МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

|  |  |
| --- | --- |
|  | Утвержден на заседании кафедры  «Вычислительная техника»  "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 г.  Заведующий кафедрой                                              М.А. Митрохин |
|  |  |

**ОТЧЕТ ПО УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКЕ**

(2023/2024 учебный год)

                                         \_\_\_\_ Пырков Денис Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения1семестр 2

Период прохождения практики с 25.06.24 по 08.07.24

Кафедра «Вычислительная техника»

Заведующий кафедрой д.т.н., профессор, Митрохин М.А.

*(должность, ученая степень, ученое звание, Ф.И.О.)*

Руководитель практики к.т.н., доцент, Карамышева Н.С.

*(должность, ученая степень, ученое звание)*

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

|  |  |
| --- | --- |
|  | Утвержден на заседании кафедры  «Вычислительная техника»  "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 г.  Заведующий кафедрой                                              М.А. Митрохин |
|  |  |

**ИНДИВИДУАЛЬНЫЙ ПЛАН ПРОХОЖДЕНИЯ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2023/2024 учебный год)

                                            Пырков Денис Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения                  1                 семестр                 2

Период прохождения практики с 25.06.24 по 08.07.24

Кафедра «Вычислительная техника»

Заведующий кафедрой д.т.н., профессор, Митрохин М.А.

*(должность, ученая степень, ученое звание, Ф.И.О.)*

Руководитель практики к.т.н., доцент, Карамышева Н.С.

*(должность, ученая степень, ученое звание)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Планируемая форма работы во время практики | Количество часов | Календарные сроки проведения работы | Подпись  руководителя  практики от вуза |
| 1 | Выбор темы и разработка индивидуального плана проведения работ | 2 | 25.06.2024 -  25.06.2024 |  |
| 2 | Подбор и изучение материала по теме работы | 15 | 26.06.2024 –  28.06.24 |  |
| 3 | Разработка алгоритма | 43 | 01.07.24 –  03.07.24 |  |
| 4 | Описание алгоритма и программы | 18 | 03.07.24 –  04.07.24 |  |
| 5 | Тестирование | 5 | 04.07.24 –  05.07.24 |  |
| 6 | Получение и анализ результатов | 10 | 05.07.24 –  08.07.24 |  |
| 7 | Оформление отчёта | 15 | 05.07.24 –  08.07.2024 |  |
|  | **Общий объём часов** | 108 |  |  |

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЧЁТ**

**О ПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2023/2024 учебный год)

                                            Пырков Денис Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения1семестр 2

Период прохождения практики с 25.06.24 по 08.07.24

Кафедра «Вычислительная техника»

Пырков Д. А. выполнял практическое задание «Сортировка Шелла». На первоначальном этапе были изучен и проанализирован алгоритм сортировки Шелла, был выбран метод решения и язык программирования С#, на котором была написана программа сортировки массива. Протестировал и отладил программу. Оформил отчёт.

Бакалавр Пырков Д. А. \_\_\_\_\_\_\_\_\_\_\_\_ "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Руководитель Карамышева Н.С. \_\_\_\_\_\_\_\_\_\_\_\_ "\_\_\_" \_\_\_\_\_\_\_\_\_\_\_ 2024 г.

практики

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

**ОТЗЫВ**

**О ПРОХОЖДЕНИИ УЧЕБНОЙ (ОЗНАКОМИТЕЛЬНОЙ) ПРАКТИКИ**

(2023/2024 учебный год)

                                            Пырков Денис Алексеевич

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Наименование профиля подготовки «Прикладной искусственный интеллект»

Форма обучения – очная Срок обучения в соответствии с ФГОС – 4 года

Год обучения1семестр 2

Период прохождения практики с 25.06.24 по 08.07.24

Кафедра «Вычислительная техника»

В процессе выполнения практики Пырков Д. А. решал следующие задачи: разработка графического интерфейса программы, создание программы, выполняющей сортировку Шелла, тестирование программы.

За период выполнения практики были освоены основные понятия и технологии сортировки Шелла. Во время выполнения работы Пырков Д. А. показал себя ответственным, добросовестным учеником, знающим свой предмет, имеющим представление о современном состоянии науки, владеющим современными общенаучными знаниями по информатике и вычислительной технике, программированию и сортировке.

За выполнение работы Пырков Д. А. заслуживает оценки «\_\_\_\_\_\_».

Руководитель практики к.т.н., доцент, Карамышева Н.С. « » 2024 г.

Содержание

Введение7

1. Постановка задачи8

2. Выбор решения8

3. Описание программы9

4. Схема программы13

5. Тестирование программы14

6. Отладка программы14

7. Совместная работа16

Заключение19

Список используемых источников20

Приложение А. Листинг программы21

Приложение В. Окно программы25

**Введение**

В современном мире информационных технологий эффективная обработка данных играет ключевую роль в различных сферах деятельности. Одной из фундаментальных задач в области программирования является сортировка данных, которая позволяет упорядочивать информацию для дальнейшего анализа и использования. Задачи по сортировке данных часто встречаются в различных профессиональных областях. Алгоритмы сортировки составляют отдельный класс алгоритмов, которые используются почти во всех задачах обработки информации. Они тесно связаны друг с другом и применяются для обеспечения более быстрого поиска.

Кроме того, алгоритмы сортировки играют ключевую роль в решении многих задач, таких как поиск, фильтрация, группировка, аналитика данных и т.д. Они помогают упорядочивать данные таким образом, чтобы было удобно работать с ними и извлекать из них нужную информацию.

Таким образом, понимание и применение алгоритмов сортировки является необходимым навыком для специалистов, занимающихся обработкой и анализом данных, и позволяет им эффективно управлять информационными процессами.

Метод Шелла, или сортировка Шелла в структуре данных, представляет собой эффективный алгоритм сортировки сравнением на месте. Он назван в честь Дональда Шелла, когда он предложил первоначальную идею еще в 1959 году. Сортировка Шелла — это обобщенное расширение алгоритма сортировки вставками. Фундаментальная идея этого алгоритма сортировки состоит в том, чтобы сгруппировать элементы, находящиеся далеко друг от друга, и соответствующим образом отсортировать их. Затем постепенно уменьшается промежуток между ними. Сортировка Шелла превосходит среднее время выполнения сортировки вставками путем сравнения и обмена элементами, которые находятся далеко.

# **1. Постановка задачи**

Поставленная задача: необходимо заполнить массив из n-ого количества элементов случайными числами, записать данные элементы в отдельный файл. После этого выполнить сортировку Шелла над данными, находящимися в массиве, записать отсортированные данные в другой файл и посчитать время выполнения сортировки массива. Программа должна иметь графический интерфейс.

Использовать сервис GitHub для совместной работы. Создать и выложить коммиты, характеризующие действия, выполненные каждым участником бригады.

Оформить отчет по проведенной практике.

**2. Выбор решения**

Для разработки графического интерфейса данной программы будет использован язык программирования С#. С# – один из наиболее быстро растущих, востребованных и при этом «удобных» языков программирования. Это модификация фундаментального языка С от компании Microsoft, призванная создать наиболее универсальное средство для разработки программного обеспечения для большого количества устройств и операционных систем.

Для оптимизации вычислительных задач алгоритм был написан на языке С и интегрирован в C# при помощи DLL библиотеки. Язык C был выбран из-за его эффективности и низкоуровневого управления ресурсами, что позволяет существенно ускорить выполнение задач. Сочетание C# для общей логики и C для оптимизированных алгоритмов обеспечивает баланс между производительностью и удобством разработки.

В качестве среды программирования была выбрана программа Microsoft Visual Studio. Microsoft Visual Studio — это программная среда по разработке приложений для ОС Windows, как консольных, так и с графическим интерфейсом.

Для удобства совместной работы использовался мессенджер Discord. Он позволяет общаться как в голосовом, так и в текстовом чате. Это помогает координировать действия всей бригады и эффективнее работать над проектом.

**3. Описание программы**

При запуске программы открывается окно, в котором предлагается ввести размерность массива и диапазон значений его элементов. После того, как пользователь введёт указанные данные и нажмёт кнопку «Сортировать» массив будет сгенерирован и отсортирован. Информация о том, в каких файлах находятся сгенерированный и отсортированный массивы, а также о том, сколько времени заняла сортировка, будет отображена в сообщении, которое появится после того, как программа завершит свою работу.

Разработка графического интерфейса данной программы должна начинаться с инициализации основного окна, в котором будут расположены все остальные элементы. Основное окно имеет такие параметры, как имя (Form1), размеры, фоновый цвет, названия всех элементов управления и др.

// Form1

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.BackColor = System.Drawing.SystemColors.GradientActiveCaption;

this.ClientSize = new System.Drawing.Size(252, 285);

this.Controls.Add(this.ButtonSort);

this.Controls.Add(this.TextBoxMaxVol);

this.Controls.Add(this.TextBoxMinVol);

this.Controls.Add(this.label\_MaxVol);

this.Controls.Add(this.label\_MinVol);

this.Controls.Add(this.label\_Range);

this.Controls.Add(this.label\_Razm);

this.Controls.Add(this.TextBoxRazm);

this.Cursor = System.Windows.Forms.Cursors.Arrow;

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;

this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.Icon")));

this.Name = "Form1";

this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;

this.Text = "Massive sort";

this.ResumeLayout(false);

this.PerformLayout();

Далее нужно добавить в Form1 все необходимые элементы управления. В данной программе они бывают трёх видов:

1) Label – текст описания, предоставляющий пользователю информацию о программе во время её выполнения.

2) TextBox – текстовое поле, позволяющее пользователю вводить данные.

3) Button – кнопка, при нажатии на которую программа начнёт генерацию и сортировку массива.

В данной программе элемент типа «Label» встречается 4 раза. Каждый Label имеет свои координаты, размер, текст и др.

// label\_Razm

//

this.label\_Razm.AutoSize = true;

this.label\_Razm.Font = new System.Drawing.Font("Verdana", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label\_Razm.Location = new System.Drawing.Point(24, 31);

this.label\_Razm.Name = "label\_Razm";

this.label\_Razm.Size = new System.Drawing.Size(149, 14);

this.label\_Razm.TabIndex = 1;

this.label\_Razm.Text = "Размерность массива:";

//

// label\_Range

//

this.label\_Range.AutoSize = true;

this.label\_Range.Font = new System.Drawing.Font("Verdana", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label\_Range.Location = new System.Drawing.Point(24, 109);

this.label\_Range.Name = "label\_Range";

this.label\_Range.Size = new System.Drawing.Size(214, 14);

this.label\_Range.TabIndex = 2;

this.label\_Range.Text = "Диапазон значений элементов:";

//

// label\_MinVol

//

this.label\_MinVol.AutoSize = true;

this.label\_MinVol.Font = new System.Drawing.Font("Verdana", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label\_MinVol.Location = new System.Drawing.Point(24, 141);

this.label\_MinVol.Name = "label\_MinVol";

this.label\_MinVol.Size = new System.Drawing.Size(22, 14);

this.label\_MinVol.TabIndex = 3;

this.label\_MinVol.Text = "от";

//

// label\_MaxVol

//

this.label\_MaxVol.AutoSize = true;

this.label\_MaxVol.Font = new System.Drawing.Font("Verdana", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label\_MaxVol.Location = new System.Drawing.Point(24, 178);

this.label\_MaxVol.Name = "label\_MaxVol";

this.label\_MaxVol.Size = new System.Drawing.Size(23, 14);

this.label\_MaxVol.TabIndex = 4;

this.label\_MaxVol.Text = "до";

Элемент типа «TextBox» встречается 3 раза. Он имеет такие же параметры, как «Label».

// TextBoxRazm

//

this.TextBoxRazm.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.TextBoxRazm.Location = new System.Drawing.Point(27, 63);

this.TextBoxRazm.Name = "TextBoxRazm";

this.TextBoxRazm.Size = new System.Drawing.Size(109, 21);

this.TextBoxRazm.TabIndex = 0;

this.TextBoxRazm.TextChanged += new System.EventHandler(this.TextBoxRazm\_TextChanged);

//

// TextBoxMinVol

//

this.TextBoxMinVol.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.TextBoxMinVol.Location = new System.Drawing.Point(61, 141);

this.TextBoxMinVol.Name = "TextBoxMinVol";

this.TextBoxMinVol.Size = new System.Drawing.Size(75, 21);

this.TextBoxMinVol.TabIndex = 5;

this.TextBoxMinVol.TextChanged += new System.EventHandler(this.TextBoxMinVol\_TextChanged);

//

// TextBoxMaxVol

//

this.TextBoxMaxVol.Font = new System.Drawing.Font("Microsoft Sans Serif", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.TextBoxMaxVol.Location = new System.Drawing.Point(61, 174);

this.TextBoxMaxVol.Name = "TextBoxMaxVol";

this.TextBoxMaxVol.Size = new System.Drawing.Size(75, 21);

this.TextBoxMaxVol.TabIndex = 6;

this.TextBoxMaxVol.TextChanged += new System.EventHandler(this.TextBoxMaxVol\_TextChanged);

В окне данной программы присутствует всего одна кнопка. Она имеет такие же параметры, как у «Label» и «TextBox».

// ButtonSort

//

this.ButtonSort.Cursor = System.Windows.Forms.Cursors.Hand;

this.ButtonSort.Font = new System.Drawing.Font("Verdana", 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.ButtonSort.Location = new System.Drawing.Point(27, 222);

this.ButtonSort.Name = "ButtonSort";

this.ButtonSort.Size = new System.Drawing.Size(109, 27);

this.ButtonSort.TabIndex = 7;

this.ButtonSort.Text = "Сортировать";

this.ButtonSort.UseVisualStyleBackColor = true;

this.ButtonSort.Click += new System.EventHandler(this.ButtonSort\_Click);

**4. Схема программы**

На рисунке 1 представлена схема программы.

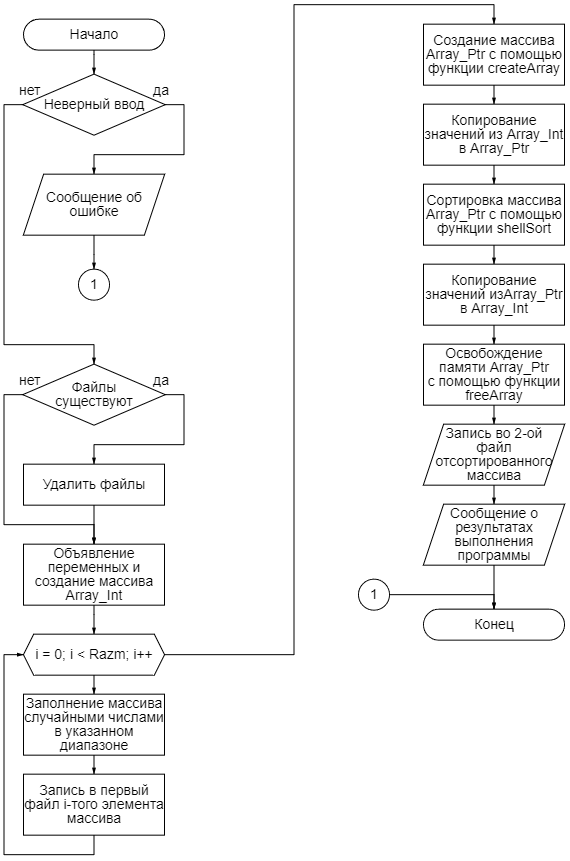


Рисунок 1 – Блок-схема программы

# **5. Тестирование программы**

На основании анализа данных, полученных в результате тестирования алгоритма сортировки Шелла (рисунок 2), можно сделать вывод, что продолжительность выполнения программы растёт экспоненциально. Другими словами, Время сортировки растет медленно при малых значениях размера массива. По мере увеличения размера массива время сортировки растет всё быстрее. Кривая на графике становится всё более крутой, что указывает на ускоряющееся возрастание времени сортировки. Таким образом, график демонстрирует квадратичную зависимость времени сортировки от размера массива.

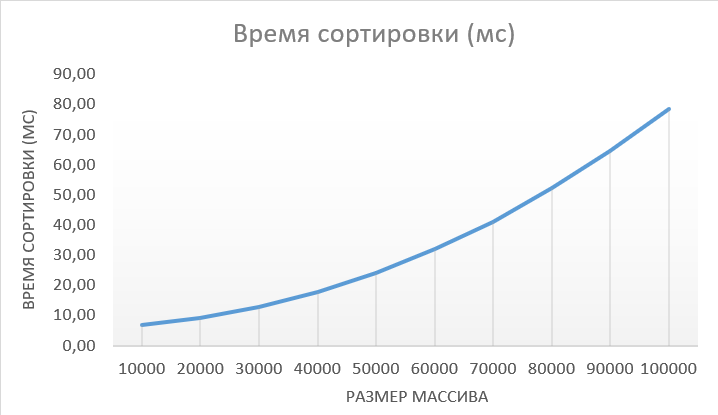


Рисунок 2 – Результаты тестирования

**6. Отладка программы**

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2022. Программа обладает всеми средствами необходимыми при разработке и отладке программы. Для отладки использовались несколько возможностей Visual Studio: точка останова, трассировка, анализ содержимого переменных. Рассмотрим каждый из этих инструментов и способы их использования.

**Точки останова (Breakpoints)**

**Точка останова** — это инструмент, позволяющий приостановить выполнение программы на определенной строке кода. Это дает возможность изучить состояние переменных и поток выполнения программы.

**Как установить точку останова:**

1. Откройте файл с исходным кодом.
2. Найдите строку, на которой вы хотите приостановить выполнение программы.
3. Щелкните левой кнопкой мыши в области отступа слева от этой строки или нажмите F9, чтобы установить/снять точку останова.

**Типы точек останова:**

* **Безусловные:** Останавливают программу всегда, когда выполняется указанная строка.
* **Условные:** Останавливают выполнение программы, только если выполнено определенное условие.
  + Щелкните правой кнопкой мыши по точке останова и выберите **Conditions**.
  + Установите условие, например x > 10.

**Трассировка (Debugging/Stepping)**

**Трассировка** позволяет пошагово выполнять код, чтобы отслеживать поведение программы и состояние переменных.

**Основные команды трассировки:**

* **F10 (Step Over):** Выполняет текущую строку и переходит к следующей, не входя внутрь вызова функции.
* **F11 (Step Into):** Входит внутрь вызываемой функции, если это возможно.
* **Shift + F11 (Step Out):** Выходит из текущей функции и возвращается к месту вызова.
* **F5 (Continue):** Продолжает выполнение программы до следующей точки останова.

**Анализ содержимого переменных**

Во время трассировки вы можете изучить значения переменных и объектов.

**Основные способы просмотра переменных:**

1. **Окно локальных переменных (Locals):**
   * Вменювыберите **Debug** > **Windows** > **Locals**.
   * Отображает все локальные переменные в текущем контексте.
2. **Окно автопеременных (Autos):**
   * Вменювыберите **Debug** > **Windows** > **Autos**.
   * Показывает переменные, связанные с текущей и предыдущей строкой кода.
3. **Окно наблюдения (Watch):**
   * В меню выберите **Debug** > **Windows** > **Watch** > **Watch 1** (или другие).
   * Позволяет вручную добавлять интересующие переменные и следить за их значениями.
4. **Выражения в строке кода (DataTip):**
   * Наведите курсор на переменную в коде во время трассировки, чтобы увидеть ее значение.
5. **Окно быстрого просмотра (QuickWatch):**
   * Выделите интересующую переменную и нажмите Shift + F9, чтобы открыть окно быстрого просмотра.

# **7. Совместная работа**

Во время работы над данной практикой наша бригада осуществляла совместную работу в GitHub.

Мною был написан код графического интерфейса, файл с ним был загружен на удаленный репозиторий Github, на ветку main.

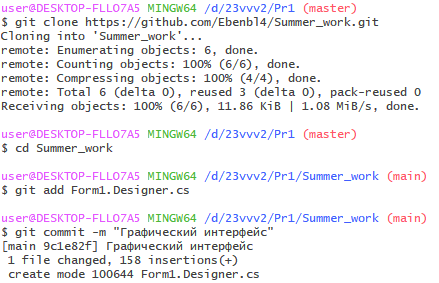




Рисунок 3 – Загрузка программы на удаленный репозиторий

Для удобства совместной работы использовался мессенджер Discord. Он позволяет общаться как в голосовом, так и в текстовом чате, а также пересылать файлы. Это помогает координировать действия всей бригады и эффективнее работать над проектом.



Рисунок 4 – Совместная работа в Discord

Ссылка на удаленный репозиторий:

<https://github.com/Ebenbl4/Summer_work.git>

**Заключение**

При выполнении данной работы были получены навыки совместной работы с помощью сервиса GitHub, навыки использования программы Git Bash. Был реализован алгоритм сортировки Шелла. Мною был разработан интерфейс программы и оформлен отчет по данной практике. При выполнении практической работы были улучшены базовые навыки программирования на языке С#. Улучшены навыки отладки, тестирования программ и работы с графическим интерфейсом. В дальнейшем программу можно улучшить путем добавления возможности заполнять массив вручную, отображать количество выполненных перестановок при сортировке и др.

**Список используемых источников**

1. Рихтер Д. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. (2022).

2. ГОСТ 19.701 – 90 Схемы алгоритмов, программ, данных и систем.

3. Сортировка Шелла [Электронный ресурс] – URL: <https://ru.wikipedia.org/wiki/Сортировка_Шелла> (дата обращения: 3.07.2024 г)

**Приложение A. Листинг программы**

Файл Form1.cs

using System;

using System.Diagnostics;

using System.IO;

using System.Runtime.InteropServices;

using System.Windows.Forms;

namespace Practica1

{

public partial class Form1 : Form

{

int Razm, MinVol, MaxVol;

const string File1Path = "Unsorted massive.txt";

const string File2Path = "Sorted massive.txt";

FileInfo File1Info = new FileInfo(File1Path);

FileInfo File2Info = new FileInfo(File2Path);

Stopwatch Sort\_Time = new Stopwatch();

public Form1()

{

InitializeComponent();

}

[DllImport("mySort.dll", CallingConvention = CallingConvention.StdCall)]

public static extern unsafe void shellSort(int size, int\* arrayPointer);

[DllImport("mySort.dll", CallingConvention = CallingConvention.StdCall)]

public static extern unsafe int\* createArray(int size);

[DllImport("mySort.dll", CallingConvention = CallingConvention.StdCall)]

public static extern unsafe void freeArray(int\* arrayPointer);

private void Write\_Generated\_Massive(string number)

{

try

{

using (StreamWriter writer = new StreamWriter(File1Path, true))

{

writer.WriteLine($"{number}");

}

}

catch (IOException ex)

{

MessageBox.Show($"Ошибка при записи в файл: {ex.Message}");

}

}

private void Write\_Sorted\_Massive(string number)

{

try

{

using (StreamWriter writer = new StreamWriter(File2Path, true))

{

writer.WriteLine($"{number}");

}

}

catch (IOException ex)

{

MessageBox.Show($"Ошибка при записи в файл: {ex.Message}");

}

}

private unsafe void ButtonSort\_Click(object sender, EventArgs e)

{

if ((TextBoxRazm.Text == "") || (TextBoxMinVol.Text == "") || (TextBoxMaxVol.Text == ""))

{

MessageBox.Show("Поля должны быть заполнены!");

}

else

{

if (File1Info.Exists) { File1Info.Delete(); }

if (File2Info.Exists) { File2Info.Delete(); }

Razm = Convert.ToInt32(TextBoxRazm.Text);

MinVol = Convert.ToInt32(TextBoxMinVol.Text);

MaxVol = Convert.ToInt32(TextBoxMaxVol.Text);

try

{

Random rand = new Random();

int[] Array\_Int = new int[Razm];

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

{

Array\_Int[i] = rand.Next(MinVol, MaxVol);

Write\_Generated\_Massive($"{Array\_Int[i]}");

}

Sort\_Time.Start();

IntPtr Array\_Ptr = (IntPtr)createArray(Razm);

Marshal.Copy(Array\_Int, 0, Array\_Ptr, Razm);

shellSort(Razm, (int\*)Array\_Ptr);

Marshal.Copy(Array\_Ptr, Array\_Int, 0, Razm);

freeArray((int\*)Array\_Ptr);

Sort\_Time.Stop();

string Array\_Str = string.Join("\n", Array\_Int);

Write\_Sorted\_Massive(Array\_Str);

string elapsedSeconds = Sort\_Time.Elapsed.TotalSeconds.ToString("F5");

MessageBox.Show($"Массив создан и отсортирован.\nСгенерированный массив находится в файле \"{File1Path}\".\nОтсортированный массив находится в файле \"{File2Path}\".\nВремя сортировки: {elapsedSeconds}c.");

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка: {ex.Message}");

}

}

}

private void TextBoxRazm\_TextChanged(object sender, EventArgs e)

{

}

private void TextBoxMinVol\_TextChanged(object sender, EventArgs e)

{

}

private void TextBoxMaxVol\_TextChanged(object sender, EventArgs e)

{

}

}

}

Файл mySort.dll

#include "pch.h"

#include "mySort.h"

#include <malloc.h>

void shellSort(int size, int\* arrayPointer) {

int step, i, j, temp;

for (step = size / 2; step > 0; step /= 2) {

for (i = step; i < size; i++) {

temp = arrayPointer[i];

for (j = i; j >= step; j -= step) {

if (temp < arrayPointer[j - step]) arrayPointer[j] = arrayPointer[j - step];

else break;

}

arrayPointer[j] = temp;

}

}

}

int\* createArray(int size) {

return (int\*)malloc(size \* sizeof(int));

}

void freeArray(int\* arrayPointer) {

free(arrayPointer);

}

**Приложение В. Окно программы**

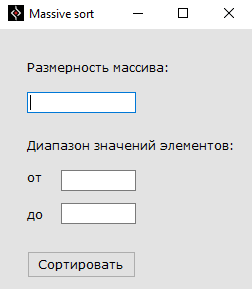


Рисунок 5 – Окно программы