniai duomenys ir rezultatai

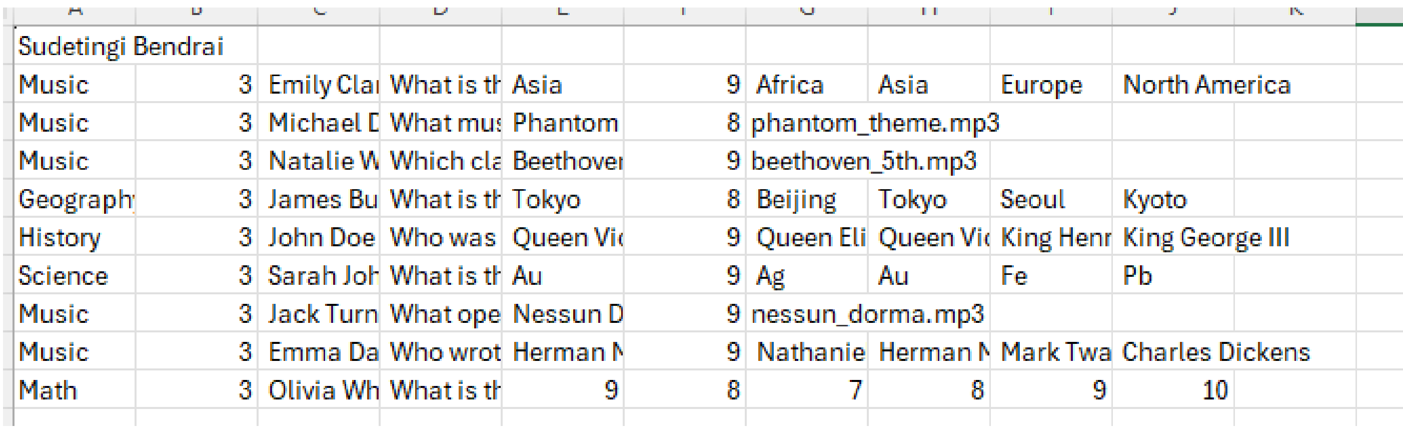
I



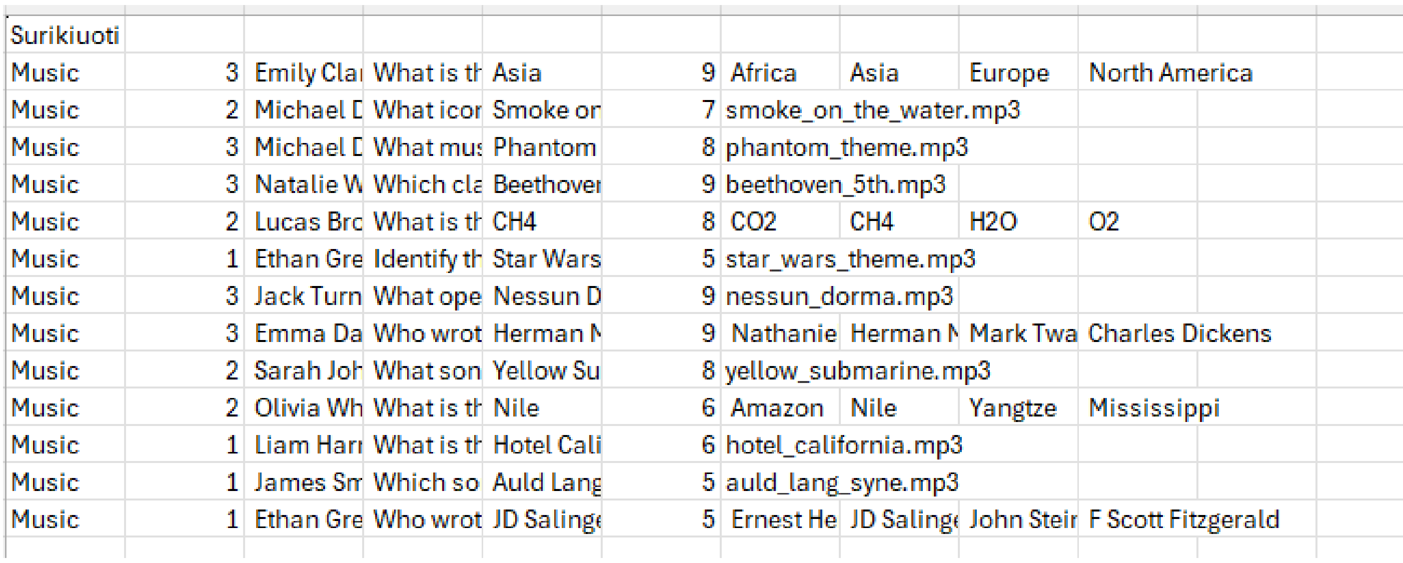
```
Pradiniai Duomenys:
InfoSA
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author           | Text                                                          | Answer                  | Points| Music Filename OR 4 different answer possibilities                                      |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          1 | Emily Clarke     | What is 6 + 7?                                                | 13                      |      6|12                     | 13                    | 14                    | 15                        |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Michael Davis    | What iconic guitar riff opens this classic rock track?        | Smoke on the Water      |      7|smoke_on_the_water.mp3 | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Emily Clarke     | What is the largest continent by area?                        | Asia                    |      9|Africa                 | Asia                  | Europe                | North America             |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Lucas Brown      | What is the chemical formula for methane?                     | CH4                     |      8|CO2                    | CH4                   | H2O                   | O2                        |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          1 | Michael Davis    | Who wrote 1984?                                               | George Orwell           |      6|Aldous Huxley          | George Orwell         | JRR Tolkien           | William Shakespeare       |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          2 | Natalie Williams | Name the composer of this famous piano piece Fur Elise        | Ludwig van Beethoven    |      7|fur_elise.mp3          | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Art        |          1 | Emily Clarke     | Who painted the Starry Night?                                 | Vincent van Gogh        |      6|Pablo Picasso          | Vincent van Gogh      | Claude Monet          | Leonardo da Vinci         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Michael Davis    | What musical is this number from The Phantom of the Opera?    | Phantom of the Opera    |      8|phantom_theme.mp3      | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          1 | Lucas Brown      | Who was the first woman to fly solo across the Atlantic?      | Amelia Earhart          |      7|Bessie Coleman         | Amelia Earhart        | Harriet Tubman        | Eleanor Roosevelt         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Natalie Williams | Which classical piece is known as the 5th Symphony?           | Beethoven's Fifth       |      9|beethoven_5th.mp3      | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          2 | Lucas Brown      | Who wrote The Great Gatsby?                                   | F Scott Fitzgerald      |      8|Ernest Hemingway       | F Scott Fitzgerald    | Mark Twain            | John Steinbeck            |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Geography  |          1 | Natalie Williams | What is the smallest country in the world?                    | Vatican City            |      6|Monaco                 | Vatican City          | San Marino            | Liechtenstein             |
--------------------------------------------------------------------------------------------------------------------------------------------------------------

Statius
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author           | Text                                                          | Answer                  | Points| Music Filename OR 4 different answer possibilities                                      |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          1 | James Buchanon   | What is 3 + 5?                                                | 8                       |      4|5                      | 6                     | 7                     | 8                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          2 | Sarah Johnson    | Who was the first president of the United States?             | George Washington       |      7|Thomas Jefferson       | George Washington     | John Adams            | Abraham Lincoln           |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Geography  |          3 | James Buchanon   | What is the capital of Japan?                                 | Tokyo                   |      8|Beijing                | Tokyo                 | Seoul                 | Kyoto                     |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Science    |          1 | Sarah Johnson    | What is H2O more commonly known as?                           | Water                   |      6|Water                  | Oxygen                | Hydrogen              | Carbon Dioxide            |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          2 | John Doe         | Who wrote Pride and Prejudice?                                | Jane Austen             |      7|Charlotte Bronte       | Jane Austen           | Mary Shelley          | Mark Twain                |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          1 | James Smith      | Which song is often associated with New Years Eve celebrations? | Auld Lang Syne        |      5|auld_lang_syne.mp3     | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Sarah Johnson    | What song did the Beatles famously sing about a yellow vehicle? | Yellow Submarine      |      8|yellow_submarine.mp3   | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          3 | John Doe         | Who was the longest-reigning British monarch before Queen Elizabeth II? | Queen Victoria |      9|Queen Elizabeth I      | Queen Victoria        | King Henry VIII       | King George III           |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Art        |          2 | James Smith      | Who painted the Mona Lisa?                                    | Leonardo da Vinci       |      7|Vincent van Gogh       | Leonardo da Vinci     | Pablo Picasso         | Michelangelo              |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Science    |          3 | Sarah Johnson    | What is the chemical symbol for gold?                         | Au                      |      9|Ag                     | Au                    | Fe                    | Pb                        |
--------------------------------------------------------------------------------------------------------------------------------------------------------------

Vivat
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author           | Text                                                          | Answer                  | Points| Music Filename OR 4 different answer possibilities                                      |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          2 | Olivia White     | What is 12 + 8?                                               | 20                      |      7|18                     | 20                    | 22                    | 24                        |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          1 | Ethan Green      | Identify this famous space-themed movie soundtrack.           | Star Wars Theme         |      5|star_wars_theme.mp3    | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Olivia White     | What is the longest river in the world?                       | Nile                    |      6|Amazon                 | Nile                  | Yangtze               | Mississippi               |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Jack Turner      | What opera is this iconic aria from?                          | Nessun Dorma            |      9|nessun_dorma.mp3       | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Emma Davis       | Who wrote Moby Dick?                                          | Herman Melville         |      9|Nathaniel Hawthorne    | Herman Melville       | Mark Twain            | Charles Dickens           |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          1 | Liam Harris      | What is the name of this classic rock song?                   | Hotel California        |      6|hotel_california.mp3   | -                     | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Art        |          1 | Emma Davis       | Who painted the Last Supper?                                  | Leonardo da Vinci       |      5|Michelangelo           | Leonardo da Vinci     | Raphael               | Caravaggio                |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          3 | Olivia White     | What is the square root of 81?                                | 9                       |      8|7                      | 8                     | 9                     | 10                        |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          2 | Liam Harris      | Who was the first emperor of China?                           | Qin Shi Huang           |      7|Liu Bang               | Qin Shi Huang         | Sun Tzu               | Genghis Khan              |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          2 | Jack Turner      | Which song is often played at graduation ceremonies?          | Pomp and Circumstance   |      6|pomp_and_circumstance.mp3 | -                  | -                     | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          1 | Ethan Green      | Who wrote The Catcher in the Rye?                             | JD Salinger             |      5|Ernest Hemingway       | JD Salinger           | John Steinbeck        | F Scott Fitzgerald        |
--------------------------------------------------------------------------------------------------------------------------------------------------------------

Questions by difficulty
Difficulty 1: 12
Difficulty 2: 12
Difficulty 3: 12
|
Authors name and the number of difficult questions they wrote
InfoSA: Emily Clarke, Michael Davis, Natalie Williams, 3
Statius: James Buchanon, John Doe, Sarah Johnson, 3
Vivat: Jack Turner, Emma Davis, Olivia White, 3
```



A	B	C	D	E	F	G	H	I	J
Sudetingi muzikiniai									
Music	3	Michael D	What mus	Phantom	8	phantom_theme.mp3			
Music	3	Natalie W	Which cla	Beethover	9	beethoven_5th.mp3			
Music	3	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3			

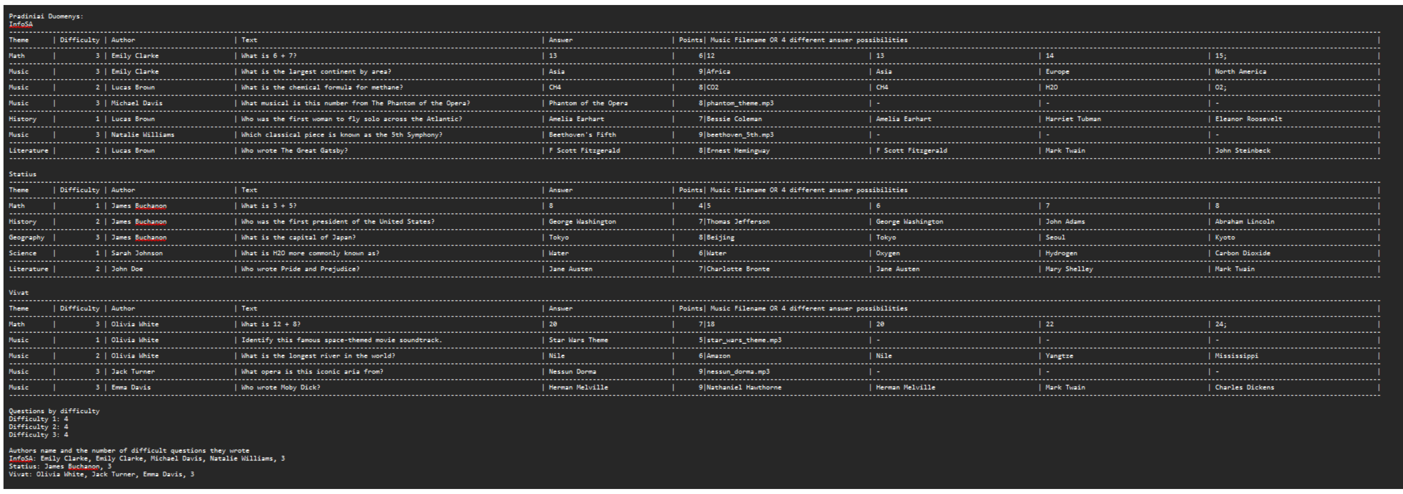


A	B	C	D	E	F	G	H	I	J	K
Sudetingi Bendrai										
Music	3	Emily Clar	What is th	Asia	9	Africa	Asia	Europe	North America	
Music	3	Michael D	What mus	Phantom	8	phantom_theme.mp3				
Music	3	Natalie W	Which cla	Beethover	9	beethoven_5th.mp3				
Geograph	3	James Bu	What is th	Tokyo	8	Beijing	Tokyo	Seoul	Kyoto	
History	3	John Doe	Who was	Queen Vic	9	Queen Eli	Queen Vic	King Henr	King George III	
Science	3	Sarah Joh	What is th	Au	9	Ag	Au	Fe	Pb	
Music	3	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3				
Music	3	Emma Da	Who wrot	Herman M	9	Nathanie	Herman M	Mark Twa	Charles Dickens	
Math	3	Olivia Wh	What is th	9	8	7	8	9	10	

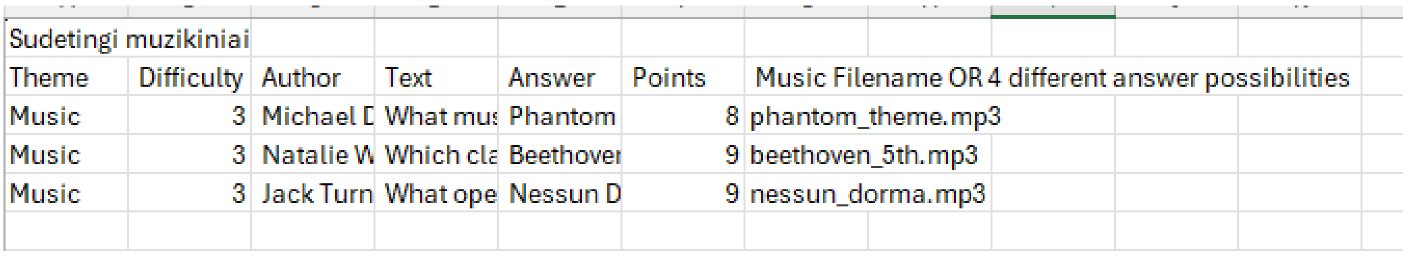


Surikiuoti									
Music	3	Emily Clai	What is th	Asia	9	Africa	Asia	Europe	North America
Music	2	Michael D	What icon	Smoke on	7	smoke_on_the_water.mp3			
Music	3	Michael D	What mus	Phantom	8	phantom_theme.mp3			
Music	3	Natalie W	Which cla	Beethover	9	beethoven_5th.mp3			
Music	2	Lucas Brc	What is th	CH4	8	CO2	CH4	H2O	O2
Music	1	Ethan Gre	Identify th	Star Wars	5	star_wars_theme.mp3			
Music	3	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3			
Music	3	Emma Da	Who wrot	Herman M	9	Nathanie	Herman M	Mark Twa	Charles Dickens
Music	2	Sarah Joh	What son	Yellow Su	8	yellow_submarine.mp3			
Music	2	Olivia Wh	What is th	Nile	6	Amazon	Nile	Yangtze	Mississippi
Music	1	Liam Harı	What is th	Hotel Cali	6	hotel_california.mp3			
Music	1	James Sm	Which so	Auld Lang	5	auld_lang_syne.mp3			
Music	1	Ethan Gre	Who wrot	JD Salinge	5	Ernest He	JD Salinge	John Steir	F Scott Fitzgerald

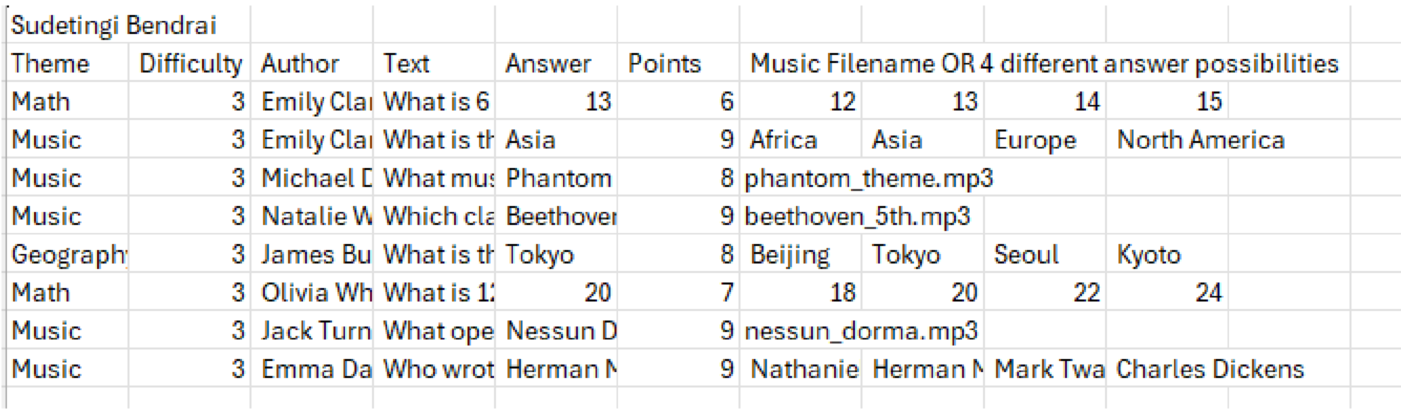
II



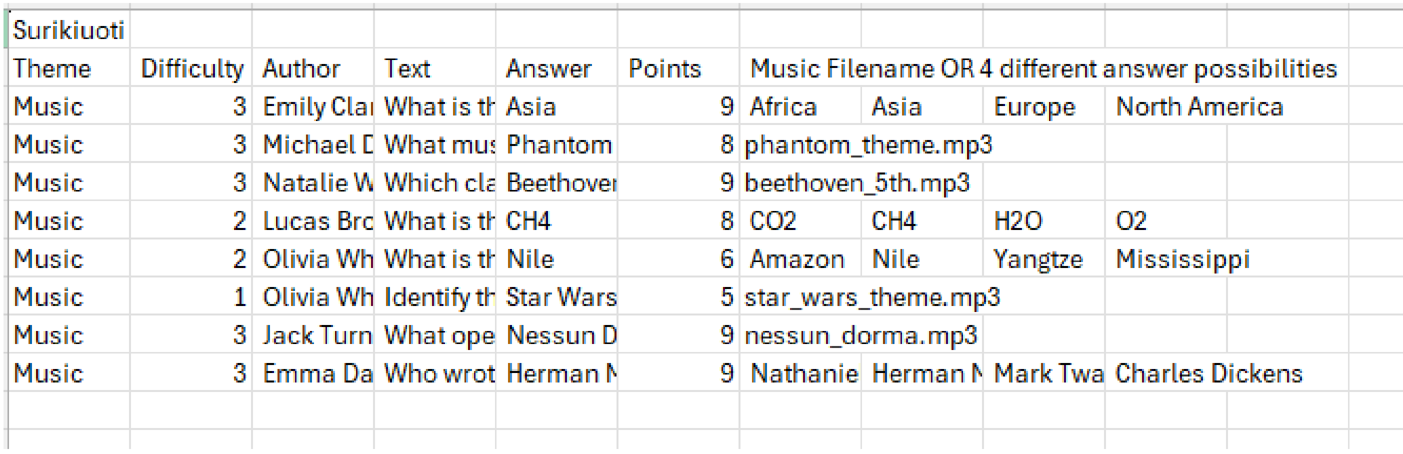
```
Pradiniai Duomenys:
InfoSA
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author                | Text                                                   | Answer                 | Points| Music Filename OR 4 different answer possibilities                                                      |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          3 | Emily Clarke          | What is 6 + 7?                                         | 13                     |      6|12                        | 13                        | 14                        | 15;                       |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Emily Clarke          | What is the largest continent by area?                 | Asia                   |      9|Africa                    | Asia                      | Europe                    | North America             |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Lucas Brown           | What is the chemical formula for methane?              | CH4                    |      8|CO2                       | CH4                       | H2O                       | O2;                       |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Michael Davis         | What musical is this number from The Phantom of the Opera? | Phantom of the Opera |      8|phantom_theme.mp3         | -                         | -                         | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          1 | Lucas Brown           | Who was the first woman to fly solo across the Atlantic? | Amelia Earhart       |      7|Bessie Coleman            | Amelia Earhart            | Harriet Tubman            | Eleanor Roosevelt         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Natalie Williams      | Which classical piece is known as the 5th Symphony?    | Beethoven's Fifth      |      9|beethoven_5th.mp3         | -                         | -                         | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          2 | Lucas Brown           | Who wrote The Great Gatsby?                            | F Scott Fitzgerald     |      8|Ernest Hemingway          | F Scott Fitzgerald        | Mark Twain                | John Steinbeck            |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Statius
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author                | Text                                                   | Answer                 | Points| Music Filename OR 4 different answer possibilities                                                      |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          1 | James Buchanon        | What is 3 + 5?                                         | 8                      |      4|5                         | 6                         | 7                         | 8                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          2 | James Buchanon        | Who was the first president of the United States?      | George Washington      |      7|Thomas Jefferson          | George Washington         | John Adams                | Abraham Lincoln           |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Geography  |          3 | James Buchanon        | What is the capital of Japan?                          | Tokyo                  |      8|Beijing                   | Tokyo                     | Seoul                     | Kyoto                     |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Science    |          1 | Sarah Johnson         | What is H2O more commonly known as?                    | Water                  |      6|Water                     | Oxygen                    | Hydrogen                  | Carbon Dioxide            |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          2 | John Doe              | Who wrote Pride and Prejudice?                         | Jane Austen            |      7|Charlotte Bronte          | Jane Austen               | Mary Shelley              | Mark Twain                |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Vivat
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author                | Text                                                   | Answer                 | Points| Music Filename OR 4 different answer possibilities                                                      |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          3 | Olivia White          | What is 12 + 8?                                        | 20                     |      7|18                        | 20                        | 22                        | 24;                       |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          1 | Olivia White          | Identify this famous space-themed movie soundtrack.    | Star Wars Theme        |      5|star_wars_theme.mp3       | -                         | -                         | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Olivia White          | What is the longest river in the world?                | Nile                   |      6|Amazon                    | Nile                      | Yangtze                   | Mississippi               |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Jack Turner           | What opera is this iconic aria from?                   | Nessun Dorma           |      9|nessun_dorma.mp3          | -                         | -                         | -                         |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          3 | Emma Davis            | Who wrote Moby Dick?                                   | Herman Melville        |      9|Nathaniel Hawthorne       | Herman Melville           | Mark Twain                | Charles Dickens           |
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Questions by difficulty
Difficulty 1: 4
Difficulty 2: 4
Difficulty 3: 4

Authors name and the number of difficult questions they wrote
InfoSA: Emily Clarke, Emily Clarke, Michael Davis, Natalie Williams, 3
Statius: James Buchanon, 3
Vivat: Olivia White, Jack Turner, Emma Davis, 3
```



Sudetingi muzikiniai									
Theme	Difficulty	Author	Text	Answer	Points	Music Filename OR 4 different answer possibilities			
Music	3	Michael D	What mus	Phantom	8	phantom_theme.mp3			
Music	3	Natalie W	Which cla	Beethover	9	beethoven_5th.mp3			
Music	3	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3			

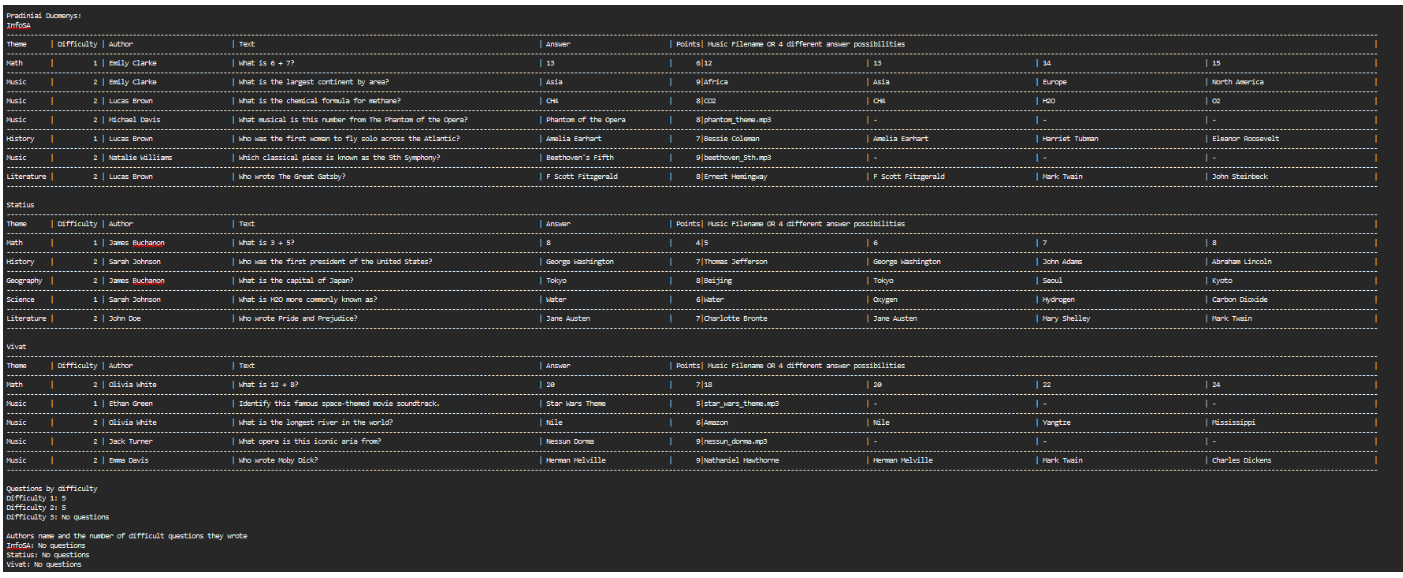


Sudetingi Bendrai									
Theme	Difficulty	Author	Text	Answer	Points	Music Filename OR 4 different answer possibilities			
Math	3	Emily Clai	What is 6	13	6	12	13	14	15
Music	3	Emily Clai	What is th	Asia	9	Africa	Asia	Europe	North America
Music	3	Michael D	What mus	Phantom	8	phantom_theme.mp3			
Music	3	Natalie W	Which cla	Beethover	9	beethoven_5th.mp3			
Geograph	3	James Bu	What is th	Tokyo	8	Beijing	Tokyo	Seoul	Kyoto
Math	3	Olivia Wh	What is 1:	20	7	18	20	22	24
Music	3	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3			
Music	3	Emma Da	Who wrot	Herman M	9	Nathanie	Herman M	Mark Twa	Charles Dickens

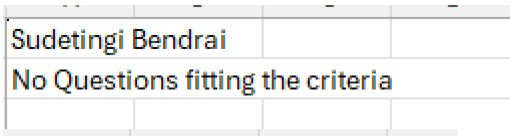


Surikiuoti									
Theme	Difficulty	Author	Text	Answer	Points	Music Filename OR 4 different answer possibilities			
Music	3	Emily Clai	What is th	Asia	9	Africa	Asia	Europe	North America
Music	3	Michael D	What mus	Phantom	8	phantom_theme.mp3			
Music	3	Natalie W	Which cla	Beethovei	9	beethoven_5th.mp3			
Music	2	Lucas Brc	What is th	CH4	8	CO2	CH4	H2O	O2
Music	2	Olivia Wh	What is th	Nile	6	Amazon	Nile	Yangtze	Mississippi
Music	1	Olivia Wh	Identify th	Star Wars	5	star_wars_theme.mp3			
Music	3	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3			
Music	3	Emma Da	Who wrot	Herman M	9	Nathanie	Herman M	Mark Twa	Charles Dickens

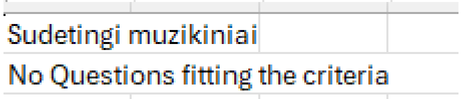
III



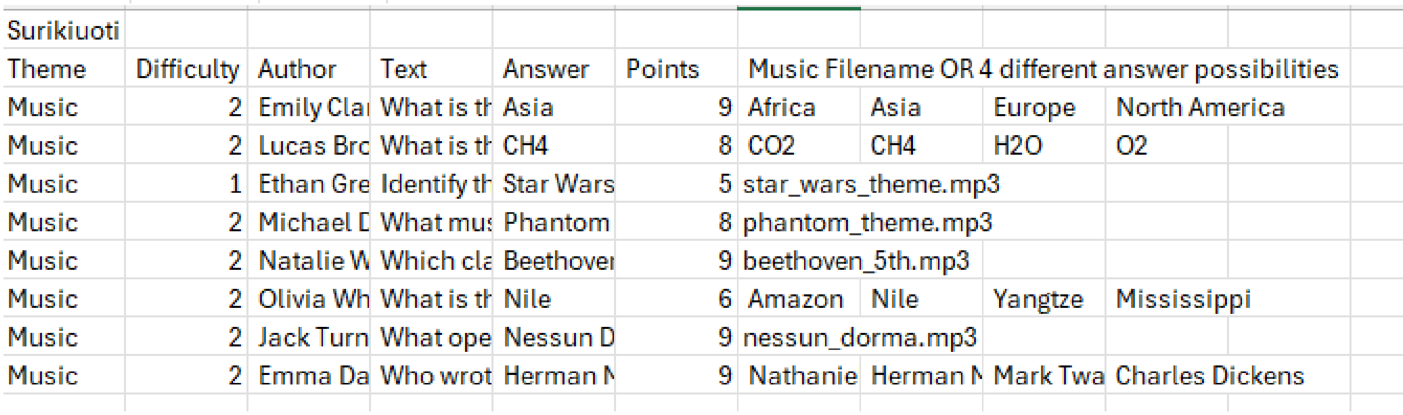
```
Pradiniai Duomenys:
InfoSA
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author          | Text                                                   | Answer               | Points| Music Filename OR 4 different answer possibilities                         |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          1 | Emily Clarke    | What is 6 + 7?                                         | 13                   |      6|12                    | 13                   | 14                   | 15                   |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Emily Clarke    | What is the largest continent by area?                 | Asia                 |      9|Africa                | Asia                 | Europe               | North America        |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Lucas Brown     | What is the chemical formula for methane?              | CH4                  |      8|CO2                   | CH4                  | H2O                  | O2                   |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Michael Davis   | What musical is this number from The Phantom of the Opera? | Phantom of the Opera |      8|phantom_theme.mp3     | -                    | -                    | -                    |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          1 | Lucas Brown     | Who was the first woman to fly solo across the Atlantic? | Amelia Earhart       |      7|Bessie Coleman        | Amelia Earhart       | Harriet Tubman       | Eleanor Roosevelt    |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Natalie Williams | Which classical piece is known as the 5th Symphony?   | Beethoven's Fifth    |      9|beethoven_5th.mp3     | -                    | -                    | -                    |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          2 | Lucas Brown     | Who wrote The Great Gatsby?                            | F Scott Fitzgerald   |      8|Ernest Hemingway      | F Scott Fitzgerald   | Mark Twain           | John Steinbeck       |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Statius
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author          | Text                                                   | Answer               | Points| Music Filename OR 4 different answer possibilities                         |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          1 | James Buchanon  | What is 3 + 5?                                         | 8                    |      4|5                     | 6                    | 7                    | 8                    |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
History    |          2 | Sarah Johnson   | Who was the first president of the United States?      | George Washington    |      7|Thomas Jefferson      | George Washington    | John Adams           | Abraham Lincoln      |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Geography  |          2 | James Buchanon  | What is the capital of Japan?                          | Tokyo                |      8|Beijing               | Tokyo                | Seoul                | Kyoto                |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Science    |          1 | Sarah Johnson   | What is H2O more commonly known as?                    | Water                |      6|Water                 | Oxygen               | Hydrogen             | Carbon Dioxide       |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Literature |          2 | John Doe        | Who wrote Pride and Prejudice?                         | Jane Austen          |      7|Charlotte Bronte      | Jane Austen          | Mary Shelley         | Mark Twain           |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Vivat
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Theme      | Difficulty | Author          | Text                                                   | Answer               | Points| Music Filename OR 4 different answer possibilities                         |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Math       |          2 | Olivia White    | What is 12 + 8?                                        | 20                   |      7|18                    | 20                   | 22                   | 24                   |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          1 | Ethan Green     | Identify this famous space-themed movie soundtrack.    | Star Wars Theme      |      5|star_wars_theme.mp3   | -                    | -                    | -                    |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Olivia White    | What is the longest river in the world?                | Nile                 |      6|Amazon                | Nile                 | Yangtze              | Mississippi          |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Jack Turner     | What opera is this iconic aria from?                   | Nessun Dorma         |      9|nessun_dorma.mp3      | -                    | -                    | -                    |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------
Music      |          2 | Emma Davis      | Who wrote Moby Dick?                                   | Herman Melville      |      9|Nathaniel Hawthorne   | Herman Melville      | Mark Twain           | Charles Dickens      |
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Questions by difficulty
Difficulty 1: 5
Difficulty 2: 5
Difficulty 3: No questions

Authors name and the number of difficult questions they wrote
InfoSA: No questions
Statius: No questions
Vivat: No questions
```



Sudetingi Bendrai
No Questions fitting the criteria



Sudetingi muzikiniai
No Questions fitting the criteria



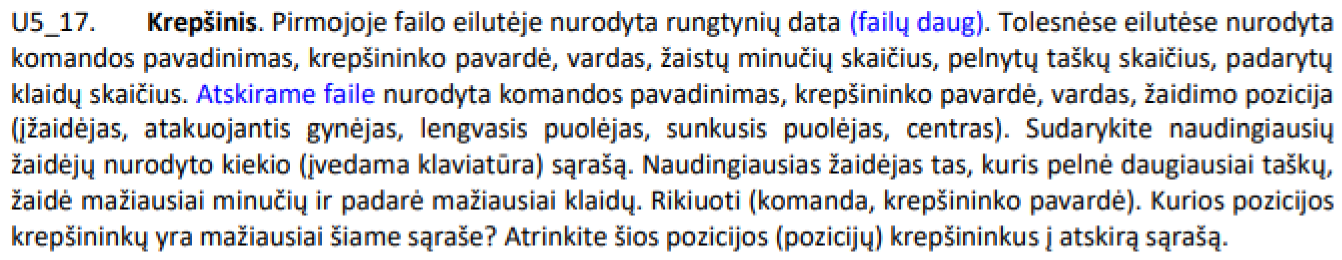
Surikiuoti									
Theme	Difficulty	Author	Text	Answer	Points	Music Filename OR 4 different answer possibilities			
Music	2	Emily Clai	What is th	Asia	9	Africa	Asia	Europe	North America
Music	2	Lucas Brc	What is th	CH4	8	CO2	CH4	H2O	O2
Music	1	Ethan Gre	Identify th	Star Wars	5	star_wars_theme.mp3			
Music	2	Michael D	What mus	Phantom	8	phantom_theme.mp3			
Music	2	Natalie W	Which cla	Beethovei	9	beethoven_5th.mp3			
Music	2	Olivia Wh	What is th	Nile	6	Amazon	Nile	Yangtze	Mississippi
Music	2	Jack Turn	What ope	Nessun D	9	nessun_dorma.mp3			
Music	2	Emma Da	Who wrot	Herman M	9	Nathanie	Herman M	Mark Twa	Charles Dickens

4.8. Dėstytojo pastabos

Sutvarkyti throw errors, kad nepalūžtų svetainė.

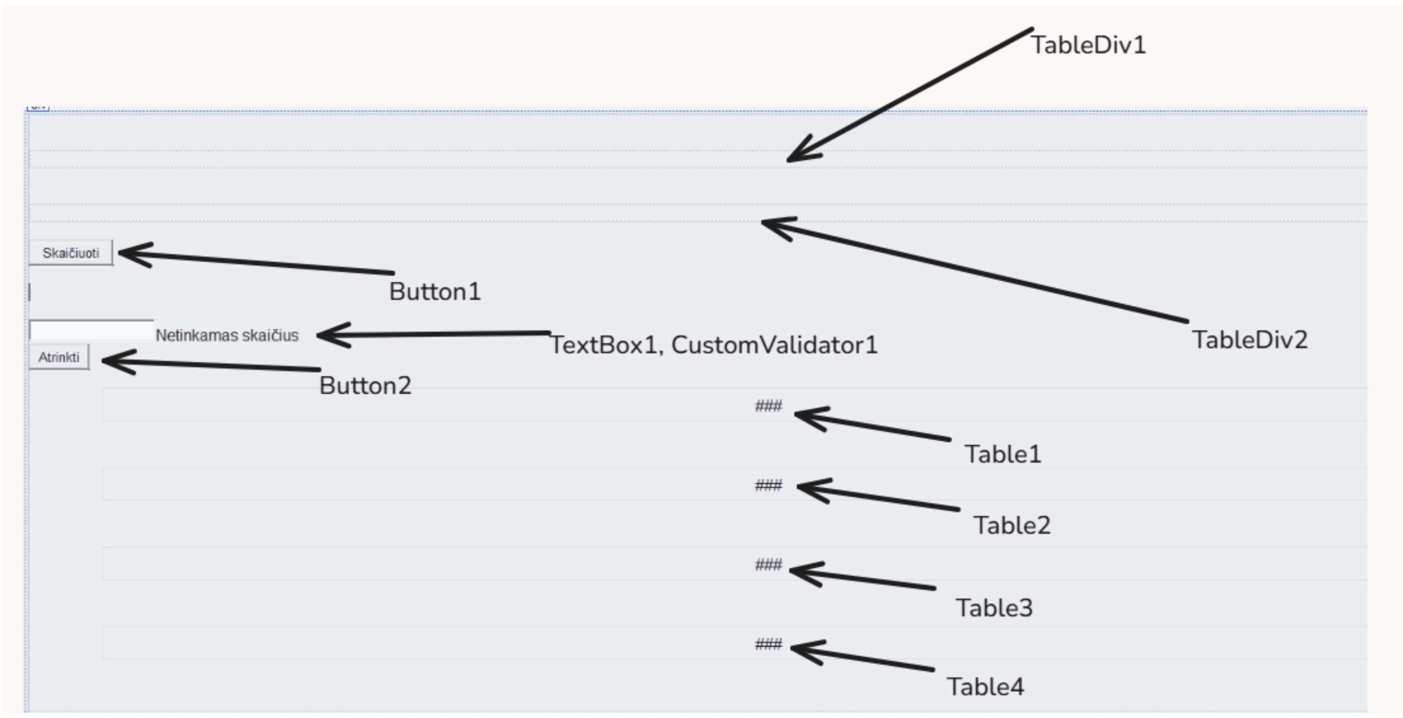
5. Deklaratyvusis programavimas (L5)

5.1. Darbo užduotis



U5_17. **Krepšinis**. 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 (įžaidėjas, atakuojantis gynėjas, lengvasis puolėjas, sunkusis puolėjas, centras). Sudarykite 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). Kurios pozicijos krepšininkų yra mažiausiai šiame sąraše? Atrinkite šios pozicijos (pozicijų) krepšininkus į atskirą sąrašą.

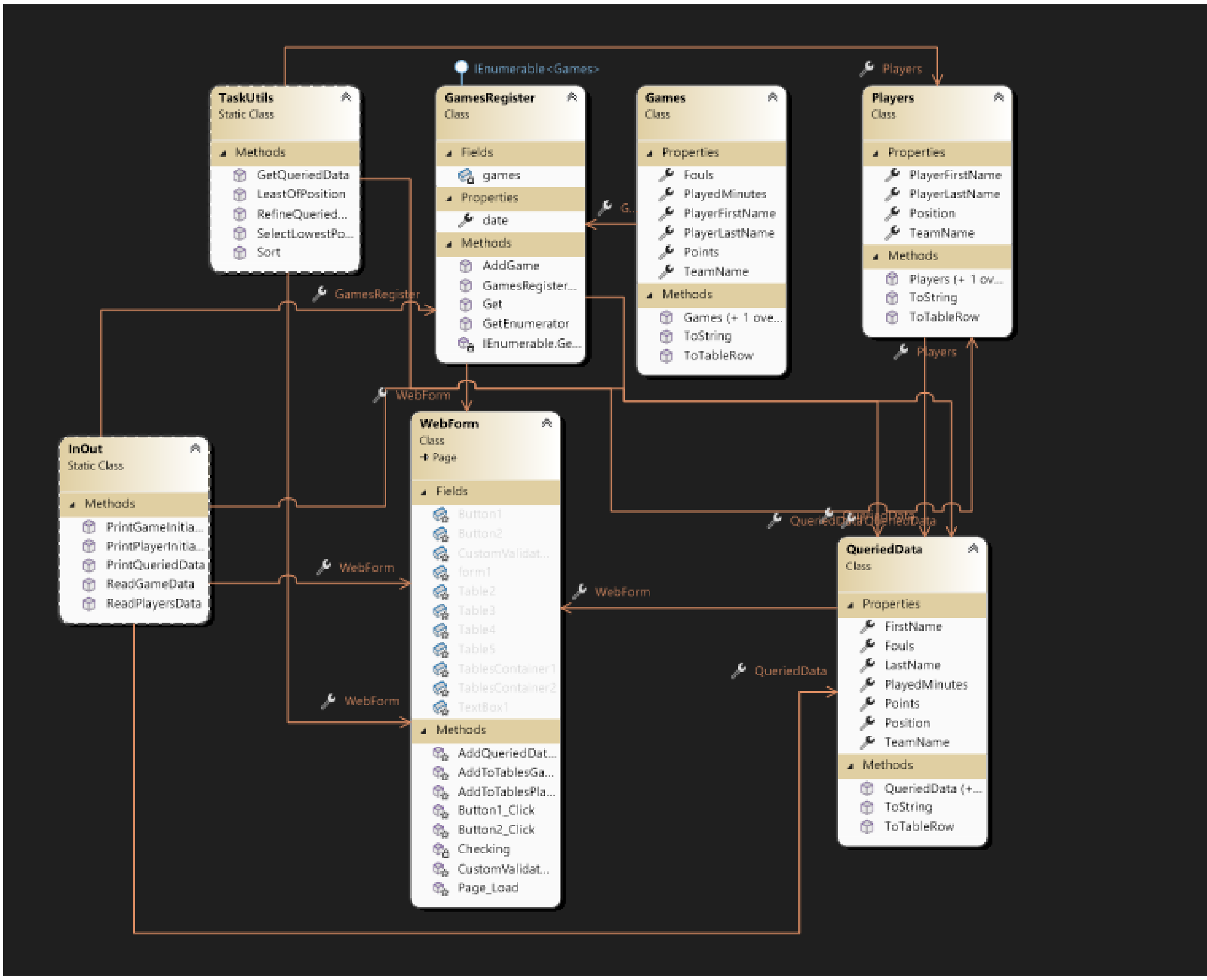
5.2. Grafinės naudotojo sąsajos schema



5.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Label(Bendri pakeitimai)	CssClass	label
Table(Bendri pakeitimai)	CssClass	table
Label1	Text	Initial Data
Label2	Text	(Tuščia, užpildoma kode vėliau)
Label3	Text	(Tuščia, užpildoma kode vėliau)
Label4	Text	(Tuščia, užpildoma kode vėliau)
Label5	Text	(Tuščia, užpildoma kode vėliau)
Label6	Text	(Tuščia, užpildoma kode vėliau)
Label7	Text	(Tuščia, užpildoma kode vėliau)
Button1	Text	Skaičiuoti

5.4. Klasių diagrama



5.5. Programos naudotojo vadovas

Įkelti žaidėjų duomenų failus į varianto faila, kuris yra „App_Data/Var1“ kataloge, surašyti duomenys į failą tokiu budu - komandos pavadinimas, krepšininko pavardė, vardas, žaistų minučių skaičius, pelnytų taškų skaičius, padarytųklaidų skaičius. Komandų duomenys įdėti į „App_Data“, surašyti - komandos pavadinimas, krepšininko pavardė, vardas, žaidimo pozicija(įžaidėjas, atakuojantis gynėjas, lengvasis puolėjas, sunkusis puolėjas, centras). Atsidaryti programą, ją paleisti, ir atsirinkti reikiama kiekį žaidėjų.

5.6. Programos tekstas

```
namespace L5.App_Code
{
      public class Games
      {
            public string TeamName { get; set; }
            public string PlayerLastName { get; set; }
            public string PlayerFirstName { get; set; }
            public int PlayedMinutes { get; set; }
            public int Points { get; set; }
            public int Fouls { get; set; }

        public Games(string teamName, string playerLastName, string playerFirstName, int
playedMinutes, int points, int fouls)
        {
            TeamName = teamName;
            PlayerLastName = playerLastName;
            PlayerFirstName = playerFirstName;
            PlayedMinutes = playedMinutes;
            Points = points;
            Fouls = fouls;
        }
        public Games()
        {
        }
        /// <summary>
        /// Converts the game data to a string for display in a console or log.
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            return $"{TeamName,-15} | {PlayerLastName,-15} | {PlayerFirstName,-15} |
{PlayedMinutes,14} | {Points,13} | {Fouls,12}|";
        }
        /// <summary>
        /// Converts the game data to a table row for display in a web form.
        /// </summary>
        /// <returns></returns>
        public TableRow ToTableRow()
        {
            TableRow row = new TableRow();
            row.Cells.Add(new TableCell() { Text = this.TeamName });
            row.Cells.Add(new TableCell() { Text = this.PlayerLastName });
            row.Cells.Add(new TableCell() { Text = this.PlayerFirstName });
            row.Cells.Add(new TableCell() { Text = this.PlayedMinutes.ToString() });
            row.Cells.Add(new TableCell() { Text = this.Points.ToString() });
            row.Cells.Add(new TableCell() { Text = this.Fouls.ToString() });
            return row;
        }

    }
}

namespace L5.App_Code
{
      public class GamesRegister : IEnumerable<Games>
      {
            public DateTime date { get; set; }

            private List<Games> games;

        public GamesRegister()
        {
            this.games = new List<Games>();
        }
        public GamesRegister(DateTime date)
        {
            this.date = date;
            this.games = new List<Games>();
        }
        /// <summary>
        /// Adds a game to the register.
        /// </summary>
        /// <param name="game"></param>
        public void AddGame(Games game)
        {
            this.games.Add(game);
        }
        /// <summary>
        /// Returns a game from register by index.
        /// </summary>
        /// <param name="index"></param>
        /// <returns></returns>
        public Games Get(int index)
        {
            return this.games[index];
        }

        public IEnumerator<Games> GetEnumerator()
        {
            foreach (Games game in this.games)
            {
                yield return game;
            }
        }

        IEnumerator IEnumerable.GetEnumerator()
        {
            return GetEnumerator();
        }
    }
}

namespace L5.App_Code
{
    public static class InOut
    {
        /// <summary>
        /// Reads game data from a specified folder.
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <returns></returns>
        public static List<GamesRegister> ReadGameData(string FileFolder)
        {
            List<GamesRegister> result = new List<GamesRegister>();

            foreach (string filePath in Directory.GetFiles(FileFolder, "*.txt"))
            {
                string[] lines = File.ReadAllLines(filePath);
                GamesRegister gamesRegister = new GamesRegister(DateTime.Parse(lines[0]));
                foreach (string line in lines.Skip(1))
                {
                    try
                    {
                        string[] parts = Regex.Split(line, "; ");
                        gamesRegister.AddGame(new Games(parts[0], parts[1], parts[2],
int.Parse(parts[3]), int.Parse(parts[4]), int.Parse(parts[5])));
                    }
                    catch (Exception ex)
                    {
                        HttpContext.Current.Response.Write($"<script>alert('Error in file
{Path.GetFileName(filePath)}: {ex.Message} - Line: {line}<br/>');</script>");
                    }
                }
                result.Add(gamesRegister);
            }
            return result;
        }

        /// <summary>
        /// Reads player data from a file.
        /// </summary>
        /// <param name="FilePath"></param>
        /// <returns></returns>
        public static List<Players> ReadPlayersData(string FilePath)
        {
            List<Players> result = new List<Players>();

            string[] lines = File.ReadAllLines(FilePath);
            foreach (string line in lines)
            {
                try
                {
                    string[] parts = Regex.Split(line, "; ");
                    Players player = new Players(parts[0], parts[1], parts[2], parts[3]);
                    result.Add(player);
                }
                catch (Exception ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in file
{Path.GetFileName(FilePath)}: {ex.Message} - Line: {line}<br/>');</script>");
                }
            }
            return result;
        }

        /// <summary>
        /// Writes game data to a file.
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="header"></param>
        /// <param name="List"></param>
        public static void PrintGameInitialData(string FileFolder, string header,
List<GamesRegister> List)
        {
            try
            {
                using (StreamWriter writer = new StreamWriter(FileFolder + @"\Output.txt",
true))
                {
                    writer.WriteLine(header);

                    int number = 1;
                    foreach (GamesRegister data in List)
                    {
                        writer.WriteLine($"Lentelė Nr.{number}");
                        writer.WriteLine(data.date.ToString("yyyy-MM-dd"));
                        writer.WriteLine(new string('-', 100));
                        writer.WriteLine($"{"Team Name",-15} | {"Last Name",-15} | {"First
Name",-15} | {"Played Minutes",-14} | {"Points Earned",-13} | {"Fouls Earned",-12}|");
                        writer.WriteLine(new string('-', 100));
                        foreach (Games question in data)
                        {
                            writer.WriteLine(question.ToString());
                            writer.WriteLine(new string('-', 100));
                        }
                        writer.WriteLine();
                        number++;
                    }
                    writer.WriteLine();
                }
            }
            catch (Exception ex)
            {
                HttpContext.Current.Response.Write($"<script>alert('Error while writing game
data: {ex.Message}<br/>');</script>");
            }
        }

        /// <summary>
        /// Writes player data to a file.
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="header"></param>
        /// <param name="List"></param>
        public static void PrintPlayerInitialData(string FileFolder, string header,
List<Players> List)
        {
            try
            {
                using (StreamWriter writer = new StreamWriter(FileFolder + @"\Output.txt",
true))
                {
                    writer.WriteLine(header);
                    writer.WriteLine(new string('-', 75));
                    writer.WriteLine($"{"Team Name",-15} | {"Last Name",-15} | {"First Name",-
15} | {"Position",-20}|");
                    writer.WriteLine(new string('-', 75));
                    foreach (Players question in List)
                    {
                        writer.WriteLine(question.ToString());
                        writer.WriteLine(new string('-', 75));
                    }
                    writer.WriteLine();
                }
            }
            catch (Exception ex)
            {
                HttpContext.Current.Response.Write($"<script>alert('Error while writing player
data: {ex.Message}<br/>');</script>");
            }
        }

        /// <summary>
        /// Writes queried data to a file.
        /// </summary>
        /// <param name="FileFolder"></param>
        /// <param name="header"></param>
        /// <param name="List"></param>
        public static void PrintQueriedData(string FileFolder, string header,
List<QueriedData> List)
        {
            try
            {
                using (StreamWriter writer = new StreamWriter(FileFolder + @"\Output.txt",
true))
                {
                    writer.WriteLine(header);
                    writer.WriteLine(new string('-', 121));
                    writer.WriteLine($"{"Team Name",-15} | {"Last Name",-15} | {"First Name",-
15} | {"Played Minutes",-14} | {"Points Earned",-13} | {"Fouls Earned",-12} | {"Position",-
20}|");
                    writer.WriteLine(new string('-', 121));
                    foreach (QueriedData question in List)
                    {
                        writer.WriteLine(question.ToString());
                        writer.WriteLine(new string('-', 121));
                    }
                    writer.WriteLine();
                }
            }
            catch (Exception ex)
            {
                HttpContext.Current.Response.Write($"<script>alert('Error while writing 
queried data: {ex.Message}<br/>');</script>");
            }
        }
    }
}

namespace L5.App_Code
{
	public class Players
	{
	 public string TeamName { get; set; }
	 public string PlayerLastName { get; set; }
	 public string PlayerFirstName { get; set; }
	 public string Position { get; set; }

	 public Players(string teamName, string playerLastName, string playerFirstName, string 
position)
        {
            TeamName = teamName;
            PlayerLastName = playerLastName;
            PlayerFirstName = playerFirstName;
            Position = position;
        }

        public Players()
        {
        }
        /// <summary>
        /// Converts the player data to a string for display in a console or log.
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            return $"{TeamName,-15} | {PlayerLastName,-15} | {PlayerFirstName,-15} | 
{Position,-20}|";
        }
        /// <summary>
        /// Converts the player data to a table row for display in a web form.
        /// </summary>
        /// <returns></returns>
        public TableRow ToTableRow()
        {
            TableRow row = new TableRow();
            row.Cells.Add(new TableCell() { Text = TeamName });
            row.Cells.Add(new TableCell() { Text = PlayerLastName });
            row.Cells.Add(new TableCell() { Text = PlayerFirstName });
            row.Cells.Add(new TableCell() { Text = Position });
            return row;
        }

    }
}

namespace L5.App_Code
{
	public class QueriedData
	{
	 public string TeamName { get; set; }
	 public string FirstName { get; set; }
	 public string LastName { get; set; }
```

```
        public int Points { get; set; }
             public int PlayedMinutes { get; set; }
        public int Fouls { get; set; }

        public string Position { get; set; }

        public QueriedData() { }

        public QueriedData(string teamName, string firstName, string lastName, int points, int
PlayedMinutes, int fouls, string position)
        {
            this.TeamName = teamName;
            this.FirstName = firstName;
            this.LastName = lastName;
            this.Points = points;
            this.PlayedMinutes = PlayedMinutes;
            this.Fouls = fouls;
            this.Position = position;
        }
        /// <summary>
        /// Converts the queried data to a string for display in a console or log.
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            return $"{TeamName,-15} | {LastName,-15} | {FirstName,-15} | {PlayedMinutes,14} |
{Points,13} | {Fouls,12} | {Position,-20}|";
        }
        /// <summary>
        /// Converts the queried data to a string for display in a console or log.
        /// </summary>
        /// <returns></returns>
        public TableRow ToTableRow()
        {
            TableRow row = new TableRow();
            row.Cells.Add(new TableCell() { Text = this.TeamName });
            row.Cells.Add(new TableCell() { Text = this.LastName });
            row.Cells.Add(new TableCell() { Text = this.FirstName });
            row.Cells.Add(new TableCell() { Text = this.PlayedMinutes.ToString() });
            row.Cells.Add(new TableCell() { Text = this.Points.ToString() });
            row.Cells.Add(new TableCell() { Text = this.Fouls.ToString() });
            row.Cells.Add(new TableCell() { Text = this.Position.ToString() });
            return row;
        }

    }
}

namespace L5.App_Code
{
    public static class TaskUtils
    {
        /// <summary>
        /// Generates queried data by combining game registers and player data.
        /// </summary>
        /// <param name="gamesRegisters"></param>
        /// <param name="players"></param>
        /// <returns></returns>
        public static List<QueriedData> GetQueriedData(List<GamesRegister> gamesRegisters,
List<Players> players)
        {
            List<QueriedData> result = new List<QueriedData>();

            try
            {
                IEnumerable<QueriedData> queriedData =
                    from gameRegister in gamesRegisters
```

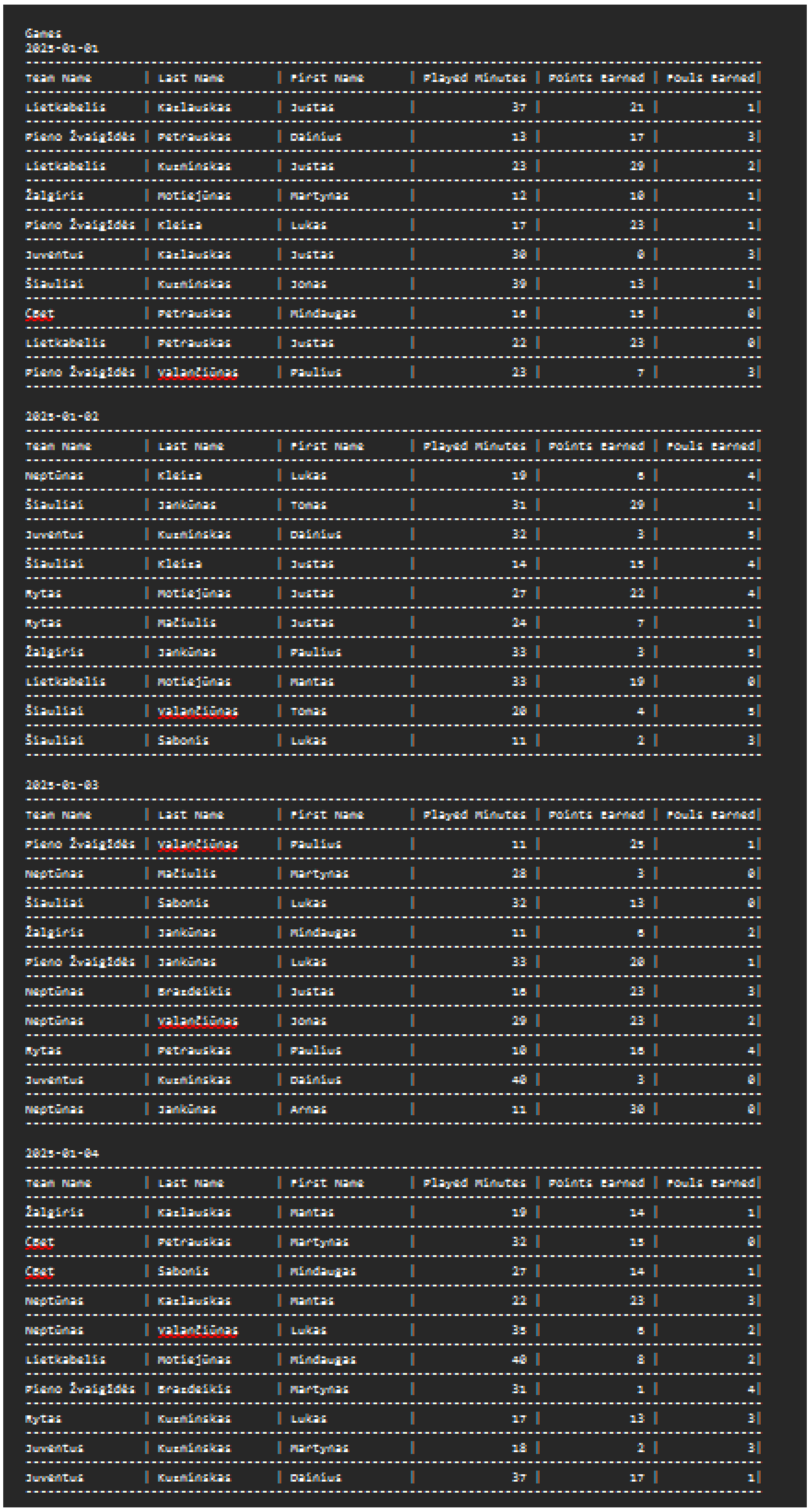
```
                        from game in gameRegister
                        from player in players
                        where game.TeamName == player.TeamName
                        select new QueriedData(player.TeamName, player.PlayerFirstName,
player.PlayerLastName, game.Points, game.PlayedMinutes, game.Fouls, player.Position);

                    result = queriedData.ToList();
                }
                catch (Exception ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in GetQueriedData:
{ex.Message}');</script>");
                }

                return result;
            }
            /// <summary>
            /// Refines the queried data by selecting the top players based on points and played
minutes.
            /// </summary>
            /// <param name="queriedData"></param>
            /// <param name="selectCount"></param>
            /// <returns></returns>
            public static List<QueriedData> RefineQueriedData(List<QueriedData> queriedData, int
selectCount)
            {
                List<QueriedData> result = new List<QueriedData>();

                try
                {
                    IEnumerable<QueriedData> refinedData = queriedData
                        .OrderByDescending(p => (double)p.Points / (p.PlayedMinutes + 1) -
p.Fouls)
                        .Take(selectCount);

                    result = refinedData.ToList();
                }
                catch (DivideByZeroException ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in
RefineQueriedData: {ex.Message}');</script>");
                }
                catch (Exception ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in
RefineQueriedData: {ex.Message}');</script>");
                }

                return result;
            }
            /// <summary>
            /// Sorts the queried data by team name and last name.
            /// </summary>
            /// <param name="queriedData"></param>
            /// <returns></returns>
            public static List<QueriedData> Sort(List<QueriedData> queriedData)
            {
                List<QueriedData> result = new List<QueriedData>();

                try
                {
                    IEnumerable<QueriedData> sortedData = queriedData
                        .OrderBy(p => p.TeamName)
                        .ThenBy(p => p.LastName);

                    result = sortedData.ToList();
                }
                catch (Exception ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in Sort: 
    {ex.Message}');</script>");
                }

                return result;
            }
            /// <summary>
            /// Selects the positiob with the least number of players.
            /// </summary>
            /// <param name="queriedData"></param>
            /// <returns></returns>
            public static string LeastOfPosition(List<QueriedData> queriedData)
            {
                string leastPlayerPosition = null;

                try
                {
                    leastPlayerPosition = queriedData
                        .GroupBy(q => q.Position)
                        .OrderBy(g => g.Count())
                        .Select(g => g.Key)
                        .FirstOrDefault();
                }
                catch (Exception ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in LeastOfPosition: 
    {ex.Message}');</script>");
                }

                return leastPlayerPosition;
            }
            /// <summary>
            /// Selects players with the lowest points in a specific position.
            /// </summary>
            /// <param name="queriedData"></param>
            /// <param name="position"></param>
            /// <returns></returns>
            public static List<QueriedData> SelectLowestPosition(List<QueriedData> queriedData, 
    string position)
            {
                List<QueriedData> result = new List<QueriedData>();

                try
                {
                    IEnumerable<QueriedData> lowestPositionPlayers = queriedData
                        .Where(q => q.Position == position)
                        .OrderBy(q => q.Points);

                    result = lowestPositionPlayers.ToList();
                }
                catch (Exception ex)
                {
                    HttpContext.Current.Response.Write($"<script>alert('Error in 
    SelectLowestPosition: {ex.Message}');</script>");
                }

                return result;
            }
        }
    }

    body {
        font-family: Arial, sans-serif;
        background-color: #f0f0f0;
    }

    .container {
        width: 80%;
        margin: 20px auto;
        padding: 20px;
        background-color: white;
        border: 1px solid #ccc;
        height: 1000px;
    }

    .label {
        font-weight: bold;
    }

    table {
        width: 90%;
        max-width: 1800px;
        table-layout: fixed;
        border-collapse: collapse;
        margin: 0 auto 10px auto;
        overflow-x: auto;
    }

        table th, table td {
            border: 1px solid #ddd;
            text-align: center;
            padding: 8px;
            word-wrap: break-word;
            white-space: normal;
        }

        table th {
            background-color: #f2f2f2;
            font-weight: bold;
        }

        table + table {
            margin-top: 10px;
        }

    .textbox {
        padding: 5px;
        border: 1px solid #ccc;
    }

    .button {
        padding: 10px 20px;
        background-color: #007bff;
        color: white;
        border: none;
        cursor: pointer;
    }

        .button:hover {
            background-color: #0056b3;
        }

    .custom-validator {
        color: red;
    }
```

```
namespace L5
{
    public partial class WebForm : System.Web.UI.Page
    {
        /// <summary>
        /// Dinamic table creation for game data.
        /// </summary>
        /// <param name="Data"></param>
        /// <param name="TablesContainer"></param>
        protected void AddToTablesGames(List<GamesRegister> Data, Panel TablesContainer)
        {
            int number = 1;
            foreach (var GameRegister in Data)
            {

                Table table = new Table
                {
                    CssClass = "table"
                };

                Label label = new Label
                {
                    Text = $"Lentele nr.{number} " + GameRegister.date.ToString("yyyy-MM-dd"),
                    CssClass = "table-header"

                };

                TableHeaderRow headerRow = new TableHeaderRow();
                headerRow.Cells.Add(new TableHeaderCell { Text = "Team Name" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Last Name" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "First Name" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Played Minutes" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Points Earned" });
                headerRow.Cells.Add(new TableHeaderCell { Text = "Fouls Earned" });

                table.Rows.Add(headerRow);

                foreach (var game in GameRegister)
                {
                    TableRow row = game.ToTableRow();
                    table.Rows.Add(row);
                }
                TablesContainer.Controls.Add(label);
                TablesContainer.Controls.Add(table);
                number++;
            }
        }
        /// <summary>
        /// Adds queried data to the specified table for players.
        /// </summary>
        /// <param name="Data"></param>
        /// <param name="TablesContainer"></param>
        protected void AddToTablesPlayers(List<Players> Data, Panel TablesContainer)
        {
            Table table = new Table
            {
                CssClass = "table"
            };
            Label label = new Label
            {
                Text = "Player Data",
                CssClass = "table-header"
            };
            TableHeaderRow headerRow = new TableHeaderRow();
            headerRow.Cells.Add(new TableHeaderCell { Text = "Team Name" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Last Name" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "First Name" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Position" });
            table.Rows.Add(headerRow);
            foreach (var player in Data)
            {
                TableRow row = player.ToTableRow();
                table.Rows.Add(row);
            }
            TablesContainer.Controls.Add(label);
            TablesContainer.Controls.Add(table);
        }
        /// <summary>
        /// Adds queried data to the specified table.
        /// </summary>
        /// <param name="Data"></param>
        /// <param name="Table"></param>
        /// <param name="Header"></param>
        protected void AddQueriedDataToTable(List<QueriedData> Data, Table Table, string
Header)
        {

            TableHeaderRow headerRow = new TableHeaderRow();
            headerRow.Cells.Add(new TableHeaderCell { Text = "Team Name" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Last Name" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "First Name" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Played Minutes" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Points Earned" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Fouls Earned" });
            headerRow.Cells.Add(new TableHeaderCell { Text = "Position" });
            Table.Rows.Add(headerRow);
            foreach (var player in Data)
            {
                TableRow row = player.ToTableRow();
                Table.Rows.Add(row);
            }
        }
        /// <summary>
        /// Custom validator for the TextBox1 control.
        /// </summary>
        /// <param name="source"></param>
        /// <param name="args"></param>
        protected void CustomValidator1_ServerValidate(object source, ServerValidateEventArgs
args)
        {
            int N;
            args.IsValid = int.TryParse(TextBox1.Text, out N) && N >= 1;
        }

    }
}

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm.aspx.cs"
Inherits="L5.WebForm" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link rel="stylesheet" type="text/css" href="StyleSheet.css" />
</head>
<body style="height: 600px">
    <form id="form1" runat="server">
        <div style="height: 605px; margin-left: 80px;">
            <br />
            <br />
            <asp:Panel ID="TablesContainer1" runat="server"></asp:Panel>
            <br />
            <br />
            <asp:Panel ID="TablesContainer2" runat="server"></asp:Panel>
            <br />
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="Skaičiuoti"
/>
            <br />
            <br />
            <br />
            <br />
            <asp:TextBox ID="TextBox1" runat="server" Visible="False"></asp:TextBox>
            <asp:CustomValidator ID="CustomValidator1" runat="server"
ControlToValidate="TextBox1" ErrorMessage="Netinkamas skaičius" ValidateEmptyText="True"
OnServerValidate="CustomValidator1_ServerValidate"></asp:CustomValidator>
            <br />
            <asp:Button ID="Button2" runat="server" OnClick="Button2_Click" Text="Atrinkti"
Visible="False" />
            <br />
            <br />

            <asp:Table ID="Table2" runat="server"></asp:Table>
            <br />
            <br />
            <asp:Table ID="Table3" runat="server"></asp:Table>
            <br />
            <br />
            <asp:Table ID="Table4" runat="server"></asp:Table>
            <br />
            <br />
            <asp:Table ID="Table5" runat="server"></asp:Table>
            <br />
        </div>
    </form>
</body>
</html>
```

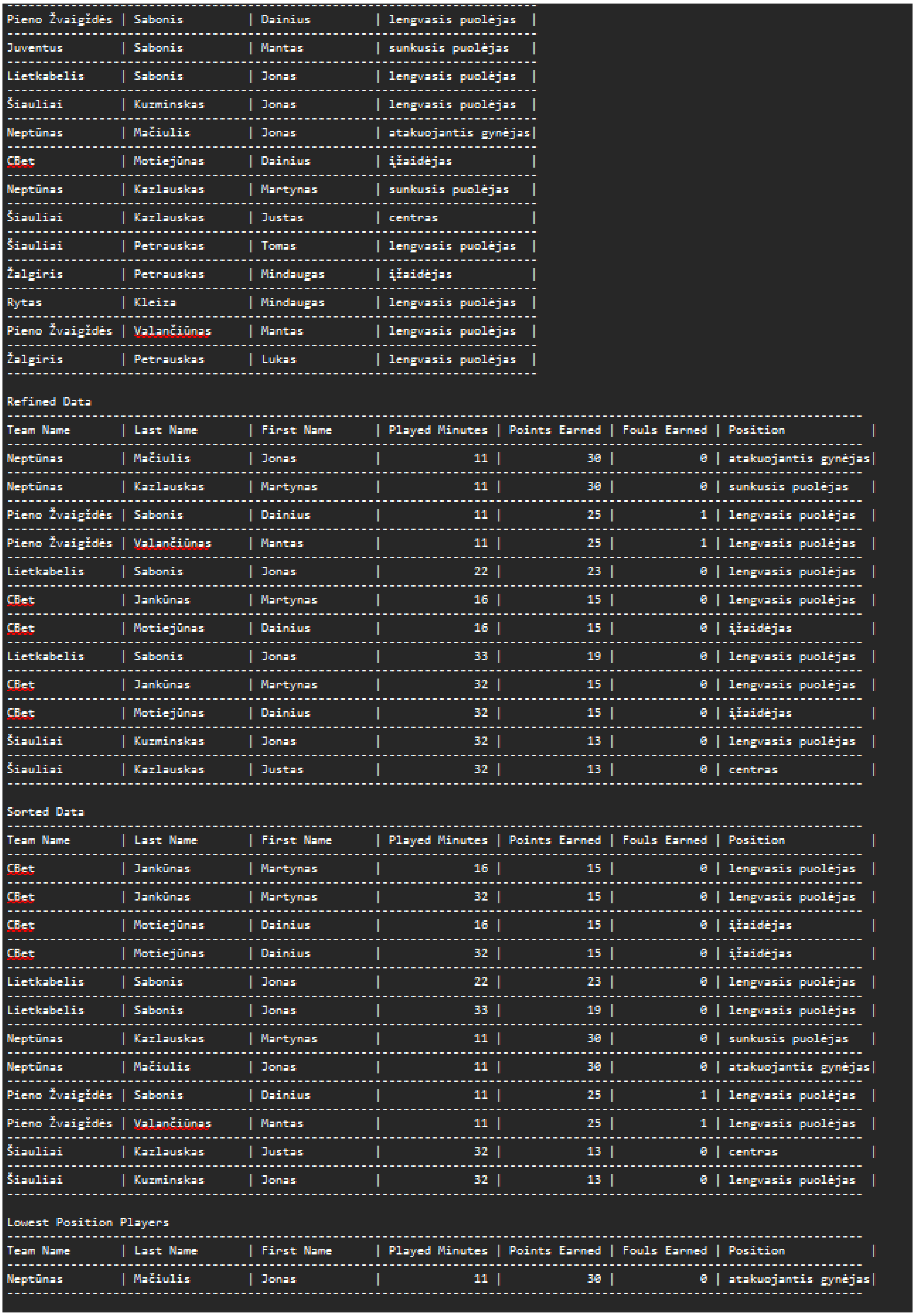
```
namespace L5
{
      public partial class WebForm : System.Web.UI.Page
      {
        protected void Page_Load(object sender, EventArgs e)
        {

            if (Session["Games"] != null && Session["Players"] != null)
            {
                string errors = string.Empty;
                var games = (List<GamesRegister>)Session["Games"];
                var players = (List<Players>)Session["Players"];
                AddToTablesGames(games, TablesContainer1);
                AddToTablesPlayers(players, TablesContainer2);

                var queriedData = TaskUtils.GetQueriedData(games, players);
                //AddQueriedDataToTable(queriedData, Table2, "Initial Queried Data");

                Button2.Visible = true;
                TextBox1.Visible = true;
            }
        }


        protected void Button1_Click(object sender, EventArgs e)
        {
            string CIN = Server.MapPath(@"App_Data/Var1");
            string CIN2 = Server.MapPath(@"App_Data/Zaidejai.txt");
            string COUT = Server.MapPath(@"App_Data");

            File.Delete(COUT + @"\Output.txt");

            string errors = string.Empty;

            List<GamesRegister> Games = InOut.ReadGameData(CIN);
            List<Players> Players = InOut.ReadPlayersData(CIN2);

            Session["Games"] = Games;
            Session["Players"] = Players;

            AddToTablesGames(Games, TablesContainer1);
            AddToTablesPlayers(Players, TablesContainer2);
            InOut.PrintGameInitialData(COUT, "Games", Games);
            InOut.PrintPlayerInitialData(COUT, "Players", Players);

            List<QueriedData> queriedData = TaskUtils.GetQueriedData(Games, Players);
            //InOut.PrintQueriedData(COUT, "Initial Queried Data", queriedData);
            Button2.Visible = true;
            TextBox1.Visible = true;


        }


        protected void Button2_Click(object sender, EventArgs e)
        {
            Page.Validate();
            if (Page.IsValid)
            {
                string CIN = Server.MapPath(@"App_Data/Var1");
                string CIN2 = Server.MapPath(@"App_Data/Zaidejai.txt");
                string COUT = Server.MapPath(@"App_Data");

                string errors = string.Empty;

                List<GamesRegister> Games = InOut.ReadGameData(CIN);
                List<Players> Players = InOut.ReadPlayersData(CIN2);

                List<QueriedData> queriedData = TaskUtils.GetQueriedData(Games, Players);
                int HowManyToTake = int.Parse(TextBox1.Text);
                List<QueriedData> refinedData = TaskUtils.RefineQueriedData(queriedData,
HowManyToTake);
                List<QueriedData> sortedData = TaskUtils.Sort(refinedData);
                string leastPlayerPosition = TaskUtils.LeastOfPosition(refinedData);
                List<QueriedData> lowestPositionPlayers =
TaskUtils.SelectLowestPosition(refinedData, leastPlayerPosition);

                AddQueriedDataToTable(refinedData, Table2, "Refined Data");
                InOut.PrintQueriedData(COUT, "Refined Data", refinedData);

                AddQueriedDataToTable(sortedData, Table3, "Sorted Data");
                InOut.PrintQueriedData(COUT, "Sorted Data", sortedData);

                AddQueriedDataToTable(lowestPositionPlayers, Table4, "Lowest Position
Players");
                InOut.PrintQueriedData(COUT, "Lowest Position Players",
lowestPositionPlayers);


            }
        }


    }
}
```

5.7. Pradiniai duomenys ir rezultatai

I VAR

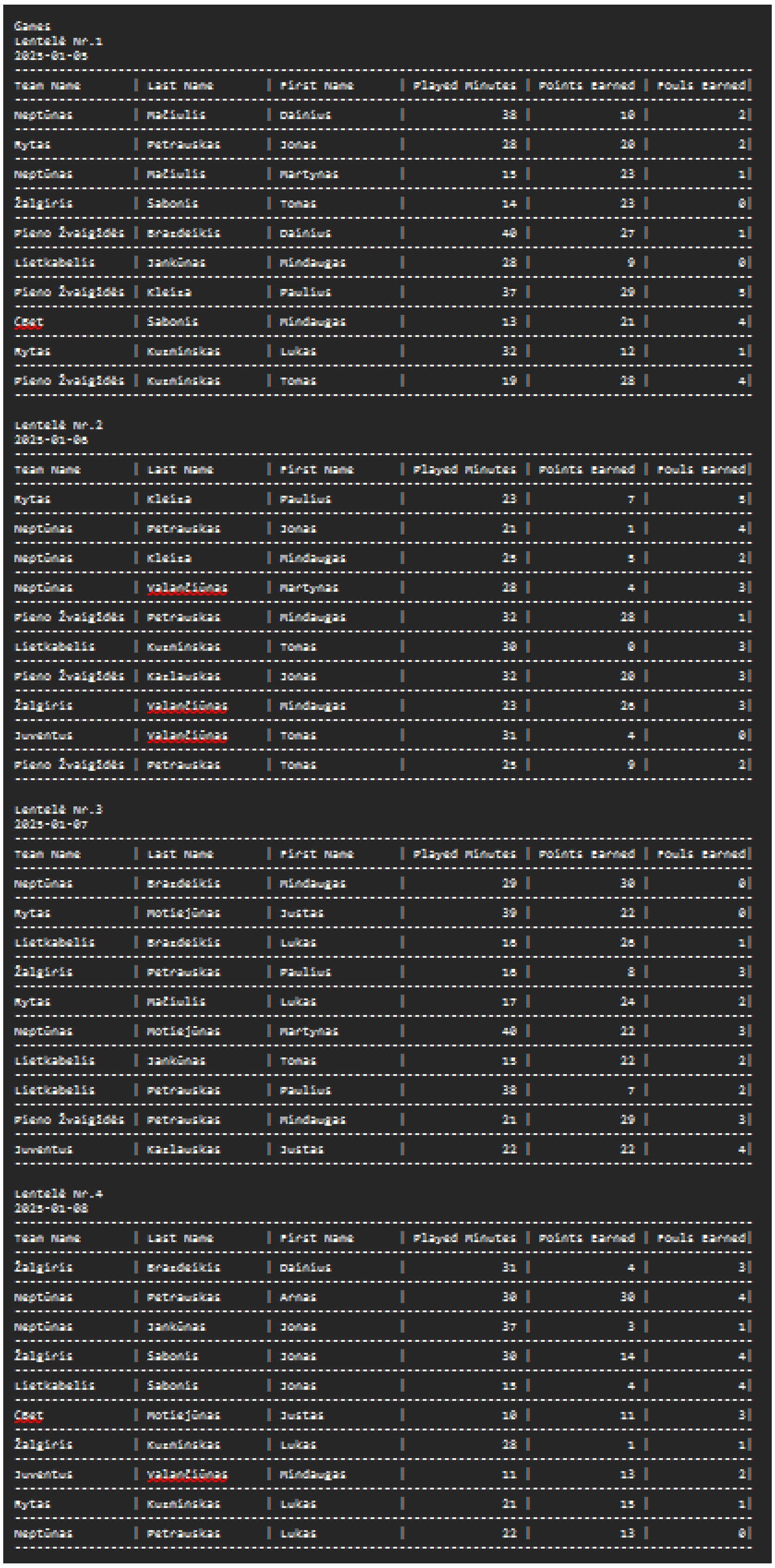


```
Games
2025-01-01
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Kazlauskas      | Justas          |             37 |            21 |            1|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Dainius         |             13 |            17 |            3|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Kuzminskas      | Justas          |             23 |            29 |            2|
----------------------------------------------------------------------------------------------------
Žalgiris        | Motiejūnas      | Martynas        |             12 |            10 |            1|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kleiza          | Lukas           |             17 |            23 |            1|
----------------------------------------------------------------------------------------------------
Juventus        | Kazlauskas      | Justas          |             30 |             0 |            3|
----------------------------------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           |             39 |            13 |            1|
----------------------------------------------------------------------------------------------------
Cbet            | Petrauskas      | Mindaugas       |             16 |            15 |            0|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Petrauskas      | Justas          |             22 |            23 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Paulius         |             23 |             7 |            3|
----------------------------------------------------------------------------------------------------

2025-01-02
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Neptūnas        | Kleiza          | Lukas           |             19 |             6 |            4|
----------------------------------------------------------------------------------------------------
Šiauliai        | Jankūnas        | Tomas           |             31 |            29 |            1|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Dainius         |             32 |             3 |            5|
----------------------------------------------------------------------------------------------------
Šiauliai        | Kleiza          | Justas          |             14 |            15 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Motiejūnas      | Justas          |             27 |            22 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Mačiulis        | Justas          |             24 |             7 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Jankūnas        | Paulius         |             33 |             3 |            5|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Motiejūnas      | Mantas          |             33 |            19 |            0|
----------------------------------------------------------------------------------------------------
Šiauliai        | Valančiūnas     | Tomas           |             20 |             4 |            5|
----------------------------------------------------------------------------------------------------
Šiauliai        | Sabonis         | Lukas           |             11 |             2 |            3|
----------------------------------------------------------------------------------------------------

2025-01-03
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Paulius         |             11 |            25 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Martynas        |             28 |             3 |            0|
----------------------------------------------------------------------------------------------------
Šiauliai        | Sabonis         | Lukas           |             32 |            13 |            0|
----------------------------------------------------------------------------------------------------
Žalgiris        | Jankūnas        | Mindaugas       |             11 |             6 |            2|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Jankūnas        | Lukas           |             33 |            20 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Brazdeikis      | Justas          |             16 |            23 |            3|
----------------------------------------------------------------------------------------------------
Neptūnas        | Valančiūnas     | Jonas           |             29 |            23 |            2|
----------------------------------------------------------------------------------------------------
Rytas           | Petrauskas      | Paulius         |             10 |            16 |            4|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Dainius         |             40 |             3 |            0|
----------------------------------------------------------------------------------------------------
Neptūnas        | Jankūnas        | Arnas           |             11 |            30 |            0|
----------------------------------------------------------------------------------------------------

2025-01-04
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Žalgiris        | Kazlauskas      | Mantas          |             19 |            14 |            1|
----------------------------------------------------------------------------------------------------
Cbet            | Petrauskas      | Martynas        |             32 |            15 |            0|
----------------------------------------------------------------------------------------------------
Cbet            | Sabonis         | Mindaugas       |             27 |            14 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Mantas          |             22 |            23 |            3|
----------------------------------------------------------------------------------------------------
Neptūnas        | Valančiūnas     | Lukas           |             35 |             6 |            2|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Motiejūnas      | Mindaugas       |             40 |             8 |            2|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Brazdeikis      | Martynas        |             31 |             1 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Kuzminskas      | Lukas           |             17 |            13 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Martynas        |             18 |             2 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Dainius         |             37 |            17 |            1|
----------------------------------------------------------------------------------------------------
```

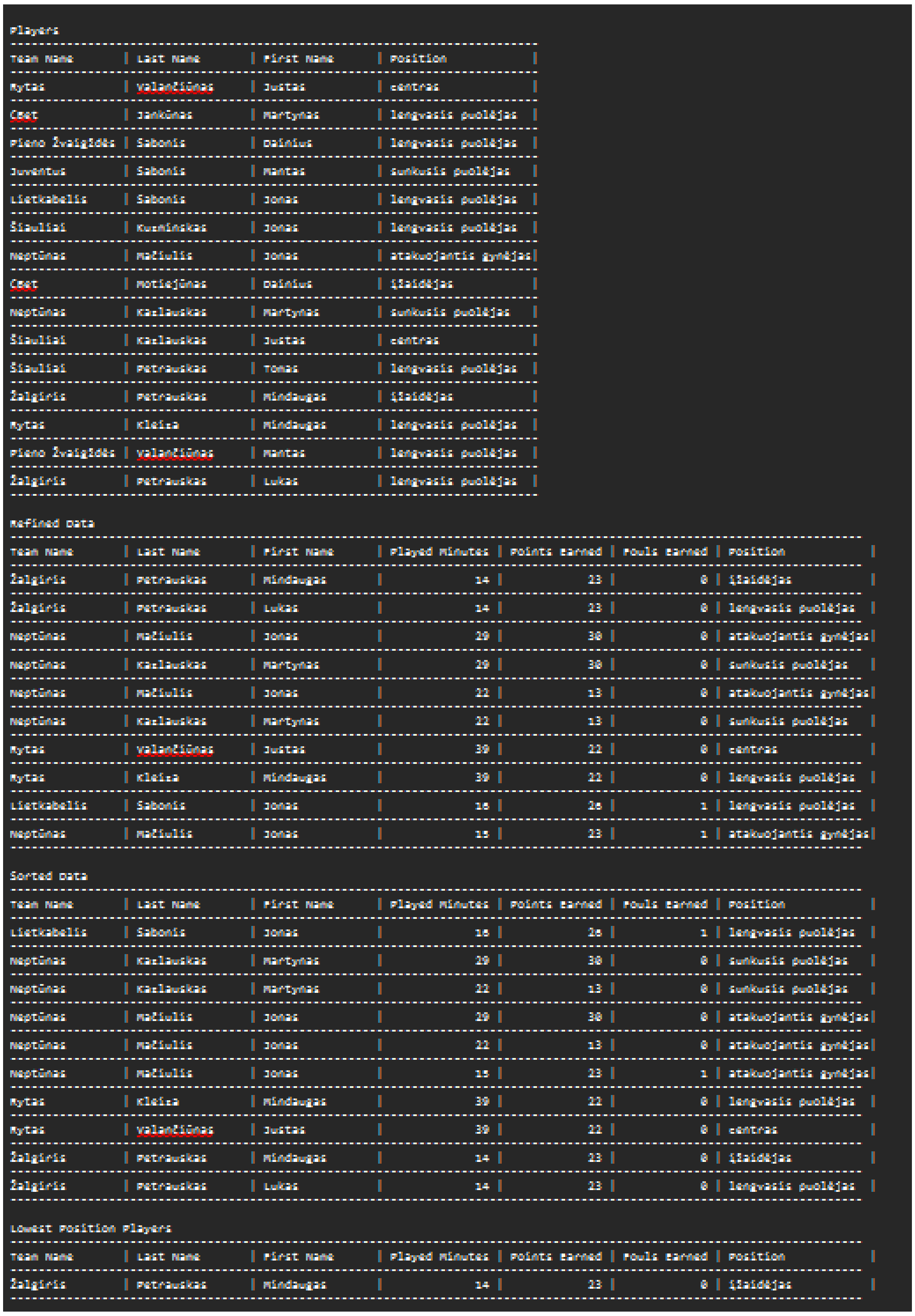


```
-----------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         | lengvasis puolėjas  |
-----------------------------------------------------------------------------
Juventus        | Sabonis         | Mantas          | sunkusis puolėjas   |
-----------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           | lengvasis puolėjas  |
-----------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           | lengvasis puolėjas  |
-----------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           | atakuojantis gynėjas|
-----------------------------------------------------------------------------
CBet            | Motiejūnas      | Dainius         | įžaidėjas           |
-----------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        | sunkusis puolėjas   |
-----------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          | centras             |
-----------------------------------------------------------------------------
Šiauliai        | Petrauskas      | Tomas           | lengvasis puolėjas  |
-----------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       | įžaidėjas           |
-----------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       | lengvasis puolėjas  |
-----------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          | lengvasis puolėjas  |
-----------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           | lengvasis puolėjas  |
-----------------------------------------------------------------------------

Refined Data
-------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             11 |            30 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             11 |            30 |            0 | sunkusis puolėjas   |
-------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         |             11 |            25 |            1 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          |             11 |            25 |            1 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             22 |            23 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Jankūnas        | Martynas        |             16 |            15 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Motiejūnas      | Dainius         |             16 |            15 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             33 |            19 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Jankūnas        | Martynas        |             32 |            15 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Motiejūnas      | Dainius         |             32 |            15 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           |             32 |            13 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          |             32 |            13 |            0 | centras             |
-------------------------------------------------------------------------------------------------------------------------

Sorted Data
-------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Jankūnas        | Martynas        |             16 |            15 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Jankūnas        | Martynas        |             32 |            15 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Motiejūnas      | Dainius         |             16 |            15 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
CBet            | Motiejūnas      | Dainius         |             32 |            15 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             22 |            23 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             33 |            19 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             11 |            30 |            0 | sunkusis puolėjas   |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             11 |            30 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         |             11 |            25 |            1 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          |             11 |            25 |            1 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          |             32 |            13 |            0 | centras             |
-------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           |             32 |            13 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------

Lowest Position Players
-------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             11 |            30 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
```

II VAR

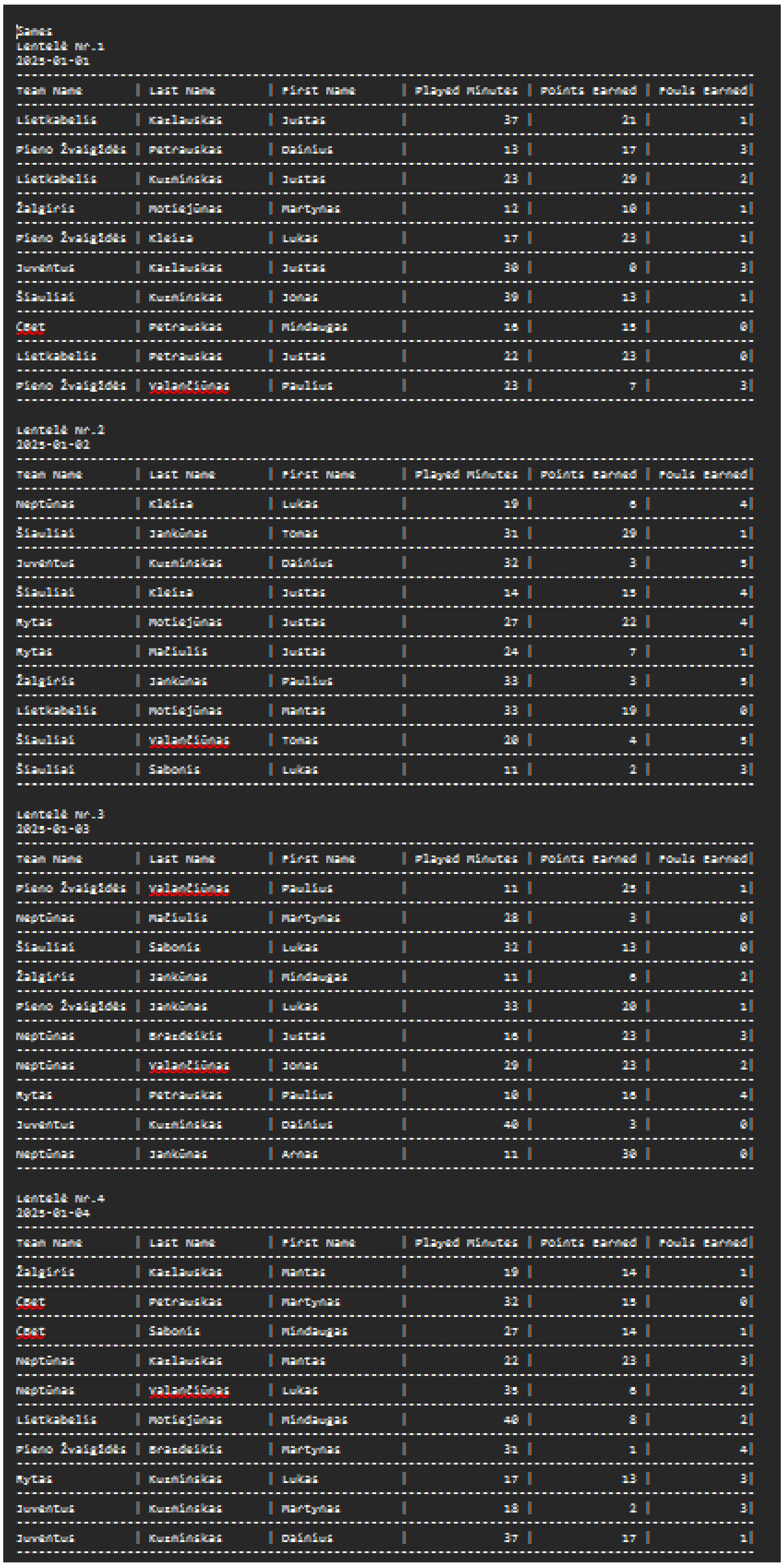


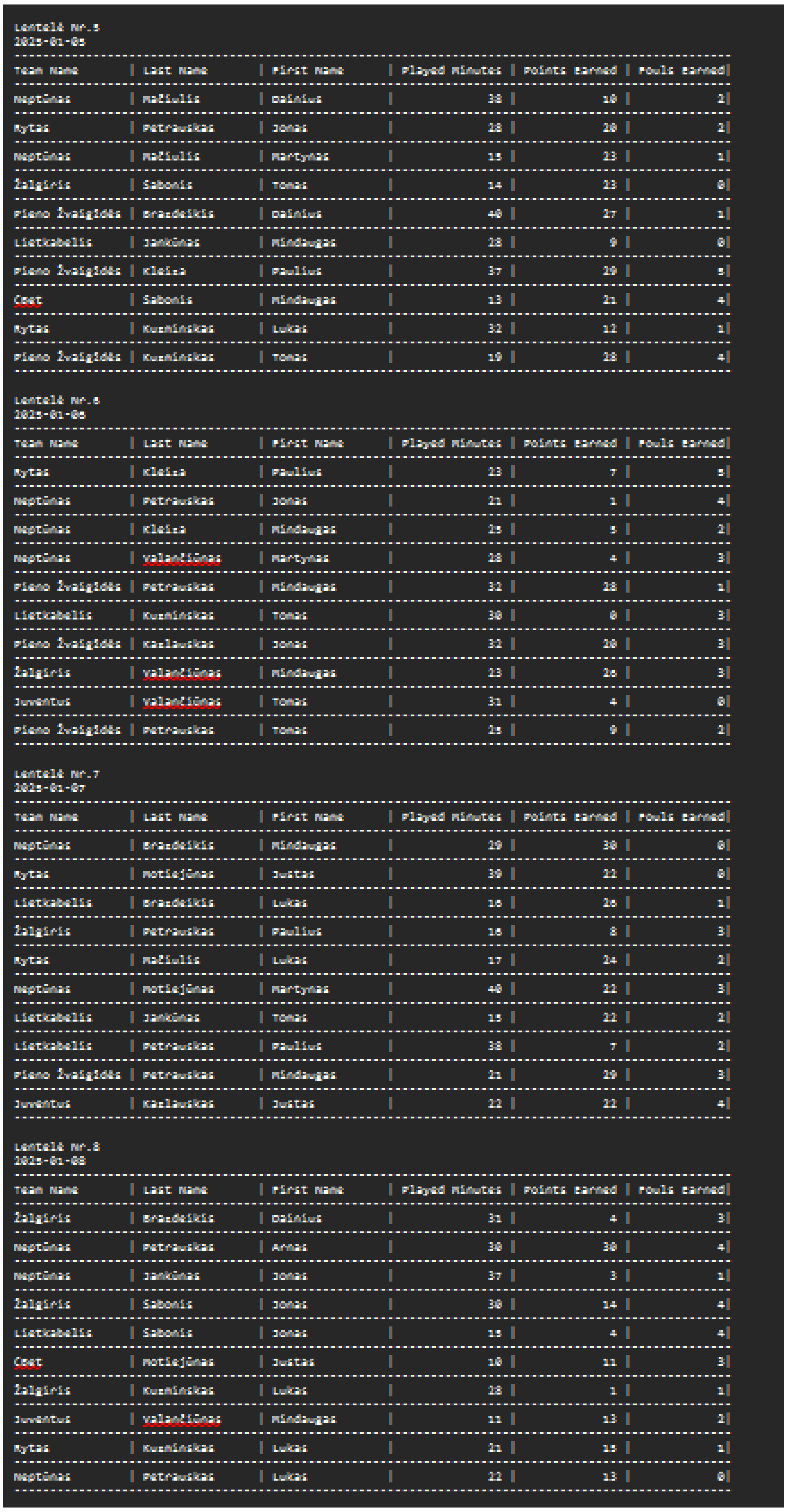
```
Games
Lentelė Nr.1
2025-01-05
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Dainius         |             38 |            10 |            2|
----------------------------------------------------------------------------------------------------
Rytas           | Petrauskas      | Jonas           |             28 |            20 |            2|
----------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Martynas        |             15 |            23 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Sabonis         | Tomas           |             14 |            23 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Brazdeikis      | Dainius         |             40 |            27 |            1|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Jankūnas        | Mindaugas       |             28 |             9 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kleiza          | Paulius         |             37 |            29 |            5|
----------------------------------------------------------------------------------------------------
Cset            | Sabonis         | Mindaugas       |             13 |            21 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Kuzminskas      | Lukas           |             32 |            12 |            1|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kuzminskas      | Tomas           |             19 |            28 |            4|
----------------------------------------------------------------------------------------------------

Lentelė Nr.2
2025-01-05
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Rytas           | Kleiza          | Paulius         |             23 |             7 |            5|
----------------------------------------------------------------------------------------------------
Neptūnas        | Petrauskas      | Jonas           |             21 |             1 |            4|
----------------------------------------------------------------------------------------------------
Neptūnas        | Kleiza          | Mindaugas       |             25 |             5 |            2|
----------------------------------------------------------------------------------------------------
Neptūnas        | Valančiūnas     | Martynas        |             28 |             4 |            3|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Mindaugas       |             32 |            28 |            1|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Kuzminskas      | Tomas           |             30 |             0 |            3|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kazlauskas      | Jonas           |             32 |            20 |            3|
----------------------------------------------------------------------------------------------------
Žalgiris        | Valančiūnas     | Mindaugas       |             23 |            26 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Valančiūnas     | Tomas           |             31 |             4 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Tomas           |             25 |             9 |            2|
----------------------------------------------------------------------------------------------------

Lentelė Nr.3
2025-01-07
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Neptūnas        | Brazdeikis      | Mindaugas       |             29 |            30 |            0|
----------------------------------------------------------------------------------------------------
Rytas           | Motiejūnas      | Justas          |             39 |            22 |            0|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Brazdeikis      | Lukas           |             16 |            26 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Paulius         |             16 |             8 |            3|
----------------------------------------------------------------------------------------------------
Rytas           | Mačiulis        | Lukas           |             17 |            24 |            2|
----------------------------------------------------------------------------------------------------
Neptūnas        | Motiejūnas      | Martynas        |             40 |            22 |            3|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Jankūnas        | Tomas           |             15 |            22 |            2|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Petrauskas      | Paulius         |             38 |             7 |            2|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Mindaugas       |             21 |            29 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Kazlauskas      | Justas          |             22 |            22 |            4|
----------------------------------------------------------------------------------------------------

Lentelė Nr.4
2025-01-08
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Žalgiris        | Brazdeikis      | Dainius         |             31 |             4 |            3|
----------------------------------------------------------------------------------------------------
Neptūnas        | Petrauskas      | Arnas           |             30 |            30 |            4|
----------------------------------------------------------------------------------------------------
Neptūnas        | Jankūnas        | Jonas           |             37 |             3 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Sabonis         | Jonas           |             30 |            14 |            4|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             15 |             4 |            4|
----------------------------------------------------------------------------------------------------
Cset            | Motiejūnas      | Justas          |             10 |            11 |            3|
----------------------------------------------------------------------------------------------------
Žalgiris        | Kuzminskas      | Lukas           |             28 |             1 |            1|
----------------------------------------------------------------------------------------------------
Juventus        | Valančiūnas     | Mindaugas       |             11 |            13 |            2|
----------------------------------------------------------------------------------------------------
Rytas           | Kuzminskas      | Lukas           |             21 |            15 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Petrauskas      | Lukas           |             22 |            13 |            0|
----------------------------------------------------------------------------------------------------
```



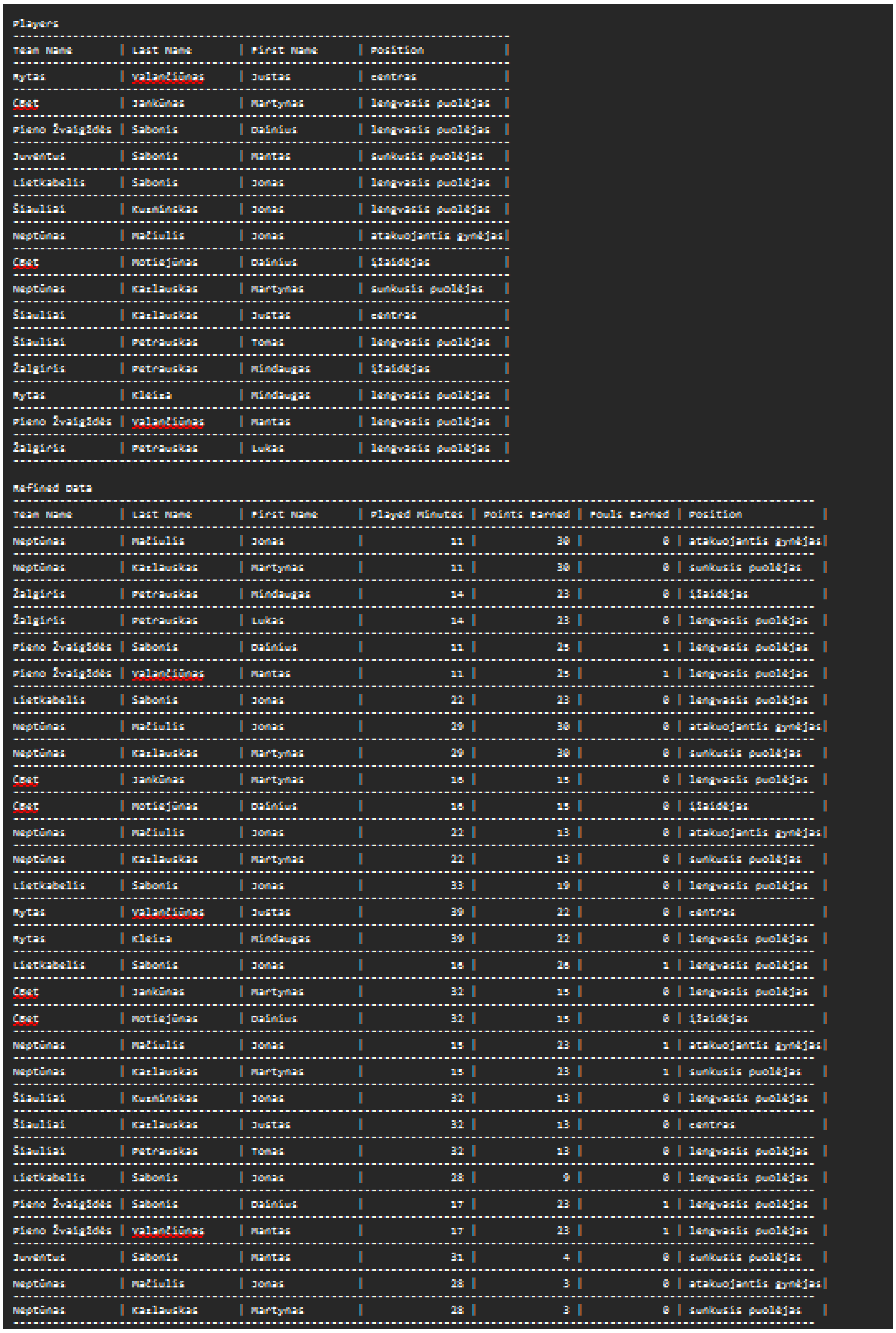
```
Players
---------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Position            |
---------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          | centras             |
---------------------------------------------------------------------------
CBet            | Jankūnas        | Martynas        | lengvasis puolėjas  |
---------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         | lengvasis puolėjas  |
---------------------------------------------------------------------------
Juventus        | Sabonis         | Mantas          | sunkusis puolėjas   |
---------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           | lengvasis puolėjas  |
---------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           | lengvasis puolėjas  |
---------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           | atakuojantis gynėjas|
---------------------------------------------------------------------------
CBet            | Motiejūnas      | Dainius         | įžaidėjas           |
---------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        | sunkusis puolėjas   |
---------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          | centras             |
---------------------------------------------------------------------------
Šiauliai        | Petrauskas      | Tomas           | lengvasis puolėjas  |
---------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       | įžaidėjas           |
---------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       | lengvasis puolėjas  |
---------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          | lengvasis puolėjas  |
---------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           | lengvasis puolėjas  |
---------------------------------------------------------------------------

Refined Data
-------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
-------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       |             14 |            23 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           |             14 |            23 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             29 |            30 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             29 |            30 |            0 | sunkusis puolėjas   |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             22 |            13 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             22 |            13 |            0 | sunkusis puolėjas   |
-------------------------------------------------------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          |             39 |            22 |            0 | centras             |
-------------------------------------------------------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       |             39 |            22 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             18 |            28 |            1 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             15 |            23 |            1 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------

Sorted Data
-------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
-------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             18 |            28 |            1 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             29 |            30 |            0 | sunkusis puolėjas   |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             22 |            13 |            0 | sunkusis puolėjas   |
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             29 |            30 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             22 |            13 |            0 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             15 |            23 |            1 | atakuojantis gynėjas|
-------------------------------------------------------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       |             39 |            22 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          |             39 |            22 |            0 | centras             |
-------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       |             14 |            23 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           |             14 |            23 |            0 | lengvasis puolėjas  |
-------------------------------------------------------------------------------------------------------------------------

Lowest Position Players
-------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
-------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       |             14 |            23 |            0 | įžaidėjas           |
-------------------------------------------------------------------------------------------------------------------------
```

III Var

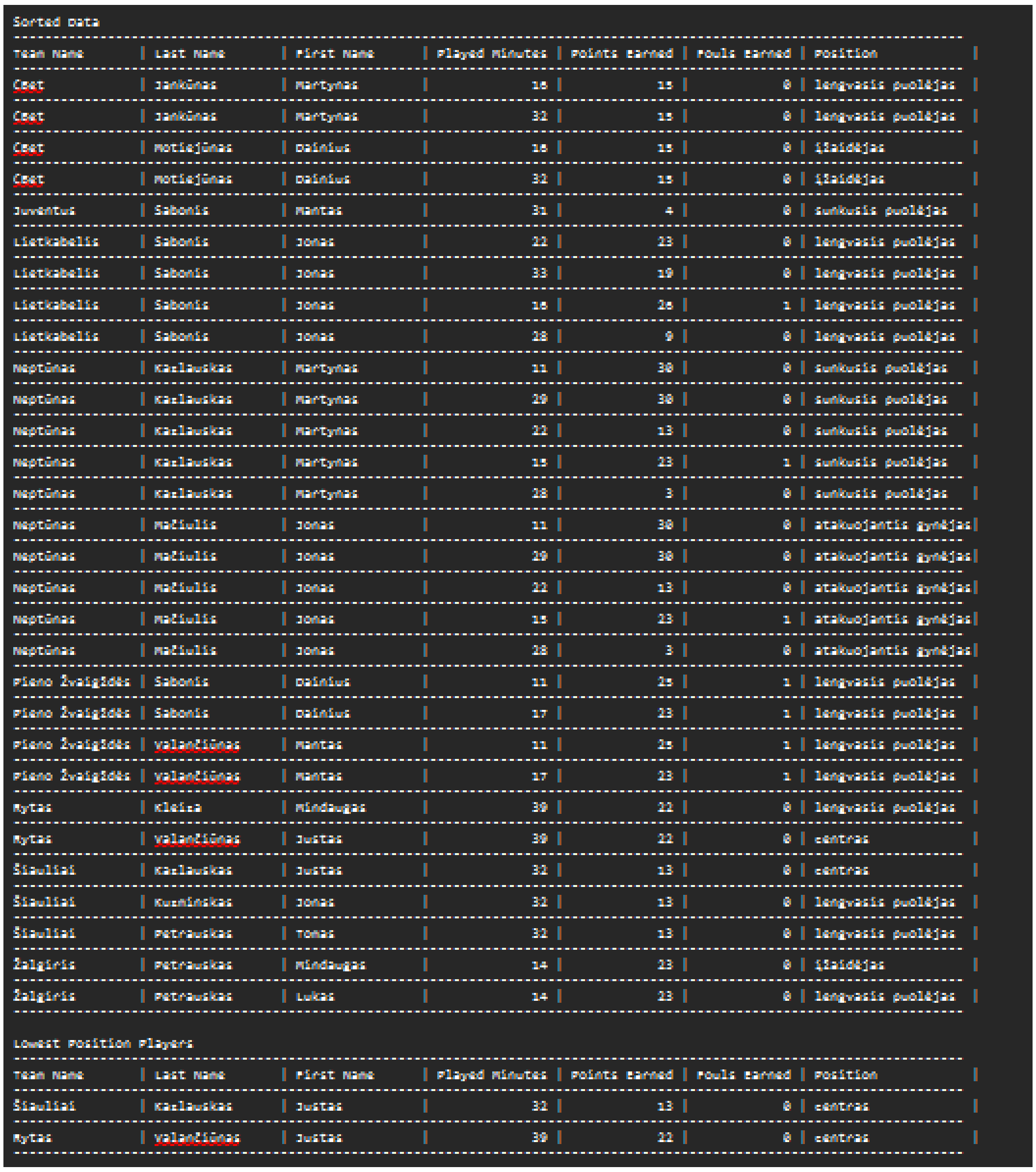

```
Games
Lentelė Nr.1
2025-01-01
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Kazlauskas      | Justas          |             37 |            21 |            1|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Dainius         |             13 |            17 |            3|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Kuzminskas      | Justas          |             23 |            29 |            2|
----------------------------------------------------------------------------------------------------
Žalgiris        | Motiejūnas      | Martynas        |             12 |            10 |            1|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kleiza          | Lukas           |             17 |            23 |            1|
----------------------------------------------------------------------------------------------------
Juventus        | Kazlauskas      | Justas          |             30 |             0 |            3|
----------------------------------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           |             39 |            13 |            1|
----------------------------------------------------------------------------------------------------
Cbet            | Petrauskas      | Mindaugas       |             16 |            15 |            0|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Petrauskas      | Justas          |             22 |            23 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Paulius         |             23 |             7 |            3|
----------------------------------------------------------------------------------------------------

Lentelė Nr.2
2025-01-02
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Neptūnas        | Kleiza          | Lukas           |             19 |             6 |            4|
----------------------------------------------------------------------------------------------------
Šiauliai        | Jankūnas        | Tomas           |             31 |            29 |            1|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Dainius         |             32 |             3 |            5|
----------------------------------------------------------------------------------------------------
Šiauliai        | Kleiza          | Justas          |             14 |            15 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Motiejūnas      | Justas          |             27 |            22 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Mačiulis        | Justas          |             24 |             7 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Jankūnas        | Paulius         |             33 |             3 |            5|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Motiejūnas      | Mantas          |             33 |            19 |            0|
----------------------------------------------------------------------------------------------------
Šiauliai        | Valančiūnas     | Tomas           |             20 |             4 |            5|
----------------------------------------------------------------------------------------------------
Šiauliai        | Sabonis         | Lukas           |             11 |             2 |            3|
----------------------------------------------------------------------------------------------------

Lentelė Nr.3
2025-01-03
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Paulius         |             11 |            25 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Martynas        |             28 |             3 |            0|
----------------------------------------------------------------------------------------------------
Šiauliai        | Sabonis         | Lukas           |             32 |            13 |            0|
----------------------------------------------------------------------------------------------------
Žalgiris        | Jankūnas        | Mindaugas       |             11 |             6 |            2|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Jankūnas        | Lukas           |             33 |            20 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Brazdeikis      | Justas          |             16 |            23 |            3|
----------------------------------------------------------------------------------------------------
Neptūnas        | Valančiūnas     | Jonas           |             29 |            23 |            2|
----------------------------------------------------------------------------------------------------
Rytas           | Petrauskas      | Paulius         |             10 |            16 |            4|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Dainius         |             40 |             3 |            0|
----------------------------------------------------------------------------------------------------
Neptūnas        | Jankūnas        | Arnas           |             11 |            30 |            0|
----------------------------------------------------------------------------------------------------

Lentelė Nr.4
2025-01-04
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Žalgiris        | Kazlauskas      | Mantas          |             19 |            14 |            1|
----------------------------------------------------------------------------------------------------
Cbet            | Petrauskas      | Martynas        |             32 |            15 |            0|
----------------------------------------------------------------------------------------------------
Cbet            | Sabonis         | Mindaugas       |             27 |            14 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Mantas          |             22 |            23 |            3|
----------------------------------------------------------------------------------------------------
Neptūnas        | Valančiūnas     | Lukas           |             35 |             6 |            2|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Motiejūnas      | Mindaugas       |             40 |             8 |            2|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Brazdeikis      | Martynas        |             31 |             1 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Kuzminskas      | Lukas           |             17 |            13 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Martynas        |             18 |             2 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Kuzminskas      | Dainius         |             37 |            17 |            1|
----------------------------------------------------------------------------------------------------
```



```
Lentelė Nr.5
2025-01-05
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Dainius         |             38 |            10 |            2|
----------------------------------------------------------------------------------------------------
Rytas           | Petrauskas      | Jonas           |             28 |            20 |            2|
----------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Martynas        |             15 |            23 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Sabonis         | Tomas           |             14 |            23 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Brazdeikis      | Dainius         |             40 |            27 |            1|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Jankūnas        | Mindaugas       |             28 |             9 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kleiza          | Paulius         |             37 |            29 |            5|
----------------------------------------------------------------------------------------------------
Cbet            | Sabonis         | Mindaugas       |             13 |            21 |            4|
----------------------------------------------------------------------------------------------------
Rytas           | Kuzminskas      | Lukas           |             32 |            12 |            1|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kuzminskas      | Tomas           |             19 |            28 |            4|
----------------------------------------------------------------------------------------------------

Lentelė Nr.6
2025-01-06
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Rytas           | Kleiza          | Paulius         |             23 |             7 |            5|
----------------------------------------------------------------------------------------------------
Neptūnas        | Petrauskas      | Jonas           |             21 |             1 |            4|
----------------------------------------------------------------------------------------------------
Neptūnas        | Kleiza          | Mindaugas       |             25 |             5 |            2|
----------------------------------------------------------------------------------------------------
Neptūnas        | Valančiūnas     | Martynas        |             28 |             4 |            3|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Mindaugas       |             32 |            28 |            1|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Kuzminskas      | Tomas           |             30 |             0 |            3|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Kazlauskas      | Jonas           |             32 |            20 |            3|
----------------------------------------------------------------------------------------------------
Žalgiris        | Valančiūnas     | Mindaugas       |             23 |            26 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Valančiūnas     | Tomas           |             31 |             4 |            0|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Tomas           |             25 |             9 |            2|
----------------------------------------------------------------------------------------------------

Lentelė Nr.7
2025-01-07
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Neptūnas        | Brazdeikis      | Mindaugas       |             29 |            30 |            0|
----------------------------------------------------------------------------------------------------
Rytas           | Motiejūnas      | Justas          |             39 |            22 |            0|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Brazdeikis      | Lukas           |             16 |            26 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Paulius         |             16 |             8 |            3|
----------------------------------------------------------------------------------------------------
Rytas           | Mačiulis        | Lukas           |             17 |            24 |            2|
----------------------------------------------------------------------------------------------------
Neptūnas        | Motiejūnas      | Martynas        |             40 |            22 |            3|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Jankūnas        | Tomas           |             15 |            22 |            2|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Petrauskas      | Paulius         |             38 |             7 |            2|
----------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Petrauskas      | Mindaugas       |             21 |            29 |            3|
----------------------------------------------------------------------------------------------------
Juventus        | Kazlauskas      | Justas          |             22 |            22 |            4|
----------------------------------------------------------------------------------------------------

Lentelė Nr.8
2025-01-08
----------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned|
----------------------------------------------------------------------------------------------------
Žalgiris        | Brazdeikis      | Dainius         |             31 |             4 |            3|
----------------------------------------------------------------------------------------------------
Neptūnas        | Petrauskas      | Arnas           |             30 |            30 |            4|
----------------------------------------------------------------------------------------------------
Neptūnas        | Jankūnas        | Jonas           |             37 |             3 |            1|
----------------------------------------------------------------------------------------------------
Žalgiris        | Sabonis         | Jonas           |             30 |            14 |            4|
----------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             15 |             4 |            4|
----------------------------------------------------------------------------------------------------
Cbet            | Motiejūnas      | Justas          |             10 |            11 |            3|
----------------------------------------------------------------------------------------------------
Žalgiris        | Kuzminskas      | Lukas           |             28 |             1 |            1|
----------------------------------------------------------------------------------------------------
Juventus        | Valančiūnas     | Mindaugas       |             11 |            13 |            2|
----------------------------------------------------------------------------------------------------
Rytas           | Kuzminskas      | Lukas           |             21 |            15 |            1|
----------------------------------------------------------------------------------------------------
Neptūnas        | Petrauskas      | Lukas           |             22 |            13 |            0|
----------------------------------------------------------------------------------------------------
```



```
Players
---------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Position            |
---------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          | centras             |
---------------------------------------------------------------------------
Cbet            | Jankūnas        | Martynas        | lengvasis puolėjas  |
---------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         | lengvasis puolėjas  |
---------------------------------------------------------------------------
Juventus        | Sabonis         | Mantas          | sunkusis puolėjas   |
---------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           | lengvasis puolėjas  |
---------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           | lengvasis puolėjas  |
---------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           | atakuojantis gynėjas|
---------------------------------------------------------------------------
Cbet            | Motiejūnas      | Dainius         | įžaidėjas           |
---------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        | sunkusis puolėjas   |
---------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          | centras             |
---------------------------------------------------------------------------
Šiauliai        | Petrauskas      | Tomas           | lengvasis puolėjas  |
---------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       | įžaidėjas           |
---------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       | lengvasis puolėjas  |
---------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          | lengvasis puolėjas  |
---------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           | lengvasis puolėjas  |
---------------------------------------------------------------------------

Refined Data
---------------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             11 |            30 |            0 | atakuojantis gynėjas|
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             11 |            30 |            0 | sunkusis puolėjas   |
---------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       |             14 |            23 |            0 | įžaidėjas           |
---------------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           |             14 |            23 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         |             11 |            25 |            1 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          |             11 |            25 |            1 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             22 |            23 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             29 |            30 |            0 | atakuojantis gynėjas|
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             29 |            30 |            0 | sunkusis puolėjas   |
---------------------------------------------------------------------------------------------------------------------------
Cbet            | Jankūnas        | Martynas        |             18 |            15 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Cbet            | Motiejūnas      | Dainius         |             18 |            15 |            0 | įžaidėjas           |
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             22 |            13 |            0 | atakuojantis gynėjas|
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             22 |            13 |            0 | sunkusis puolėjas   |
---------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             33 |            19 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          |             39 |            22 |            0 | centras             |
---------------------------------------------------------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       |             39 |            22 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             18 |            28 |            1 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Cbet            | Jankūnas        | Martynas        |             32 |            15 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Cbet            | Motiejūnas      | Dainius         |             32 |            15 |            0 | įžaidėjas           |
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             15 |            23 |            1 | atakuojantis gynėjas|
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             15 |            23 |            1 | sunkusis puolėjas   |
---------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           |             32 |            13 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          |             32 |            13 |            0 | centras             |
---------------------------------------------------------------------------------------------------------------------------
Šiauliai        | Petrauskas      | Tomas           |             32 |            13 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             28 |             9 |            0 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         |             17 |            23 |            1 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          |             17 |            23 |            1 | lengvasis puolėjas  |
---------------------------------------------------------------------------------------------------------------------------
Juventus        | Sabonis         | Mantas          |             31 |             4 |            0 | sunkusis puolėjas   |
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             28 |             3 |            0 | atakuojantis gynėjas|
---------------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             28 |             3 |            0 | sunkusis puolėjas   |
---------------------------------------------------------------------------------------------------------------------------
```



```
Sorted Data
--------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
--------------------------------------------------------------------------------------------------------------------
Cbet            | Jankūnas        | Martynas        |             16 |            15 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Cbet            | Jankūnas        | Martynas        |             32 |            15 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Cbet            | Motiejūnas      | Dainius         |             16 |            15 |            0 | įžaidėjas           |
--------------------------------------------------------------------------------------------------------------------
Cbet            | Motiejūnas      | Dainius         |             32 |            15 |            0 | įžaidėjas           |
--------------------------------------------------------------------------------------------------------------------
Juventus        | Sabonis         | Mantas          |             31 |             4 |            0 | sunkusis puolėjas   |
--------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             22 |            23 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             33 |            19 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             16 |            26 |            1 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Lietkabelis     | Sabonis         | Jonas           |             28 |             9 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             11 |            30 |            0 | sunkusis puolėjas   |
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             29 |            30 |            0 | sunkusis puolėjas   |
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             22 |            13 |            0 | sunkusis puolėjas   |
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             15 |            23 |            1 | sunkusis puolėjas   |
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Kazlauskas      | Martynas        |             28 |             3 |            0 | sunkusis puolėjas   |
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             11 |            30 |            0 | atakuojantis gynėjas|
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             29 |            30 |            0 | atakuojantis gynėjas|
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             22 |            13 |            0 | atakuojantis gynėjas|
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             15 |            23 |            1 | atakuojantis gynėjas|
--------------------------------------------------------------------------------------------------------------------
Neptūnas        | Mačiulis        | Jonas           |             28 |             3 |            0 | atakuojantis gynėjas|
--------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         |             11 |            25 |            1 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Sabonis         | Dainius         |             17 |            23 |            1 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          |             11 |            25 |            1 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Pieno Žvaigždės | Valančiūnas     | Mantas          |             17 |            23 |            1 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Rytas           | Kleiza          | Mindaugas       |             39 |            22 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          |             39 |            22 |            0 | centras             |
--------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          |             32 |            13 |            0 | centras             |
--------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kuzminskas      | Jonas           |             32 |            13 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Šiauliai        | Petrauskas      | Tomas           |             32 |            13 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Mindaugas       |             14 |            23 |            0 | įžaidėjas           |
--------------------------------------------------------------------------------------------------------------------
Žalgiris        | Petrauskas      | Lukas           |             14 |            23 |            0 | lengvasis puolėjas  |
--------------------------------------------------------------------------------------------------------------------

Lowest Position Players
--------------------------------------------------------------------------------------------------------------------
Team Name       | Last Name       | First Name      | Played Minutes | Points Earned | Fouls Earned | Position            |
--------------------------------------------------------------------------------------------------------------------
Šiauliai        | Kazlauskas      | Justas          |             32 |            13 |            0 | centras             |
--------------------------------------------------------------------------------------------------------------------
Rytas           | Valančiūnas     | Justas          |             39 |            22 |            0 | centras             |
--------------------------------------------------------------------------------------------------------------------
```

5.8. Dėstytojo pastabos

Nėra pastabų