Тема 7: Построение графиков и диаграмм

**Цель работы:** получить практические навыки разработки проектов с использованием команд графики для построения графиков, диаграмм, зависимостей по таблице данных.

Задание 1. Основной проект

Разработать программу для ввода данных по вашему варианту задания и добавления в таблицу вводимых данных. Опрос пользователей производится при помощи отдельной вкладки на форме, результаты опроса в виде таблицы должны отображаться на другой вкладке. Для организации вкладок используйте компонент TabControl. Хранение и обработку данных организовать с помощью дополнительного класса.

Задание 2. Сохранение и загрузка данных

Добавьте возможность сохранить результаты опроса в текстовый файл и загрузить результаты опроса из текстового файла в таблицу на форме.

Задание 3. Построение диаграммы

Добавьте в проект 3-ую вкладку, на которой требуется построить график или диаграмму (в зависимости от варианта). Построение диаграммы производиться по таблице данных с помощью компонента Сhart.

Для работы с этим компонентом необходимо подключить библиотеку System.Windows.Forms.DataVisualization.Charting.

Вариант 14

Разработать проект для ввода фамилий участников соревнований, длины дистанции (в м.) и времени (в сек.), добавления этих сведений таблицу на второй вкладке и поиска участников, показавших лучшую скорость бега. Также должна быть возможность загрузить данные из текстового файла (длина дистанции, количество участников соревнования, перечень фамилий участников и результат каждого участника) и сохранить данные в текстовый файл в таком же формате. На третьей вкладке построить столбчатую диаграмму, показывающую сравнение участников по скорости бега.

Выполнение задания

1. **Математическая модель**
   1. **Дано:**

distance – дистанция, которую пробегает участник (метры)

time – время, за которое участник пробегает дистанцию (секунды)

* 1. **Найти:**

speed – скорость участника

* 1. **Связь:**

1. **Проектирование пользовательского интерфейса**

Внешний вид проекта представлен на рисунке 1–3, значения измененных свойств компонентов представлены в таблице 1.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

label1

label2

numericUpDown1

dataGriedView1

button2

radioButton2

radioButton1

groupBox1

Form1

button1

**Рис. 1 Вкладка «Ввод данных»**

**Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание**

**Рис. 2 Вкладка «Таблица данных»**

**Изображение выглядит как текст, снимок экрана, диаграмма, График

Автоматически созданное описание**

**Рис. 3 Вкладка «Диаграмма»**

1. **Свойства компонентов формы**

Значения свойств, установленные на этапе конструирования интерфейса:

*Таблица 1.*

|  |  |  |
| --- | --- | --- |
| **Название компонента** | **Свойства** | **Значения** |
| Form1 | Text | «Form 1» |
| BackColor | Thistle |
| Font | Гарнитура: Microsoft Sans Serif |
| Размер: 8,25 |
| button1 | Text | **«**Вычислить**»** |
| Button2 | Text | **«**Заполнить**»** |
| groupBox1 | Text | «Способ ввода массива:» |
| radioButton1 | Text | «Вручную» |
| Checked | True |
| radioButton2 | Text | «Случайно» |
| Checked | False |
| dataGridView1 | AllowUserToAddRows | False |
| AllowUserToDeleteRows | False |
| AutoSizeColumnsMode | None |
| numericUpDown1 | Minimum | 1 |
| Maximum | 100 |
| label1 | Text | **«**Размер массива**»** |
| label2 | Text | **«**» |

1. **Сценарий использования программы**
   1. Запуск программы
   2. Ввод данных об участнике соревнования
   3. Нажатие на кнопку «Добавить»
   4. Переход на вкладку «Таблица данных»
   5. Нажатие на кнопку «Сохранить» для сохранения данных в текстовый файл
   6. Возможность нажатия на кнопку «Загрузить» для загрузки данных из текстового файла
   7. Переход на вкладку «Диаграмма»
   8. Возможность вернуться к пунктам 4.2 и/или 4.7
   9. Завершение работы программы
2. **Код программы**

Код модуля Form1

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

using ClassLibrary7;

namespace WindowsFormsApp7

{

public partial class Form1 : Form

{

private List<Participant> participants = new List<Participant>();

// Конструктор формы

public Form1()

{

InitializeComponent();

MaximumSize = new Size(571, 462);

MinimumSize = new Size(571, 462);

}

// Кнопка Добавить

private void addButton\_Click(object sender, EventArgs e)

{

try

{

string name = surnameTextBox.Text;

double distance = double.Parse(distanceTextBox.Text);

double time = double.Parse(timeTextBox.Text);

// создаем нового участника и добавляем его в список

var participant = new Participant(name, distance, time);

participants.Add(participant);

UpdateTable();

UpdateChart();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка ввода данных: " + ex.Message);

}

surnameTextBox.Text = "";

distanceTextBox.Text = "";

timeTextBox.Text = "";

}

// Кнопка Сохранить

private void saveButton\_Click(object sender, EventArgs e)

{

try

{

SaveFileDialog saveFileDialog = new SaveFileDialog

{

Filter = "Text Files (\*.txt)|\*.txt",

DefaultExt = "txt",

AddExtension = true

};

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

Participant.SaveToFile(participants, saveFileDialog.FileName);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

// Кнопка Загрузить

private void loadButton\_Click(object sender, EventArgs e)

{

try

{

OpenFileDialog openFileDialog = new OpenFileDialog

{

Filter = "Text Files (\*.txt)|\*.txt",

DefaultExt = "txt",

AddExtension = true

};

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

participants = Participant.LoadFromFile(openFileDialog.FileName);

UpdateTable();

UpdateChart();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

// Метод для обновления данных в таблице

private void UpdateTable()

{

dataGridView.Rows.Clear();

dataGridView.Columns.Clear();

dataGridView.Columns.Add("Name", "Фамилия");

dataGridView.Columns.Add("Distance", "Дистанция, м");

dataGridView.Columns.Add("Time", "Время, с");

dataGridView.Columns.Add("Speed", "Скорость, м/с");

// заполняем таблицу данными участников

foreach (var participant in participants)

{

dataGridView.Rows.Add(participant.Name, participant.Distance, participant.Time, participant.Speed.ToString("F2"));

}

}

// Метод для обновления данных на диаграмме

private void UpdateChart()

{

// очищаем существующие серии данных

chart1.Series.Clear();

// создаем новую серию данных для диаграммы

var series = new Series("Скорость, м/с")

{

ChartType = SeriesChartType.Column

};

// заполняем серию данными участников

foreach (var participant in participants)

{

series.Points.AddXY(participant.Name, participant.Speed);

}

// добавляем серию к диаграмме

chart1.Series.Add(series);

}

private void surnameTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsLetter(e.KeyChar) && !char.IsControl(e.KeyChar) && e.KeyChar != ' ')

{

e.Handled = true;

}

}

private void distanceTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar) && (e.KeyChar != ',' || distanceTextBox.Text.Contains(",")))

{

e.Handled = true;

}

}

private void timeTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar) && (e.KeyChar != ',' || timeTextBox.Text.Contains(",")))

{

e.Handled = true;

}

}

}

}

Код модуля Participant

using System;

using System.Collections.Generic;

using System.IO;

using System.Text;

namespace ClassLibrary7

{

public class Participant

{

// Свойства класса

public string Name { get; private set; }

public double Distance { get; private set; }

public double Time { get; private set; }

public double Speed => Distance / Time; // cкорость, вычисляемая на основе дистанции и времени

// Конструктор класса

public Participant(string name, double distance, double time)

{

Name = name;

Distance = distance;

Time = time;

}

// Метод для сохранения списка участников в файл

public static void SaveToFile(List<Participant> participants, string fileName)

{

try

{

using (StreamWriter sw = new StreamWriter(fileName, false, Encoding.UTF8))

{

sw.WriteLine(participants[0].Distance); // записываем дистанцию (одинаковую для всех участников)

sw.WriteLine(participants.Count); // записываем количество участников

foreach (var participant in participants) // записываем данные каждого участника

{

sw.WriteLine(participant.Name);

sw.WriteLine(participant.Time);

}

}

}

catch (Exception ex)

{

throw new Exception("Ошибка доступа к файлу: " + ex.Message);

}

}

// Метод для загрузки списка участников из файла

public static List<Participant> LoadFromFile(string fileName)

{

try

{

using (StreamReader sr = new StreamReader(fileName, Encoding.UTF8))

{

double distance = double.Parse(sr.ReadLine());

int count = int.Parse(sr.ReadLine());

List<Participant> participants = new List<Participant>();

for (int i = 0; i < count; i++)

{

string name = sr.ReadLine();

double time = double.Parse(sr.ReadLine());

participants.Add(new Participant(name, distance, time));

}

return participants;

}

}

catch (Exception ex)

{

throw new Exception("Ошибка доступа к файлу: " + ex.Message);

}

}

}

}

1. **Схемы основных подпрограмм**



**Рис. 4 Метод SaveToFile**



**Рис. 5 Метод saveButton\_Click**

1. **Протокол испытаний**

**Протокол испытаний:**

*Таблица 2 Протокол испытаний*

|  |  |  |  |
| --- | --- | --- | --- |
| **Проверяемые требования** | **Сообщения программы и вводимые значения** | **Ожидаемые результаты** | **Фактические результаты** |
| Случайное заполнение данных массива | Выбор способа заполнения «случайно», нажать кнопку «Заполнить», затем «Вычислить» | Datagridview заполняется случайными числовыми данными в диапазоне от -100 до 100. | Соответствуют ожиданиям  (См. Рис. 6). |
| Контроль ввода данных массива при ручном заполнении | Выбор способа заполнения «вручную» нажать кнопку «Заполнить»,  Входные данные:  0: шшшш  1: 1.9 | Выходные данные:  1: 0  2: 1 | Соответствуют ожиданиям  (См. Рис. 7). |
| Способность определения последнего максимального | Выбор способа заполнения «вручную» нажать кнопку «Заполнить»,  Входные данные:  0: 5  1: 9  2: 2  3: 9  нажать кнопку «Вычислить». | Вывод сообщения «Нет элементов после последнего максимального элемента». | Соответствуют ожиданиям  (См. Рис. 8). |
| Способность подсчета произведения чисел после последнего максимального | Выбор способа заполнения «вручную» нажать кнопку «Заполнить»,  Входные данные:  0: 10  1: 5  2: 2  нажать кнопку «Вычислить». | Вывод «Произведение элементов после последнего максимального: 10» | Соответствуют ожиданиям  (См. Рис. 9-10). |
| Работоспособность подсветки результата при наведении курсора | Наведение курсора мыши на поле результата и позже снятие курсора с поля | Поле изменит фоновый цвет пока курсор на нём и изменит его обратно при отведении курсора | Соответствуют ожиданиям  (См. Рис. 11-12). |

1. **Скриншоты на этапе тестирования проекта**

**Рис. 6**

**Рис. 7**

**Рис. 8**

**Рис. 9**

**Рис. 10**

**Рис. 11**