Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«Российский университет транспорта»

(ФГАОУ ВО РУТ(МИИТ), РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа № 20

по дисциплине: «Программирование и основы алгоритмизации»

на тему: «Диаграммы»

Выполнил: ст. гр. ТУУ-211

Воронов А.В.

Вариант №3

Проверил: к.т.н., доц. Сафронов А.И.

1. **Формулировка цели работы**

Освоить навыки графического отображения статистических данных и построения диаграмм на платформе Windows Forms.

1. **Описание задачи**



Рисунок 1 – Задача к разработке

Загружать данные, представленные на диаграмме, в программное обеспечение из *csv*-файла. Воспроизвести отображение представленных двух видов диаграмм. Предоставить пользователю возможность обмена положением проиллюстрированных диаграмм, а также возможность отображения только одного из видов диаграмм.

1. **Сеть Петри запрограммированного технологического процесса**

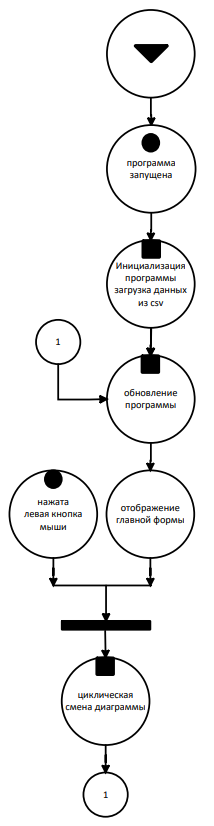


Рисунок 2 – Сеть Петри запрограммированного технологического процесса

1. **Блок-схема алгоритма**
2. **Подбор тестовых примеров**

В качестве тестовых данных данные из диаграмм приведены к формату csv:

Файл data.csv содержит следующий текст:

Анна\_15

Борис\_28

Виктор\_8

Галина\_22

Дарья\_11

1. **Листинг (код) составленного программного обеспечения**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

using System.IO;

using System.Linq;

using System.Windows.Forms;

namespace CustomChart

{

public partial class ChartForm : Form

{

private List<Tuple<string, int>> data;

private bool isBarChart = true;

private const int MaxValue = 30;

private const int Padding = 20;

private const int BarWidth = 60;

private const int PieRadius = 100;

private const int HorizontalOffset = 50;

private const int VerticalOffset = 20;

private const int ArrowSize = 10;

private const int HoleRadius = 30;

private Font chartFont = new Font("Times New Roman", 14, FontStyle.Bold);

public ChartForm()

{

SuspendLayout();

ClientSize = new Size(600, 400);

ResumeLayout(false);

LoadData();

}

private void LoadData()

{

data = File.ReadAllLines("data.csv")

.Select(line => line.Split('\_'))

.Select(parts => new Tuple<string, int>(parts[0], int.Parse(parts[1])))

.ToList();

}

protected override void OnPaint(PaintEventArgs e)

{

base.OnPaint(e);

DrawChart(e.Graphics);

}

private void DrawChart(Graphics g)

{

if (data.Count == 0) return;

List<string> names = data.Select(d => d.Item1).ToList();

List<int> values = data.Select(d => d.Item2).ToList();

int chartWidth = ClientSize.Width - Padding \* 2 - HorizontalOffset;

int chartHeight = ClientSize.Height - Padding \* 2 - VerticalOffset \* 2;

Rectangle chartRect = new Rectangle(Padding + HorizontalOffset, Padding + VerticalOffset, chartWidth, chartHeight);

if (isBarChart)

DrawBarChart(g, chartRect, names, values);

else

DrawPieChart(g, chartRect, names, values);

}

private void DrawBarChart(Graphics g, Rectangle chartRect, List<string> names, List<int> values)

{

// Оси

g.DrawLine(Pens.Black, chartRect.Left, chartRect.Bottom, chartRect.Right, chartRect.Bottom);

g.DrawLine(Pens.Black, chartRect.Left, chartRect.Bottom, chartRect.Left, chartRect.Top);

// Стрелка на оси Y

Point[] arrowPoints = new Point[]

{

new Point(chartRect.Left - ArrowSize, chartRect.Top),

new Point(chartRect.Left, chartRect.Top - ArrowSize),

new Point(chartRect.Left + ArrowSize, chartRect.Top)

};

g.FillPolygon(Brushes.Black, arrowPoints);

// Подпись оси Y

g.DrawString("Время в мин", chartFont, Brushes.Black, new Point(chartRect.Left - 50, chartRect.Top - 35));

// Чёрточки с шагом 5

for (int y = 5; y <= MaxValue; y += 5)

{

int yPos = chartRect.Bottom - (int)((double)y / MaxValue \* (chartRect.Height - 20));

g.DrawLine(Pens.Black, chartRect.Left - 5, yPos, chartRect.Left + 5, yPos);

g.DrawString(y.ToString(), chartFont, Brushes.Black, new Point(chartRect.Left - 30, yPos - 5));

}

// Прерывистая линия на отметке 15

int targetY = chartRect.Bottom - (int)((double)15 / MaxValue \* (chartRect.Height - 20));

Pen dashPen = new Pen(Color.Black, 1);

dashPen.DashStyle = DashStyle.Dash;

g.DrawLine(dashPen, chartRect.Left, targetY, chartRect.Right, targetY);

// Столбцы

double xStep = (double)chartRect.Width / (names.Count + 1);

double yScale = (double)(chartRect.Height - 20) / MaxValue;

for (int i = 0; i < names.Count; i++)

{

int x = chartRect.Left + (int)((i + 1) \* xStep - BarWidth / 2);

int y = chartRect.Bottom - (int)(values[i] \* yScale);

int height = chartRect.Bottom - y;

// Рисование столбца

using (SolidBrush brush = new SolidBrush(Color.FromArgb(255, 255, 192, 203)))

{

g.FillRectangle(brush, x, y, BarWidth, height);

}

g.DrawRectangle(Pens.Black, x, y, BarWidth, height);

// Подпись под столбцом

SizeF textSize = g.MeasureString(names[i], chartFont);

int textX = x + (BarWidth - (int)textSize.Width) / 2;

g.DrawString(names[i], chartFont, Brushes.Black, new Point(textX, chartRect.Bottom + 20));

}

}

private void DrawPieChart(Graphics g, Rectangle chartRect, List<string> names, List<int> values)

{

Point center = new Point(chartRect.Left + chartRect.Width / 2, chartRect.Top + chartRect.Height / 2);

int total = values.Sum();

double angle = 0;

Color[] colors = { Color.SkyBlue, Color.LightGreen, Color.Tomato, Color.Gold, Color.Plum };

int radius = Math.Min(chartRect.Width, chartRect.Height) / 2 - 20;

// Рисование сегментов

for (int i = 0; i < names.Count; i++)

{

double sliceAngle = (double)values[i] / total \* 360;

SolidBrush brush = new SolidBrush(colors[i % colors.Length]);

// Рисование сегмента

g.FillPie(brush, new Rectangle(center.X - radius, center.Y - radius, radius \* 2, radius \* 2),

(float)angle, (float)sliceAngle);

g.DrawPie(Pens.Black, new Rectangle(center.X - radius, center.Y - radius, radius \* 2, radius \* 2),

(float)angle, (float)sliceAngle);

// Рисование дырки

g.FillEllipse(Brushes.White, new Rectangle(center.X - HoleRadius, center.Y - HoleRadius, HoleRadius \* 2, HoleRadius \* 2));

g.DrawEllipse(Pens.Black, new Rectangle(center.X - HoleRadius, center.Y - HoleRadius, HoleRadius \* 2, HoleRadius \* 2));

// Рисование числовых значений на сегментах

double textAngle = angle + sliceAngle / 2;

int textX = center.X + (int)(radius \* 0.8 \* Math.Cos(textAngle \* Math.PI / 180));

int textY = center.Y + (int)(radius \* 0.8 \* Math.Sin(textAngle \* Math.PI / 180));

g.DrawString(values[i].ToString(), chartFont, Brushes.Black, new Point(textX, textY));

// Рисование имён в стороне

int legendX = chartRect.Right + 20;

int legendY = chartRect.Top + (int)(i \* 30);

g.DrawString($"{names[i]}: {values[i]}", chartFont, Brushes.Black, new Point(legendX, legendY));

g.FillRectangle(new SolidBrush(colors[i % colors.Length]), legendX - 15, legendY, 10, 10);

angle += sliceAngle;

}

}

protected override void OnMouseClick(MouseEventArgs e)

{

base.OnMouseClick(e);

isBarChart = !isBarChart;

Invalidate();

}

}

public static class Program

{

public static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new ChartForm());

}

}

}

1. **Графический пользовательский интерфейс программного обеспечения и его описание**

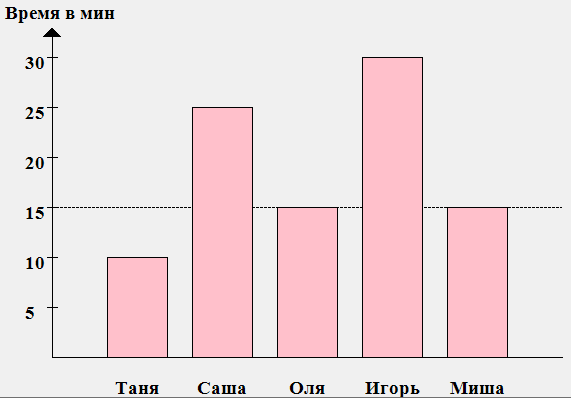


Рисунок 3 – Интерфейс главной формы  
(столбчатая диаграмма)

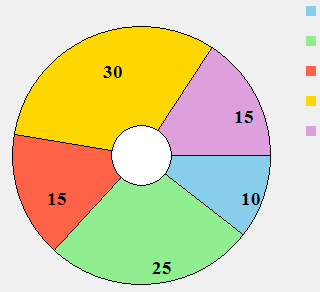


Рисунок 4 – Интерфейс главной формы

(пончиковая диаграмма)

Основное пространство формы занимает столбчатая диаграмма, отображающая имена и присвоенные им значения времени. Цвет столбцов статичен.

По нажатию левой кнопки мыши, она меняется на пончиковую. Есть разделение сегментов по цветам для большей читаемости статистики. Так же каждое значение подписано на своём сегменте пончиковой диаграммы.

1. **Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению**



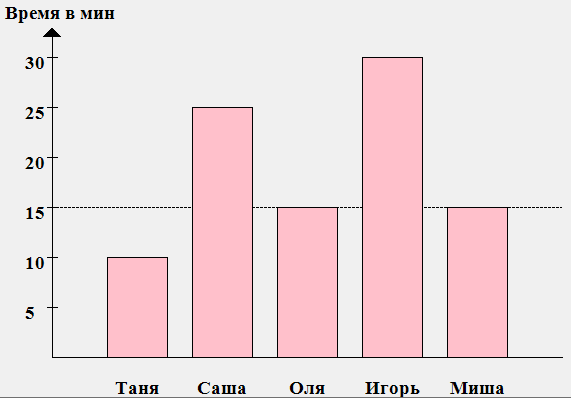


Рисунок 5 – Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению.

Данные, представленные на диаграмме, соответствуют данным, представленным в csv файле. Общий вид диаграммы соответствует, данному в задаче.

1. **Расчёт тестовых примеров с использованием составленного программного обеспечения**

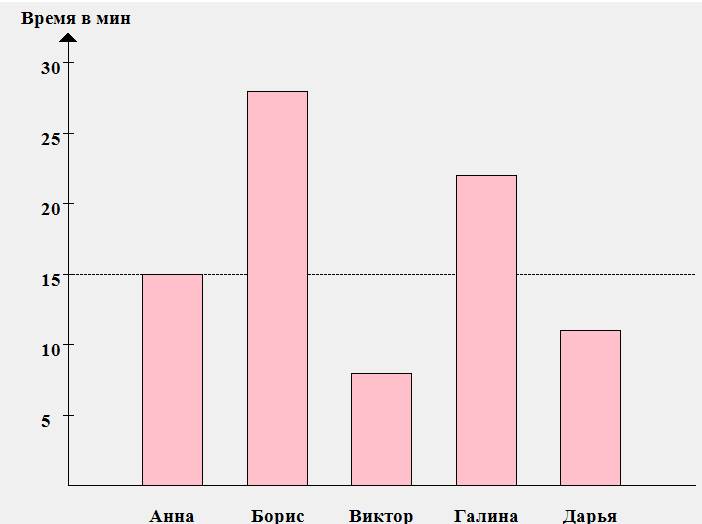


Рисунок 6 – столбчатая диаграмма, построенная по данным из

выбранного тестового примера

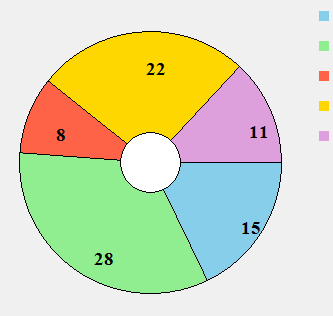


Рисунок 7-пончиковая диаграмма построенная по этим же данным

**Формулировка вывода о проделанной работе**

Освоил навыки графического отображения статистических данных и построения диаграмм на платформе Windows Forms. В ходе работы с диаграммами, удалось подробнее познакомиться с программированием графических элементов на платформе WindowsForms. Также закрепил навыки работы с файлами, парсингом данных из csv-файлов.