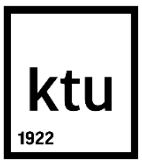


Kauno technologijos universitetas
Informatikos fakultetas

Objektinis programavimas 2 (P175B123)

Laboratorinių darbų ataskaita

Arnas Tamašauskas IFF-9/11

Studentas

Lekt. Lauraitis Andrius

Dėstytojas

Kaunas 2020

TURINYS

1. Rekursija (L1)

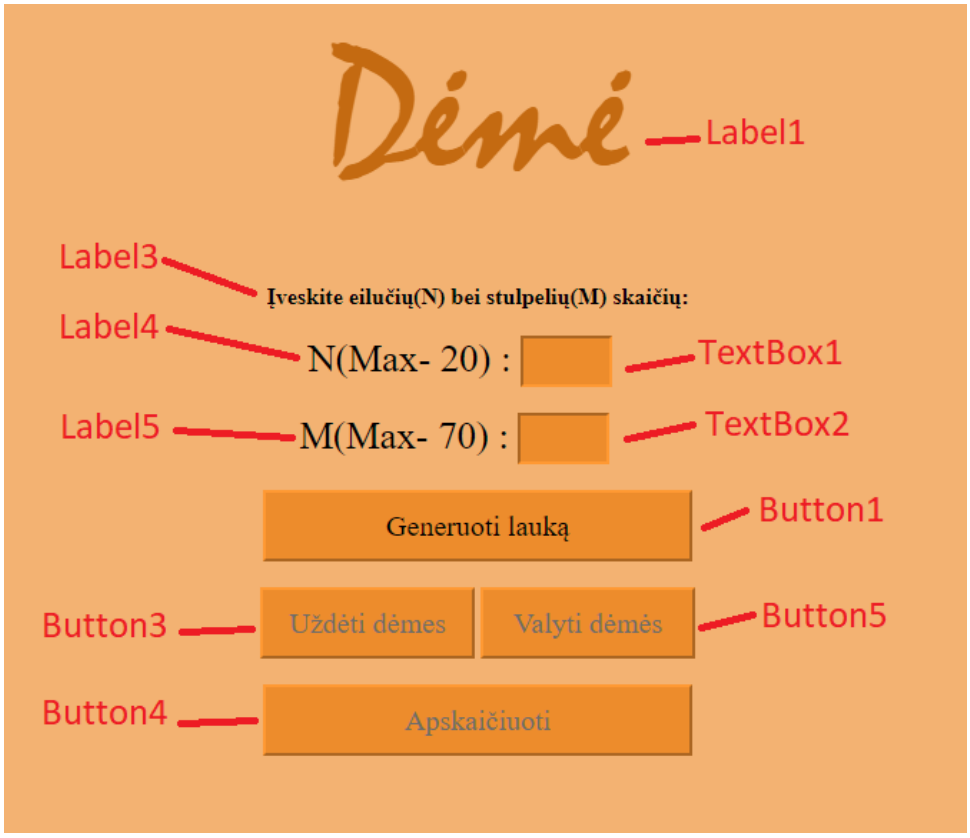
1.1. Darbo užduotis

LD_19. Dėmė.
Turime popieriaus lapą langeliais. Langelių eilutės ir stulpeliai sunumeruoti, pradedant vienetu. Numeracijos pradžia yra kairysis viršutinis langelis. Vaikas padažė teptuką dažuose ir krestelėjo virš lapo. Suskaičiuokite, kiek dėmių lape ir iš kiek langelių sudaryta didžiausia dėmė. Langeliai arba tušti, arba dažyti. Pusiau dažytų nėra. Tai pačiai dėmei priklauso visi dažyti langeliai, kurių bent viena koordinatė skiriasi vienetu nuo gretimo dažyto langelio. Reikia parašyti dėmių generatorių, kuris NxM lape nudažytų atsitiktinai trečdalį langelių, kai 1<=N<=20 ir 1<=M<=70. Čia N – eilučių skaičius, o M – stulpelių skaičius. N ir M reikšmės įvedamos klaviatūra. Ekrane parodykite dėmėmis išmargintą lapą. Tuščius langelius žymėkite tašku, dažytus langelius žymėkite žvaigždute * arba spalva. Apačioje parašykite, kiek yra dėmių, kiek langelių sudaro didžiausią dėmę ir didžiausios dėmės vieno bet kurio langelio koordinates: eilutės numerį ir stulpelio numerį.
Sugeneruoto lauko ir rezultatų pavyzdys, kai N 10 ir M 15.

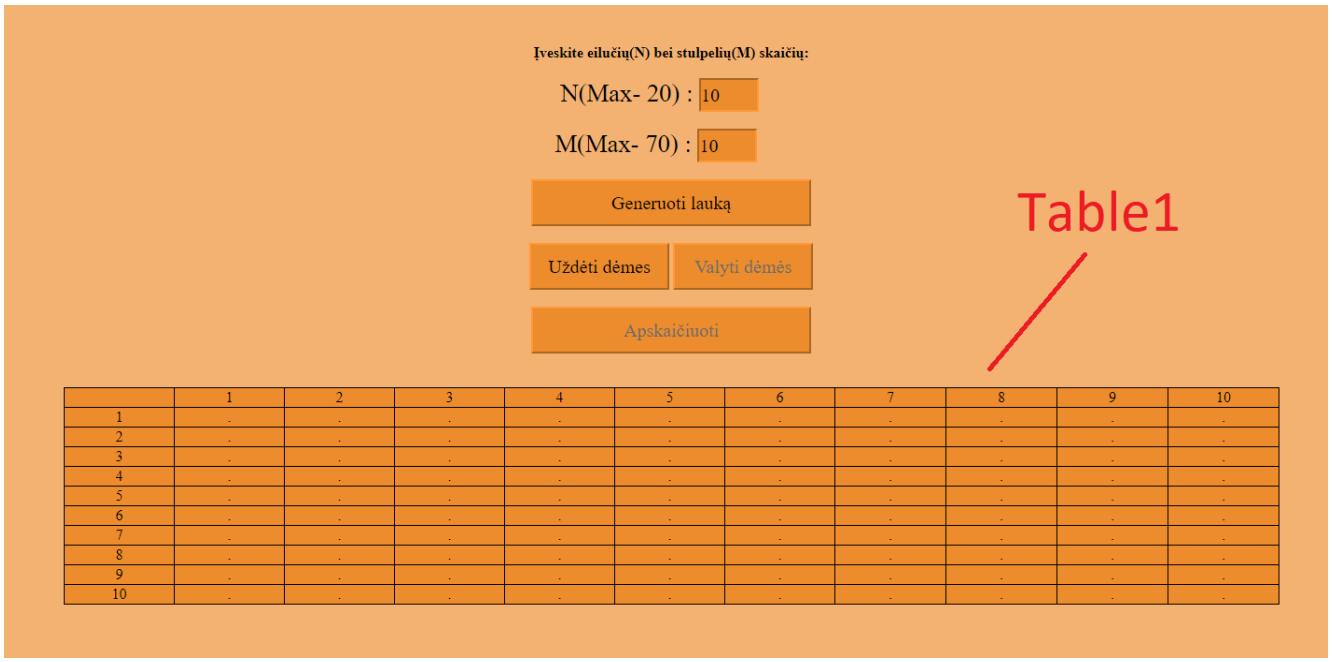
```
    1  2  3  4  5  6  7  8  9  10 11 12 13 14 15
1   *  *  *  .  .  .  *  *  *  .  .  .  .  .  .
2   *  *  *  .  .  *  .  .  *  .  .  .  .  .  *
3   *  .  .  .  *  .  .  .  *  .  .  .  .  *  .
4   *  *  .  .  *  .  .  .  *  .  .  .  *  .  .
5   .  .  .  .  .  .  .  .  *  .  .  *  .  .  *
6   .  *  *  .  .  .  .  .  *  .  .  .  .  .  *
7   .  *  *  .  .  .  .  .  *  .  .  .  .  .  *
8   .  .  .  .  *  .  .  .  *  .  .  .  .  *  .
9   *  *  *  *  .  .  .  .  *  .  .  .  .  *  .
10  .  .  .  .  .  *  *  *  *  *  *  *  .  *  *
Dėmių skaičius: 6. Didžiausia dėmė: 21. Eilutė: 3. Stulpelis: 5.
```

1.2. Grafinės vartotojo sąsajos schema

Paveikslėlis Nr.1 Pradinis sąsajos vaizdas:

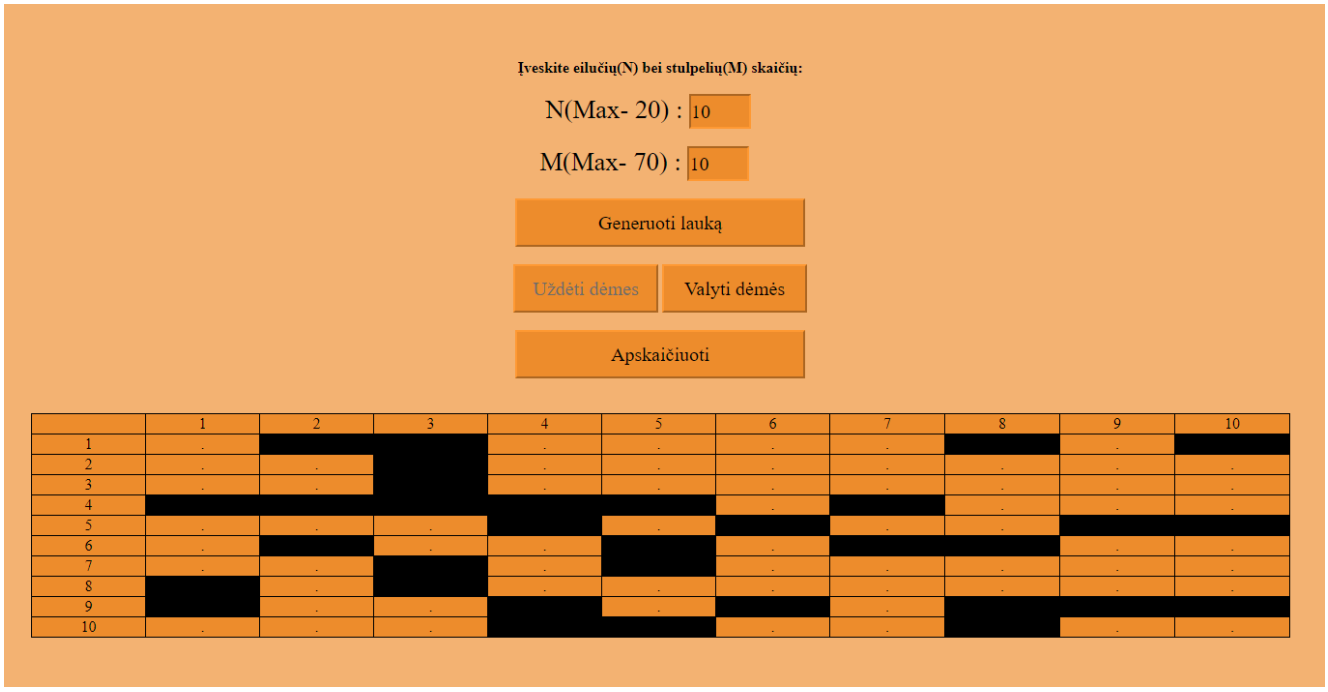


Paveikslėlis Nr.2 Vaizdas sugeneravus tuščią lapą:



	1	2	3	4	5	6	7	8	9	10
1	.	.	.	.	.	.	.	.	.	.
2	.	.	.	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.	.	.
6	.	.	.	.	.	.	.	.	.	.
7	.	.	.	.	.	.	.	.	.	.
8	.	.	.	.	.	.	.	.	.	.
9	.	.	.	.	.	.	.	.	.	.
10	.	.	.	.	.	.	.	.	.	.

Paveikslėlis Nr.3 Vaizdas paspaudus mygtuką „Uždėti dėmes“:



	1	2	3	4	5	6	7	8	9	10
1	.	■	■	.	.	.	.	■	.	■
2	.	.	■	.	.	.	.	.	.	.
3	.	.	■	.	.	.	.	.	.	.
4	■	■	■	■	■	.	■	.	.	.
5	.	.	.	■	.	■	.	.	■	■
6	.	■	.	.	■	.	■	■	.	.
7	.	.	■	.	■	.	.	.	.	.
8	■	.	.	.	.	.	.	.	.	.
9	■	.	.	■	.	■	.	■	■	■
10	.	.	.	■	■	.	.	■	.	.

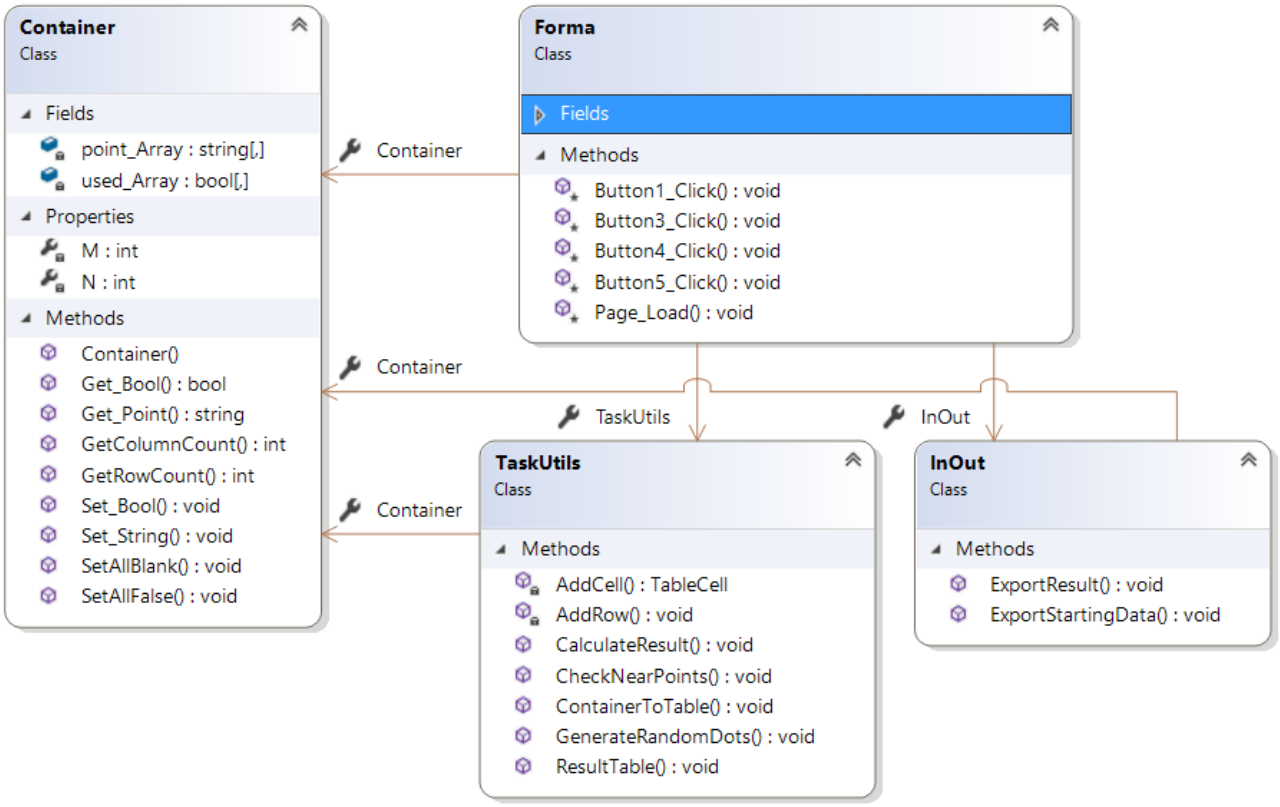
Paveikslėlis Nr.4 Vaizdas paspaudus mygtuką „Apskaičiuoti“:

1.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė
Label1	Font	Mistral, 100pt
	ForeColor	#C46A11
	Text	Dėmė
Label3	Font	Malgun Gothic Semilight
	Text	Įveskite eilučių(N) bei stulpelių(M) skaičių:
Label4	Font	Malgun Gothic Semilight, 20pt
	Text	N(Max- 20) :
Label5	Font	Malgun Gothic Semilight, 20pt
	Text	M(Max- 70) :
Label6	Font	Malgun Gothic Semilight
	Text	REZULTATAI
TextBox1	Font	Malgun Gothic Semilight, 15pt
	BackColor	#ED8C2C
	BorderColor	#FF9933
TextBox2	BackColor	#ED8C2C
	BorderColor	#FF9933
Button1	BackColor	#ED8C2C
	BorderColor	#FF9933
	Text	Generuoti lauką
	Font	Malgun Gothic Semilight, 15pt
Button2	BackColor	#ED8C2C
	BorderColor	#FF9933
	Text	Uždėti dėmes
	Font	Malgun Gothic Semilight, 15pt

Button5	BackColor	#ED8C2C
	BorderColor	#FF9933
	Text	Valyti dėmės
	Font	Malgun Gothic Semilight, 15pt
Button4	BackColor	#ED8C2C
	BorderColor	#FF9933
	Text	Apskaičiuoti
	Font	Malgun Gothic Semilight, 15pt
Table1	HorizontalAlign	Center
	GridLines	Both
	Font	Malgun Gothic Semilight
	BorderColor	Black
	BorderStyle	Solid
Table2	HorizontalAlign	Center
	GridLines	Both
	Font	Malgun Gothic Semilight
	BorderColor	Black
	BorderStyle	Solid
RangeValidator1	ErrorMessage	Netinkamas eilučių skaičius!
	Font	Malgun Gothic Semilight
	ControlToValidate	TextBox1
	MaximumValue	20
	MinimumValue	1
RangeValidator2	ErrorMessage	Netinkamas stulpelių skaičius!
	Font	Malgun Gothic Semilight
	ControlToValidate	TextBox2
	MaximumValue	70
	MinimumValue	1
RequiredFieldValidator1	ErrorMessage	Būtina įvesti eilučių kiekį !
	ControlToValidate	TextBox1
	Text	*
	ErrorMessage	Būtina įvesti stulpelių kiekį !
	ControlToValidate	TextBox2
	Text	*

1.4. Klasių diagrama



1.5. Programos vartotojo vadovas

Vartotojas turi įvesti norimą eilučių(N) skaičių ir stulpelių(M) skaičių. Įvedus norimas reikšmes ir paspaudus mygtuką „Generuoti lauką“, sudaromas tuščias lapas (lentelė) pagal duotas N ir M reikšmes. Paspaudus mygtuką „Uždėti dėmes“ atsitiktiniu būdu nudažomas trečdalis lapo(uždedamas „*“ simbolis bei nudažoma atitinkama lentelės dalis). Norit išvalyti lapą, spaudžiamas mygtukas „Valyti dėmes“. Jeigu nudažytas lapas vartotojui yra tinkamas, spaudžiamas mygtukas „Apskaičiuoti“ ir į atskirą lentelę išvedamas atskirų dėmių skaičius, didžiausios dėmės dydis bei jos bet kurio taško eilutės ir stulpelio numeris.

1.6. Programos tekstas

Container.cs

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

namespace L1
{
    /// <summary>
    /// Container class that holds the "Table"
    /// </summary>
    public class Container
    {
        // Array to know which sign is at a specifc point
        private string[,] point_Array = new string[22, 72];
        // Array to know which points are painted
        // and to know the limits of table
        private bool[,] used_Array = new bool[22, 72];
        private int N { get; set; } // row number
        private int M { get; set; } // column number

        public Container(int n, int m)
        {
            this.N = n;
            this.M = m;

            SetAllBlank();
            SetAllFalse();
        }

        /// <summary>
        /// // Makes a starting "Empty" table
        /// </summary>
        public void SetAllBlank()
        {
            for (int i = 0; i < 22; i++)
            {
                for (int j = 0; j < 72; j++)
                {
                    if(i == 0 && j == 0)
                    {
                        point_Array[i, j] = " ";
                    }

                    // Infomration line for rows
                    if (i == 0 && j != 0)
                    {
                        point_Array[i, j] = Convert.ToString(j);
                    }

                    // Infomration line for columns
                    if (i != 0 && j == 0)
                    {
                        point_Array[i, j] = Convert.ToString(i);
                    }

                    // "." - empty point
                    if(i != 0 && j != 0)
                    {
                        point_Array[i, j] = ".";
                    }
                }
            }
```

```
}

/// <summary>
/// Sets every point to blank - "Not painted"
/// </summary>
public void SetAllFalse()
{
    for (int i = 1; i <= this.N; i++)
    {
        for (int j = 1; j <= this.M; j++)
        {
            this.used_Array[i, j] = false;
        }
    }
}

/// <summary>
/// Used to get an array element
/// </summary>
/// <param name="x"> row number </param>
/// <param name="y"> column number </param>
/// <returns> the string element of an array </returns>
public string Get_Point(int x, int y)
{
    return this.point_Array[x, y];
}

/// <summary>
/// Gets the bool value of specific point
/// </summary>
/// <param name="x"> row number </param>
/// <param name="y"> column number </param>
/// <returns> bool value of point </returns>
public bool Get_Bool(int x, int y)
{
    return this.used_Array[x, y];
}

/// <summary>
/// Sets a new string value for a point
/// </summary>
/// <param name="x"> row number </param>
/// <param name="y"> column number </param>
/// <param name="sign"> new string sign </param>
public void Set_String(int x, int y, string sign)
{
    this.point_Array[x, y] = sign;
}

/// <summary>
/// Sets a new bool value for a point
/// </summary>
/// <param name="x"> row number </param>
/// <param name="y"> column number </param>
/// <param name="sign"> new bool sign </param>
public void Set_Bool(int x, int y, bool sign)
{
    this.used_Array[x, y] = sign;
}

/// <summary>
/// Returns the number of rows
/// </summary>
/// <returns></returns>
public int GetRowCount()
{
```

```
            return this.N;
        }

        /// <summary>
        /// Returns the number of columns
        /// </summary>
        /// <returns></returns>
        public int GetColumnCount()
        {
            return this.M;
        }
    }
}
```

InOut.cs

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

namespace L1
{
    public class InOut
    {
        public Container Container
        {
            get => default(Container);
            set
            {
            }
        }

        /// <summary>
        /// Prints the painted table, number of rows and columns to a text file
        /// </summary>
        /// <param name="container"> Container object </param>
        public static void ExportStartingData(Container container)
        {
            List<string> lines = new List<string>();

            lines.Add("N(Eilučių skaičius) : " + container.GetRowCount());
            lines.Add("M(Stuleplių skaičius) : " + container.GetColumnCount());
            lines.Add("");
            lines.Add("Sugeneruota lentelė:");

            for (int i = 0; i <= container.GetRowCount(); i++)
            {
                lines.Add(new string('-', container.GetColumnCount() * 5 + 6));

                string line = "|";

                for (int j = 0; j <= container.GetColumnCount(); j++)
                {
                    line += " ";
                    line += string.Format("{0,2}", container.Get_Point(i, j));
                    line += " |";
                }

                lines.Add(line);
            }

            lines.Add(new string('-', container.GetColumnCount() * 5 + 6));
```

```
            File.WriteAllLines
                (@"C:\Users\arntam1\Desktop\arntam1\02-
20\L1(1)\L1\L1\App_Data\StartingData.txt",
                lines);

        }

        /// <summary>
        /// Exports the results to a txt file
        /// </summary>
        /// <param name="spotCount"> The total amoun of spots </param>
        /// <param name="spotBiggest"> The size of the biggest spot </param>
        /// <param name="row"> row number of the biggest spot </param>
        /// <param name="column"> column number of the biggest spot </param>
        public static void ExportResult(int spotCount, int spotBiggest, int row, int column)
        {
            List<string> lines = new List<string>();

            lines.Add("Dėmių skaičius: " + spotCount);
            lines.Add("Didžiausia dėmė: " + spotBiggest);
            lines.Add("Eilutė: " + row);
            lines.Add("Stulpelis: " + column);

            File.WriteAllLines
                (@"C:\Users\arntam1\Desktop\arntam1\02-20\L1(1)\L1\L1\App_Data\Result.txt",
                lines);
        }

    }
}
```

TaskUtils.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Drawing;
using System.Drawing.Imaging;

namespace L1
{
    public class TaskUtils
    {
        public Container Container
        {
            get => default(Container);
            set
            {
            }
        }

        /// <summary>
        /// Transfers the data from container into a table in Web
        /// </summary>
        /// <param name="container"> A container object </param>
        /// <param name="table"> A table object in the web </param>
        public void ContainerToTable(Container container, Table table)
        {
            for (int i = 0; i <= container.GetRowCount(); i++)
            {
                AddRow(container, i, table);
            }
```

```
}

// Adds a row
private void AddRow(Container container, int rowCount, Table table)
{
    TableRow row = new TableRow();

    for (int i = 0; i <= container.GetColumnCount(); i++)
    {
        row.Cells.Add(AddCell(container, rowCount, i));
    }

    table.Rows.Add(row);
}

/// <summary>
/// Adds a cell
/// </summary>
/// <param name="container"> Container object</param>
/// <param name="rowIndex"> the number of current row </param>
/// <param name="columnIndex"> the number of current column </param>
/// <returns></returns>
private TableCell AddCell(Container container, int rowIndex, int columnIndex)
{
    TableCell cell = new TableCell();

    cell.Text = Convert.ToString(container.Get_Point(rowIndex, columnIndex));

    // Paints all painted table cells Black
    if(cell.Text == "*")
    {
        cell.BackColor = Color.Black;
    }

    return cell;
}

/// <summary>
/// Paints random points until a third of a table is painted
/// </summary>
/// <param name="container"></param>
public void GenerateRandomDots(Container container)
{
    Random rnd = new Random();
    int paintedCount = 0;

    while(paintedCount < ((container.GetColumnCount()) *
        (container.GetRowCount()))/3)
    {
        int row = rnd.Next(1, container.GetRowCount() + 1);
        int column = rnd.Next(1, container.GetColumnCount() + 1);

        if (!container.Get_Point(row, column).Equals("*"))
        {
            container.Set_String(row, column, "*");
            container.Set_Bool(row, column, true);
            paintedCount++;
        }
    }
}

/// <summary>
/// Prints out the results to a table in Web
/// </summary>
/// <param name="spotCount"> the amount of spots </param>
/// <param name="spotBiggest"> the size of the biggest spot </param>
```

```
/// <param name="spotRow"> row number of any point of biggest spot </param>
/// <param name="spotColumn"> column number of any point of biggest spot</param>
/// <param name="table"> The table object in Web </param>
public void ResultTable(int spotCount, int spotBiggest,
    int spotRow, int spotColumn, Table table)
{
    // Adding the amount of spots
    TableCell count = new TableCell();
    count.Text = Convert.ToString(spotCount);
    TableCell count1 = new TableCell();
    count1.Text = "Dėmių skaičius";
    TableRow tableRow1 = new TableRow();
    tableRow1.Cells.Add(count1);
    tableRow1.Cells.Add(count);
    table.Rows.Add(tableRow1);

    // Adding the number of biggest spot
    TableCell biggest = new TableCell();
    biggest.Text = Convert.ToString(spotBiggest);
    TableCell biggest1 = new TableCell();
    biggest1.Text = "Didžiausia dėmė";
    TableRow tableRow2 = new TableRow();
    tableRow2.Cells.Add(biggest1);
    tableRow2.Cells.Add(biggest);
    table.Rows.Add(tableRow2);

    // Adding the row number of  biggest spot
    TableCell row = new TableCell();
    row.Text = Convert.ToString(spotRow);
    TableCell row1 = new TableCell();
    row1.Text = "Eilutė";
    TableRow tableRow3 = new TableRow();
    tableRow3.Cells.Add(row1);
    tableRow3.Cells.Add(row);
    table.Rows.Add(tableRow3);

    // Adding the column number of  biggest spot
    TableCell column = new TableCell();
    column.Text = Convert.ToString(spotColumn);
    TableCell column1 = new TableCell();
    column1.Text = "Stulpelis";
    TableRow tableRow4 = new TableRow();
    tableRow4.Cells.Add(column1);
    tableRow4.Cells.Add(column);
    table.Rows.Add(tableRow4);
}

/// <summary>
/// Calculates the amount of spots, biggest spot
/// number of points, spot row and column number
/// </summary>
/// <param name="container"> A container object </param>
/// <param name="spotCount"> the amount of spots </param>
/// <param name="spotBiggest"> the size of the biggest spot </param>
/// <param name="spotRow"> row number of any point of biggest spot </param>
/// <param name="spotColumn"> column number of any point of biggest spot</param>
public void CalculateResult(Container container, ref int spotCount,
    ref int spotBiggest, ref int spotRow, ref int spotColumn)
{
    int spotSize;
    spotBiggest = 0;

    // Goes through every point
    for (int i = 1; i <= container.GetRowCount(); i++)
    {
        for (int j = 1; j <= container.GetColumnCount(); j++)
```

```
                {
                    // If point is painted
                    if(container.Get_Bool(i, j) == true)
                    {
                        spotCount++;
                        spotSize = 1;

                        // Means we already used
                        container.Set_Bool(i, j, false);
                        // Checks if spot connects to another near spot
                        CheckNearPoints(container, ref spotSize, i, j);

                        // Checks whether the currect spots is the biggest yet
                        if(spotSize > spotBiggest)
                        {
                            spotRow = i;
                            spotColumn = j;
                            spotBiggest = spotSize;
                        }
                    }
                }
            }
        }

        /// <summary>
        /// Checks all 9 near coordinates of a point
        /// </summary>
        /// <param name="container"></param>
        /// <param name="spotSize"></param>
        /// <param name="x"></param>
        /// <param name="y"></param>
        public void CheckNearPoints(Container container, ref int spotSize, int x, int y)
        {
            // Goes through (x-1)(x)(x+1) coordinates
            for(int i = - 1; i <= 1; i++)
            {
                // Goes through (y-1)(y)(y+1) coordinates
                for (int j = -1; j <= 1; j++)
                {
                    // Checks whetеher the point is painted or not
                    if (container.Get_Bool(x + i, y + j) == true)
                    {
                        spotSize++;
                        container.Set_Bool(x + i, y + j, false);
                        // Checks other near points
                        CheckNearPoints(container, ref spotSize, x + i, y + j);
                    }
                }
            }
        }
    }
}
```

Forma.aspx.cs

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

namespace L1
{
    public partial class Forma
    {
```

```
public TaskUtils TaskUtils
{
    get => default(TaskUtils);
    set
    {
    }
}

public Container Container
{
    get => default(Container);
    set
    {
    }
}

public InOut InOut
{
    get => default(InOut);
    set
    {
    }
}

protected void Page_Load(object sender, EventArgs e)
{
    Button3.Enabled = false;
    Button5.Enabled = false;
    Button4.Enabled = false;
    Label6.Visible = false;

    Table1.Visible = false;
    Table2.Visible = false;
}

// Button for generating an empty table
protected void Button1_Click(object sender, EventArgs e)
{
    // Takes the starting Info
    Container container = new Container(Convert.ToInt16(TextBox1.Text),
        Convert.ToInt16(TextBox2.Text));

    // Prints the starting "Empty" table to the Web
    TaskUtils obj = new TaskUtils();
    obj.ContainerToTable(container, Table1);

    Session["Container"] = container;
    Button3.Enabled = true;
    Button5.Enabled = false;
    Label6.Visible = false;

    Table1.Visible = true;
}

// Button for generating random points
protected void Button3_Click(object sender, EventArgs e)
{
    Table1.Rows.Clear();

    Container container = (Container)Session["Container"];

    TaskUtils obj = new TaskUtils();
    // Generates random painted points
    obj.GenerateRandomDots(container);
    // Prints the  table to the Web
    obj.ContainerToTable(container, Table1);
```

```
            Button3.Enabled = false;
            Button5.Enabled = true;
            Button4.Enabled = true;

            Table1.Visible = true;
        }

        // Button for calculating the number of spots and the biggest spot
        protected void Button4_Click(object sender, EventArgs e)
        {
            Container container = (Container)Session["Container"];
            TaskUtils obj = new TaskUtils();

            int spotCount = 0;
            int spotBiggest = 0;
            int spotRow = 0;
            int spotColumn = 0;

            // Calculates ant prints the results to the Web
            obj.CalculateResult(container, ref spotCount,
                ref spotBiggest, ref spotRow, ref spotColumn);
            obj.ResultTable(spotCount, spotBiggest, spotRow, spotColumn, Table2);
            obj.ContainerToTable(container, Table1);

            Label6.Visible = true;
            Table2.Visible = true;

            Table1.Visible = true;

            // Exports the information to the files
            InOut.ExportStartingData(container);
            InOut.ExportResult(spotCount, spotBiggest, spotRow, spotColumn);
        }

        // Button for erasing painted points
        protected void Button5_Click(object sender, EventArgs e)
        {
            Table1.Rows.Clear();

            Container container = (Container)Session["Container"];
            container.SetAllBlank();
            container.SetAllFalse();

            TaskUtils obj = new TaskUtils();
            obj.ContainerToTable(container, Table1);

            Button3.Enabled = true;
            Button5.Enabled = false;

            Table1.Visible = true;
        }
    }
}
```

Forma.aspx

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

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        #form1
```

```
        {
            text-align: center;
        }

        table
        {
            table-layout: fixed;
            width: 100%;
            background-color: #ed8c2c;
        }

        body
        {
            background-color: #f3b272;
        }

    </style>

</head>
<body>
    <form id="form1" runat="server" aria-orientation="horizontal">
        <asp:Label ID="Label1" runat="server" Font-Bold="False"
            Font-Names="Mistral" Font-Size="100pt" Text="Dėmė"
            ForeColor="#C46A11"></asp:Label>
        <br />
        <br />
        <br />
        <asp:Label ID="Label3" runat="server"
            Text="Įveskite eilučių(N) bei stulpelių(M) skaičių:"
            Font-Bold="True" Font-Names="Malgun Gothic Semilight"></asp:Label>
        <br />
        <asp:ValidationSummary ID="ValidationSummary1"
            runat="server" ForeColor="Black" Font-Names="Malgun Gothic Semilight" />
        <br />
               
               
               
               
               
           
        <asp:Label ID="Label4" runat="server"
            Font-Size="20pt" Text="N(Max- 20) : "
            Font-Names="Malgun Gothic Semilight"></asp:Label>
        <asp:TextBox ID="TextBox1" runat="server"
            Height="30px" OnTextChanged="TextBox1_TextChanged"
            Width="60px" Font-Names="Malgun Gothic Semilight"
            BackColor="#ED8C2C" BorderColor="#FF9933" Font-Size="15pt"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
            runat="server" ControlToValidate="TextBox1"
            ErrorMessage="Būtina įvesti eilučių kiekį !"
            ForeColor="Black"> *</asp:RequiredFieldValidator>
        <asp:RangeValidator ID="RangeValidator1" runat="server"
            ControlToValidate="TextBox1" ErrorMessage="Netinkamas eilučių skaičius!"
            ForeColor="Black" MaximumValue="20" MinimumValue="1"
            Type="Integer" BorderStyle="None"
            Font-Names="Malgun Gothic Semilight"></asp:RangeValidator>
        <br />
        <br />
                 
                 
                 
                 
            
        <asp:Label ID="Label5" runat="server" Font-Size="20pt"
            Text="M(Max- 70) : " Font-Names="Malgun Gothic Semilight"></asp:Label>
        <asp:TextBox ID="TextBox2" runat="server" Height="30px"
            Width="60px" Font-Names="Malgun Gothic Semilight"
            BackColor="#ED8C2C" BorderColor="#FF9933"
            Font-Size="15pt"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
            runat="server" ControlToValidate="TextBox2"
            ErrorMessage="Būtina įvesti stulpelių kiekį !"
            ForeColor="Black">*</asp:RequiredFieldValidator>
```

```
        <asp:RangeValidator ID="RangeValidator2"
            runat="server" ControlToValidate="TextBox2"
            ErrorMessage="Netinkamas stulpelių skaičius!"
            ForeColor="Black" MaximumValue="70" MinimumValue="1"
            Type="Integer" Font-Names="Malgun Gothic Semilight"></asp:RangeValidator>
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" Height="50px"
            OnClick="Button1_Click" Text="Generuoti lauką"
            Width="300px" Font-Names="Malgun Gothic Semilight"
            BackColor="#ED8C2C" BorderColor="#FF9933" Font-Size="15pt" />
        <br />
        <br />
        <asp:Button ID="Button3" runat="server"
            Height="50px" Text="Uždėti dėmes" Width="150px"
            OnClick="Button3_Click" Font-Names="Malgun Gothic Semilight"
            BackColor="#ED8C2C" BorderColor="#FF9933" Font-Size="15pt" />
        <asp:Button ID="Button5" runat="server"
            OnClick="Button5_Click" Text="Valyti dėmės"
            Width="150px" Height="50px" Font-Names="Malgun Gothic Semilight"
            BackColor="#ED8C2C" BorderColor="#FF9933" Font-Size="15pt" />
        <br />
        <br />
        <asp:Button ID="Button4" runat="server" Height="50px"
            Text="Apskaičiuoti" Width="300px" OnClick="Button4_Click"
            Font-Names="Malgun Gothic Semilight" BackColor="#ED8C2C"
            BorderColor="#FF9933" Font-Size="15pt" />
        <br />
        <br />
        <br />
        <asp:Table ID="Table1" runat="server" BorderColor="Black"
            BorderStyle="Solid" BorderWidth="1px" GridLines="Both"
            Height="16px" style="margin-left: 301px; text-align: center;"
            Width="1303px" HorizontalAlign="Center"
            Font-Names="Malgun Gothic Semilight">
        </asp:Table>
        <br />
        <asp:Label ID="Label6" runat="server" Font-Bold="True"
            Text="REZULTATAI" Font-Names="Malgun Gothic Semilight"></asp:Label>
        <br />
        <br />
        <asp:Table ID="Table2" runat="server" BorderColor="Black"
            BorderStyle="Solid" BorderWidth="1px"
            GridLines="Both" Height="16px"
            style="margin-left: 702px; text-align: center;"
            Width="500px" HorizontalAlign="Center"
            Font-Names="Malgun Gothic Semilight">
        </asp:Table>
        <br />
        <br />
        <br />
    </form>
</body>
</html>
```

1.7. Pradiniai duomenys ir rezultatai

1 variantas

StartingData.txt

StartingData - Notepad

File Edit Format View Help

```
N(Eilučių skaičius) : 10
M(Stuleplių skaičius) : 10

Sugeneruota lentelė:
-------------------------------------------------------
|    | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
-------------------------------------------------------
| 1  | . | * | . | . | . | . | * | . | * | * |
-------------------------------------------------------
| 2  | * | . | . | . | * | . | . | . | . | . |
-------------------------------------------------------
| 3  | . | . | * | * | * | . | . | * | * | . |
-------------------------------------------------------
| 4  | . | . | . | * | . | * | . | * | . | . |
-------------------------------------------------------
| 5  | . | . | * | * | . | . | . | . | . | * |
-------------------------------------------------------
| 6  | . | . | . | . | * | * | . | . | . | . |
-------------------------------------------------------
| 7  | . | . | . | * | . | * | * | . | * | . |
-------------------------------------------------------
| 8  | . | . | . | * | . | . | . | . | . | . |
-------------------------------------------------------
| 9  | * | * | * | . | * | * | . | . | . | . |
-------------------------------------------------------
| 10 | . | . | * | . | . | * | . | * | * | . |
-------------------------------------------------------
```

Result.txt

Result - Notepad

File Edit Format View Help

```
Dėmių skaičius: 8
Didžiausia dėmė: 21
Eilutė: 2
Stulpelis: 5
```

2 Variantas

StartingData.txt

StartingData - Notepad

File Edit Format View Help

```
N(Eilučių skaičius) : 3
M(Stuleplių skaičius) : 3

Sugeneruota lentelė:
---------------------
|    | 1 | 2 | 3 |
---------------------
| 1 | . | * | . |
---------------------
| 2 | . | . | . |
---------------------
| 3 | * | . | * |
---------------------
```

Result.txt

Result - Notepad

File Edit Format View Help

```
Dėmių skaičius: 3
Didžiausia dėmė: 1
Eilutė: 1
Stulpelis: 2
```

3 Variantas

StartingData.txt

(1 dalis)

StartingData - Notepad

File Edit Format View Help

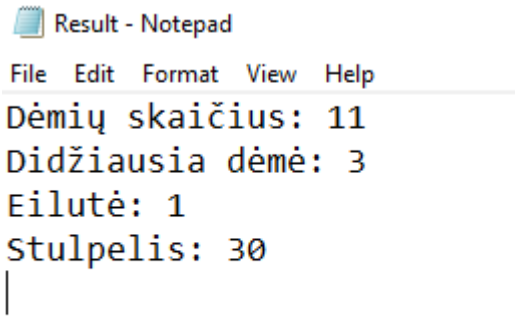
```
N(Eilučių skaičius) : 1
M(Stuleplių skaičius) : 50

Sugeneruota lentelė:
-----------------------------------------------------------------------------------------------------
|    | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
-----------------------------------------------------------------------------------------------------
| 1 | . | . | . | * | . | . | . | . | . | . | . | . | . | * | * | . | . | * | * | . | . | . | . | * | . |
-----------------------------------------------------------------------------------------------------
```

(2 dalis)

26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
*	.	*	.	*	*	*	.	*	.	.	.	.	*	.	.	.	*	*	.	.	*	.	.	.

Result.txt



1.8. Dėstytojo pastabos

- P7
- P13

2. Dinaminis atminties valdymas (L2)

2.1. Darbo užduotis

2.2. Grafinės vartotojo sąsajos schema

2.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė

2.4. Klasių diagrama

2.5. Programos vartotojo vadovas

2.6. Programos tekstas

2.7. Pradiniai duomenys ir rezultatai

2.8. Dėstytojo pastabos

3. Bendrinės klasės ir testavimas (L3)

3.1. Darbo užduotis

3.2. Grafinės vartotojo sąsajos schema

3.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė

3.4. Klasių diagrama

3.5. Programos vartotojo vadovas

3.6. Programos tekstas

3.7. Pradiniai duomenys ir rezultatai

3.8. Dėstytojo pastabos

4. Polimorfizmas ir išimčių valdymas (L4)

4.1. Darbo užduotis

4.2. Grafinės vartotojo sąsajos schema

4.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė

4.4. Klasių diagrama

4.5. Programos vartotojo vadovas

4.6. Programos tekstas

4.7. Pradiniai duomenys ir rezultatai

4.8. Dėstytojo pastabos

5. Deklaratyvusis programavimas (L5)

5.1. Darbo užduotis

5.2. Grafinės vartotojo sąsajos schema

5.3. Sąsajoje panaudotų komponentų keičiamos savybės

Komponentas	Savybė	Reikšmė

5.4. Klasių diagrama

5.5. Programos vartotojo vadovas

5.6. Programos tekstas

5.7. Pradiniai duomenys ir rezultatai

5.8. Dėstytojo pastabos