Федеральное агентство по рыболовству
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Астраханский государственный технический университет»
Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована DQS по международному стандарту ISO 9001:2015

Институт Информационных технологий и коммуникаций

Направление 09.03.01 Информатика и вычислительная техника

Профиль Автоматизированные системы обработки информации и управления

Кафедра Автоматизированные системы обработки информации и управления

КУРСОВОЙ ПРОЕКТ

Исследование метода сортировки Хоара

по дисциплине «Алгоритмы и структуры данных»

Допущен к защите

«__» _____2023г.

Руководитель

_____

Оценка, полученная на защите

«_____»

Члены комиссии:

_____

_____

_____

Проект выполнил:
обучающийся группы ДИНРБ-21

Самодуров Вячеслав Александрович

_____

Руководитель

асс. Кравченкова Е.П.

Астрахань 2023

АСТРАХАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и коммуникаций

УТВЕРЖДАЮ
Заведующий кафедрой
д.т.н., проф.
Т.В. Хоменко _____
«____»_____2023 г.

Кафедра
«Автоматизированные системы
обработки информации и управления»

ЗАДАНИЕ

на выполнение курсового проекта

Обучающийся ***Самодуров Вячеслав Александрович***

Группа ***ДИНРб-21***

Дисциплина ***Алгоритмы и структуры данных***

Тема ***Исследование метода сортировки Хоара***

Дата получения задания «_____»_____2023г.

Срок представления обучающимся КП на кафедру «_____»_____2023г.

Руководитель ***ассистент*** _____ ***Кравченкова Е.П.*** «____»_____2023г.
должность, степень, звание подпись ФИО

Обучающийся _____ ***Самодуров В.А.*** «__»_____2023г.
подпись ФИО

Задачи

Разработка программного продукта, который

- предоставляет пользователю интерфейс управления сортировкой и статистику;
- демонстрирует наглядно метод сортировки Хоара;

Список рекомендуемой литературы

1 Кнут Д.Э. Искусство программирования, том 3. Сортировка и поиск, 2-е изд. : Пер. с англ. – М.: ООО «И.Д. Вильямс», 2018. – 832 с. : ил. – Парал. тит. англ.
2 Дж. Макконелл Анализ алгоритмов. Активный обучающий подход. — 3-е дополненное издание. М: Техносфера, 2009. -416с.

УТВЕРЖДАЮ
Заведующий кафедрой
д.т.н., профессор
Т.В. Хоменко_____
«____»_____2023г.

К заданию на курсовой проект
по дисциплине
«Алгоритмы и структуры данных»

КАЛЕНДАРНЫЙ ГРАФИК
курсового проектирования

№ п/п	Разделы, темы и их содержание, графический материал	Дата сдачи	Объем, %
1	Выбор темы	04.10.2023	1
2	Техническое задание	14.10.2023	3
3	Разработка модели, проектирование системы ▪ *введение,* ▪ *технический проект,* ▪ *программа и методика испытаний,* ▪ *литература*	30.10.2023	25
4	Программная реализация системы ▪ *работающая программа,* ▪ *рабочий проект* ▪ *скорректированное техническое задание (при необходимости)*	30.10.2023	40
5	Тестирование и отладка системы, эксперименты ▪ *работающая программа с внесенными изменениями,* ▪ *окончательные тексты всех разделов*	11.11.2023	50
6	Компоновка текста Подготовка презентации и доклада ▪ *пояснительная записка* ▪ *презентация* ▪ *электронный носитель с текстом пояснительной записки, исходным кодом проекта, презентацией и готовым программным продуктом*	18.11.2023	59
7	Защита курсового проекта	26.12.2023–29.12.2023	60-100

С графиком ознакомлен «____» _____ 2023г.

Самодуров В.А. _____, обучающийся группы ДИНРб-21
фамилия, инициалы, подпись

График курсового проектирования выполнен
без отклонений / с незначительными отклонениями / со значительными отклонениями
нужное подчеркнуть

Руководитель курсового проекта_____ ассистент Кравченкова Е.П.
подпись, ученая степень, звание, фамилия, инициалы

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире, где информация играет ключевую роль в развитии общества и экономики, ее обработка и систематизация становятся неотъемлемой частью жизни. Сортировка данных является одним из ключевых этапов обработки информации, и от эффективности выбранного метода сортировки зависит скорость и точность выполнения многих задач. В данной курсовой работе будет рассмотрен метод сортировки Хоара и проведено его сравнение с другим методом - сортировкой вставками.

Метод сортировки Хоара является одним из самых быстрых и эффективных алгоритмов сортировки и широко используется в различных областях программирования. Основная идея метода заключается в разделении входных данных на две части, сортировке каждой из них отдельно, а затем объединении этих частей в отсортированный массив.

Однако, для того чтобы определить, насколько эффективен метод сортировки Хоара, необходимо провести его тестирование и сравнение с другими методами. В целях объективного исследования выбран дополнительный метод сортировки - вставками, который, судя по теоретическим материалам, хорошо справляется с малым количеством данных. Сортировка вставками представляет собой последовательное сравнение каждого элемента с уже упорядоченными элементами и занимает $O(n^2)$ времени.

Для проведения исследования необходимо создать удобную и эффективную программу, которая предоставит необходимые инструменты для тщательного анализа сортировки метода Хоара в самой программе и возможность провести ряд тестирований с последующим выводом результатов в файл в удобном для восприятия и анализа виде.

Целью создания исследовательской программы является наглядное и понятное изучение положительных и отрицательных сторон метода сортировки Хоара и его анализ на основе сортировок массивов от 10 до 1 миллиона элементов и сравнения с методом сортировки вставками.

Назначение программы – получение точной и подробной статистики работы метода сортировки Хоара с нужными пользователю условиями и параметрами сортировки.

1 ТЕХНИЧЕСКИЙ ПРОЕКТ

1.1 Анализ предметной области

1.1.1 Метод сортировки Хоара. Понятие, принцип

Метод сортировки Хоара - это алгоритм сортировки, разработанный в 1960 году Чарльзом Э. Р. Хоаром. Является существенно улучшенным вариантом алгоритма сортировки с помощью прямого обмена (его варианты известны как «Пузырьковая сортировка» и «Шейкерная сортировка»), известного в том числе своей низкой эффективностью. Принципиальное отличие состоит в том, что в первую очередь производятся перестановки на наибольшем возможном расстоянии и после каждого прохода элементы делятся на две независимые группы (таким образом улучшение самого неэффективного прямого метода сортировки дало в результате один из наиболее эффективных улучшенных методов).

1.1.2 Алгоритм

Общая идея алгоритма состоит в следующем:

- Выбрать из массива элемент, называемый *опорным*. Это может быть любой из элементов массива. От выбора опорного элемента не зависит корректность алгоритма, но в отдельных случаях может сильно зависеть его эффективность (см. ниже).
- Сравнить все остальные элементы с опорным и переставить их в массиве так, чтобы разбить массив *на три непрерывных отрезка*, следующих друг за другом: «элементы меньшие опорного», «равные» и «большие».
- Для отрезков «меньших» и «больших» значений выполнить *рекурсивно* ту же *последовательность операций*, если длина отрезка больше единицы.

На практике массив обычно делят не на три, а на *две части*: например, «меньшие опорного» и «равные и большие»; такой подход в общем случае эффективнее, так как упрощает алгоритм разделения. На рисунке 1.1 показан пример сортировки (см. на следующей странице).

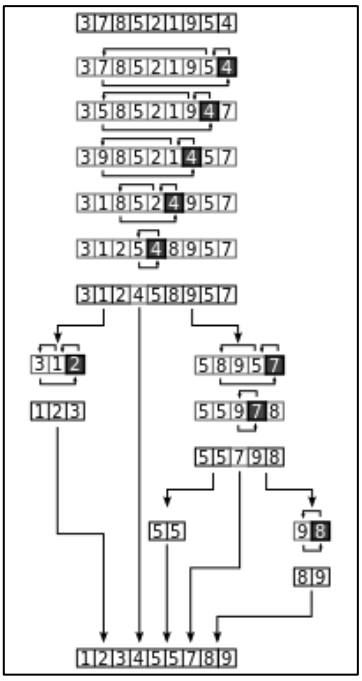

Рисунок 1.3 - Пример быстрой сортировки. Здесь опорным является последний элемент массива (ячейка черного цвета), что в отсортированных массивах может приводить к ухудшению производительности.

1.1.3 Оценка сложности алгоритма

Операция разделения массива на две части относительно опорного элемента занимает время $O(n)$. Поскольку все операции разделения, проделываемые на одной глубине рекурсии, обрабатывают разные части исходного массива, размер которого постоянен, суммарно на каждом уровне рекурсии потребуется также $O(n)$ операций. Следовательно, общая сложность алгоритма определяется лишь количеством разделений, то есть глубиной рекурсии. Глубина рекурсии, в свою очередь, зависит от сочетания входных данных и способа определения опорного элемента.

- Лучший случай.

В наиболее сбалансированном варианте при каждой операции разделения массив делится на две одинаковые (плюс-минус один элемент) части, следовательно, максимальная глубина рекурсии, при которой размеры обрабатываемых подмассивов достигнут 1, составит $\log_2 n$. В результате количество сравнений, совершаемых быстрой сортировкой, было бы равно значению рекурсивного выражения $C_n = 2 * C_{n/2} + n$, что дает общую сложность алгоритма $O(n * \log_2 n)$.

- Средний случай.

Среднюю сложность при случайном распределении входных данных можно оценить лишь вероятностно. Прежде всего необходимо заметить, что в действительности необязательно, чтобы опорный элемент всякий раз делил массив на две *одинаковых* части. Например, если на каждом этапе будет происходить разделение на массивы длиной 75 % и 25 % от исходного, глубина рекурсии будет равна $O(n * \log_{4/3} n)$, а это по-прежнему дает сложность $O(n * \log n)$.

- Худший случай.

В самом несбалансированном варианте каждое разделение дает два подмассива размерами ***1*** и ***n - 1***, то есть при каждом рекурсивном вызове больший массив будет на ***1*** короче, чем в предыдущий раз. При простейшем выборе опорного элемента — первого или последнего в массиве, — такой эффект даст отсортированный (в прямом или обратном порядке) массив, для среднего или любого другого фиксированного элемента «массив худшего случая» также может быть специально подобран. В этом случае потребуется ***n - 1*** операций разделения, а общее время работы составит $O(n^2)$ операций, то есть сортировка будет выполняться за квадратичное время. Для больших значений ***n*** худший случай может привести к исчерпанию памяти (переполнению стека) во время работы программы.

1.1.4 Достоинства и недостатки

Достоинства:

- Один из самых *быстродействующих* (на практике) из алгоритмов внутренней сортировки общего назначения.
- Алгоритм очень *короткий*: запомнив основные моменты, его легко написать «из головы», невелика константа при ***n * log n***.
- С модификациями требует лишь ***O(log n)*** дополнительной памяти в виде стека.
- Хорошо *сочетается* с механизмами кэширования и виртуальной памяти.
- Допускает естественное *распараллеливание* (сортировка выделенных подмассивов в параллельно выполняющихся подпроцессах).
- Допускает эффективную *модификацию* для сортировки по нескольким *ключам* (в частности — алгоритм Седжвика для сортировки строк): благодаря тому, что в процессе разделения автоматически выделяется отрезок элементов, равных опорному, этот отрезок можно сразу же сортировать по следующему ключу.
- Работает на *связных списках* и других структурах с последовательным доступом, допускающих эффективный проход как от начала к концу, так и от конца к началу.

Недостатки:

- *Сильно деградирует* по скорости (до $O(n^2)$) в худшем или близком к нему случае, что может случиться при неудачных входных данных.
- Прямая реализация в виде функции с двумя рекурсивными вызовами может привести к ошибке *переполнения стека*, так как в худшем случае ей может потребоваться сделать $O(n)$ вложенных рекурсивных вызовов.
- *Неустойчив*. Меняет относительный порядок сортируемых элементов, имеющих одинаковые ключи, по которым происходит сортировка.

1.1.5 Исследование метода сортировки Хоара

Исследование метода сортировки Хоара предоставит точную статистику работы метода и понимание работы алгоритма с помощью его визуализации. Для объективной статистики используется метод сортировки вставками в качестве сравнения, что позволит на основе отношения минимизировать субъективность. Данное решение принято также в связи с легкостью понимания недостатков и достоинств, если использовать сравнение, и различием системных комплектующих у пользователей, из-за чего статистика без сравнения станет лишь показателем скорости системных комплектующих.

1.2 Технология обработки данных

Вариант использования «Меню» включает одну из 3 обязательных функций: «О программе», «Туториал», «Начать».

Вариант использования «О программе» и «Туториал» включает одну обязательную функцию, возвращающую в «Меню».

Вариант использования «Начать» включает одну из необязательных 13 функций, где 2 из них меняют параметры массива: «Длина массива», «Тип массива»; 3 – изменяют параметры сортировки: «Сортировка», «Real Time сортировка», «Скорость перестановок»; 3 – отвечают за тестирование: «Результаты», «Тестировать» и функция, меняющая количество тестов; 2 – отвечают за изменение массива: «Сортировать», «Обновить»; и 3 остальные это «Остановить» «Очистить» и функция, возвращающую в «Меню».

1.2.1 Форматы данных

Исследование метода сортировки Хоара представляет собой интерфейс с диаграммой и кнопками управления массивом и сортировкой. На рисунке 1.2 приведен интерфейс программы.

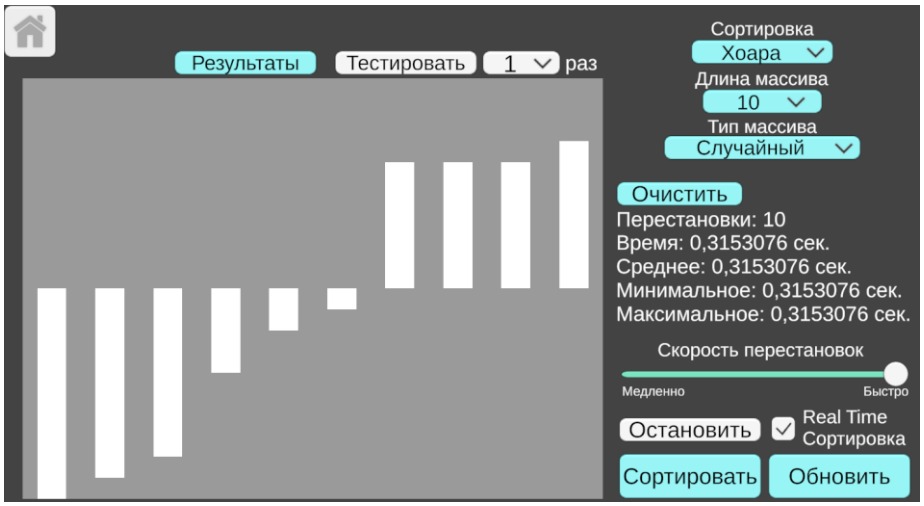

Рисунок 1.2 – Интерфейс исследовательской программы

1.2.2 Алгоритм исследования метода сортировки Хоара

Дано: меню.

Получить: понятное, удобное и точное исследование.

1. Вывод интерфейса с меню из трех кнопок: «О программе», «Туториал», «Начать»;
2. если пользователь выбрал «О программе», вывести интерфейс с информацией о программе и кнопкой, возвращающей в меню, после которого перейти к пункту 1;
3. если пользователь выбрал «Туториал», вывести интерфейс с изображением главного интерфейса программы (рисунок 1.2) и поясняющие подписи к элементам интерфейса и кнопкой, возвращающей в меню, после которого перейти к пункту 1;
4. если пользователь выбрал «Начать», вывести главный интерфейс программы (рисунок 1.2) состоящий из области диаграммы и 10 элементов массива случайного типа на ней, 7 кнопок: «Результаты», «Тестировать», «Очистить», «Остановить», «Сортировать», «Обновить» и кнопка, возвращающая в меню; 4 выпадающих списков, изменяющих типы сортировки и массива, количество тестов и элементов массива; ползунка «Скорость перестановок» и переключателя «Real Time Сортировка». Если пользователь выбрал «Начать» впервые, то кнопки «Обновить», «Остановить» и ползунок «Скорость перестановок» не показаны;
5. если пользователь включил «Real Time Сортировку», то заблокировать кнопку «Тестирование» и выпадающий список, отвечающий за количество тестов, показать кнопку «Остановить» и показать и активировать ползунок «Скорость перестановок», если выключил, то произвести противоположные действия;
6. если пользователь выбрал «Сортировать», то произвести сортировку над массивом выбранного типа и размера в соответствии с типом сортировки и состоянием переключателя «Real Time Сортировка». Если переключатель выключен, то произвести сортировку элементов и затем обновить область диаграммы и статистику, если включен, то деактивировать все кнопки и выпадающие списки, кроме кнопок «Очистить» и возвращающей в меню (при ее нажатии сортировка не прекращается) активировать кнопку «Остановить», произвести сортировку и показывать каждую перестановку элементов на области диаграммы, после сортировки обновить статистику и активировать и деактивировать предыдущие элементы управления противоположно. Если сортировка производится в первый раз, то показать кнопку «Обновить»;
7. если пользователь выбрал «Обновить», то обновить массив в соответствии с размером и типом массива и вывести на область диаграммы;
8. если пользователь выбрал «Остановить», то остановить сортировку;

9. если пользователь выбрал «Очистить», то очистить статистику;
10. если пользователь выбрал «Результаты», то открыть текстовый документ с результатами тестирования, если до этого уже было произведено тестирование;
11. если пользователь выбрал «Тестировать», то произвести тестирование в соответствии с выбранным типом сортировки и количеством тестов. Если проводится тестирование метода Хоара, то производится тестирование с массивом из 10, 10^2, 10^3, 10^4, 10^5 и 10^6 элементов, а если метода вставками – с массивом из 10, 10^2, 10^3 и 10^4 элементов;
12. если пользователь выбрал выпадающие списки, то показать ему список доступных значений для выбора;
13. если пользователь изменяет положение ползунка, то изменять скорость перестановок сортировки от 0 до 1 секунд;
14. если пользователь нажал кнопку возвращения в меню, то перейти к пункту 1.

1.3 Входные и выходные данные

Входные данные:

- нажатие кнопок;
- выбор длины и типа массива;
- выбор типа сортировки;
- выбор количества тестов.

Выходные данные:

- интерфейсы;
- элементы массива;
- результаты сортировки и тестирования

1.4 Системные требования

Рекомендуемая конфигурация:

- Intel-совместимый процессор с частотой не менее 1,6 ГГц;
- не менее 512 МБ ОЗУ;
- не менее 100 МБ свободного места на диске;
- дисковод CD-ROM/DVD-ROM.

Операционная система: Windows 7. Среда разработки – Unity, Microsoft Visual Studio 2022.

2 РАБОЧИЙ ПРОЕКТ

2.1 Общие сведения о работе системы

Программный продукт разработан в кроссплатформенной среде разработки компьютерных игр Unity и интегрированной среде Microsoft Visual Studio 2022 на языке C#. Программа работает под управлением операционной системы Windows 7 (x64) и более поздними.

2.2 Функциональное назначение программного продукта

Разработанный программный продукт предназначен для исследования метода сортировки Хоара и, в связи с решением сравнения, метода сортировки вставками. Программа имеет следующие функциональные возможности:

- предоставление пользователю свободного перемещения по программе;
- предоставление пользователю удобного и достаточного интерфейса для исследования;
- наглядная демонстрация сортировок и их статистика;
- обнуление статистики;
- изменение размера и типа массива, типа сортировки;
- изменение скорости перестановок сортировки;
- прекращение сортировки по желанию пользователя;
- вывод элементов массива на область диаграммы;
- тестирование методов сортировки;
- запись результатов теста в текстовый файл и его открытие;
- прекращение работы программы по желанию пользователя.

Программа имеет следующие функциональное ограничение: ввод осуществляется только нажатием левой кнопки мыши.

2.3 Инсталляция и выполнение программного продукта

Для выполнения программы необходимо:

1. скопировать на жесткий диск компьютера установщик «Hoare Sorting Setup»;
2. запустить установщик;
3. установить программу;
4. запустить программу;

2.4 Описание программы

Программа состоит из 3 файлов на языке «C#». Файл General Settings.cpp (таблица 2.1) предназначен для глобальных параметров, определяющих работу интерфейса программы и 2-х остальных файлов.

Таблица 2.1 – Функции файла General Settings.cpp

Прототип	Назначение
void Start()	Функция Unity. Срабатывает при запуске программы
void UpdateTime()	Обновляет переменную скорости перестановок
void UpdateSize()	Обновляет переменную размера массива
void UpdateType()	Обновляет массив в соответствии с типом
void UpdateGraph()	Обнуляет и обновляет диаграмму
void UpdateAll()	Обновляет размер, тип массива и диаграмму
void EditGraph(int lindex, int rindex)	Меняет местами элементы на диаграмме
void IncreasingRow()	Заполняет массив возрастающими элементами
void DecreasingRow()	Заполняет массив возрастающими элементами
void RandomRow()	Заполняет массив случайными элементами
void StopSorting()	Делает переменную остановки правдой
void CallHoar(t)	Вызывает метод сортировки Хоара
void CallInsertion()	Вызывает метод сортировки вставками
void CallSort()	Вызывает один из методов сортировки
void SetRealTimeSort()	Активирует, деактивирует элементы интерфейса
void UpdateStats()	Обновляет переменные для статистики
void ResetStats()	Обнуляет переменные для статистики
string UnitOfTime()	Возвращает единицу измерения
void EditStatistic()	Обновляет статистику сортировки
void ResetStatistic()	Обнуляет статистику сортировки
void AvailableButton()	Активирует кнопки

Продолжение таблицы 2.1

void DisaiableButton()	Деактивирует кнопки
void ClearStatisticFile()	Обнуляет результат тестирования в файле
void WriteStatisticFile(int i)	Записывает статистику сортировки в файл
void StartTest()	Запускает тестирование
void Results()	Открывает файл с результатами
void UpdateSize(int i)	Обновляет переменную размера массива
void UpdateType(int i)	Обновляет массив в соответствии с типом
void TestHoar()	Тестирование методом Хоара
void TestInsertion()	Тестирование методом вставок
void OpenMe()	Открывает ссылку на GitHub автора

Файл Hoar And Insertion.cpp (таблица 2.2) содержит в себе 2 метода сортировки: метод Хоара и метод вставками. В файле представлены реализации обычной (внутри компьютера) и наглядной (на экране) сортировки.

Таблица 2.2 – Функции файла Hoar And Insertion.cpp

Прототип	Назначение
void EndSorting()	Вызов общих функций после сортировок
void PartOfSortHoaraRealTime(List<int> arr, int left, int right)	Часть метода сортировки Хоара. Отвечает за перестановку элементов и показ перестановок
IEnumerator QuickSortHoaraRealTime(List<int> arr, int start, int end, float time)	Часть метода сортировки Хоара. Отвечает за рекурсивный вызов сортировки для показа перестановок
IEnumerator QuickSortHoaraRealTime(List<int> arr)	Часть метода сортировки Хоара. Отвечает за вызов рекурсивной функции для показа перестановок
void PartOfSortHoara(List<int> arr, int left, int right)	Часть метода сортировки Хоара. Отвечает за перестановку элементов
void QuickSortHoara(List<int> arr, int start, int end, float time)	Часть метода сортировки Хоара. Отвечает за рекурсивный вызов сортировки
void QuickSortHoara(List<int> arr)	Часть метода сортировки Хоара. Отвечает за вызов рекурсивной функции

Продолжение таблицы 2.2

void QuickSortHoaraForTests(List<int> arr)	Часть метода сортировки Хоара. Отвечает за вызов рекурсивной функции для тестирования
IEnumerator InsertionSortRealTime(List<int> arr)	Метод сортировки вставками для показа перестановок
IEnumerator StartInsertionSortRealTime(List<int> arr)	Отвечает за начало сортировки вставками для показа перестановок
void InsertionSort(List<int> arr)	Метод сортировки вставками
void InsertionSortForTests(List<int> arr)	Метод сортировки вставками для тестирования

Файл Graph.cpp (таблица 2.3) предназначен для работы с диаграммой: создание и изменение.

Таблица 2.3 – Функции файла Graph.cpp

Прототип	Назначение
void Start()	Функция Unity. Срабатывает при запуске программы
GameObject CreateCircle(Vector2 anchoredPosition, float xSize)	Возвращает элемент диаграммы в соответствии с элементом массива
void ShowGraph(List<int> valueList, int count)	Создает диаграмму на основе массива
void EditGraph(int lindex, int rindex)	Меняет элементы диаграммы местами

2.5 Разработанные меню и интерфейсы

После запуска программы появится ее меню с указанием названия, автора и его GitHub. Среди пунктов меню представлен выбор из трех различных вариантов: информация о программе, инструктаж по интерфейсу исследования и сам интерфейс исследования. Программа открывается в оконном режиме, поэтому выход осуществляется при нажатии соответствующей кнопки. На рисунке 2.1 представлено меню программы.

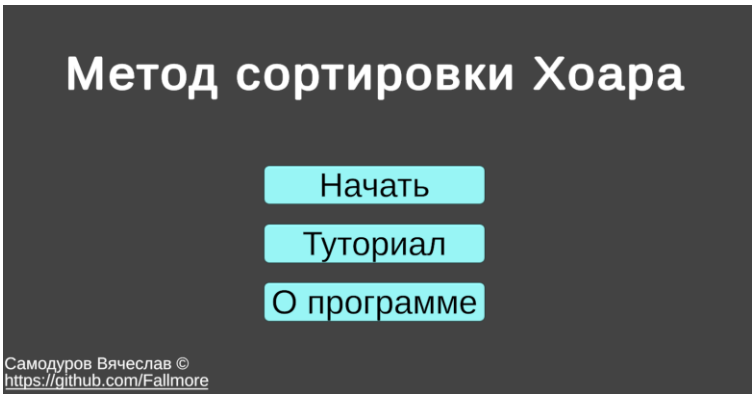

Рисунок 2.1 – Меню

При нажатии на «О программе» пользователю выведется информация о программе. (рисунок 2.2), при нажатии на «Туториал» – обучающий интерфейс (рисунок 2.3), при нажатии на «Начать» – интерфейс исследования (рисунок 2.4).

Рисунок 2.2 - Информация о программе

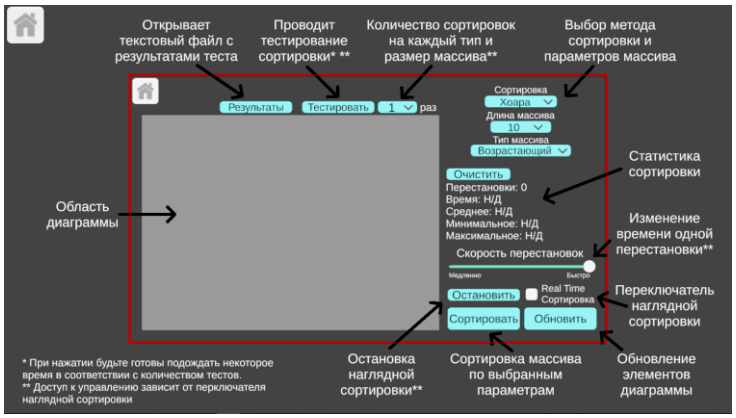

Рисунок 2.3 - Обучающий интерфейс

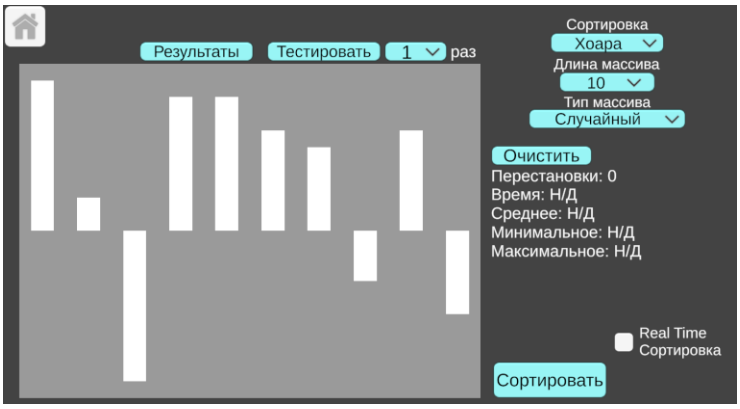

Рисунок 2.4 - Интерфейс исследования

На рисунке 2.5 представлен результат первой сортировки массива, изображенного на рисунке 2.4. На рисунке 2.6 представлена диаграмма массива из 1000 случайных элементов во время сортировки методом Хоара с включенным переключателем наглядной сортировки.

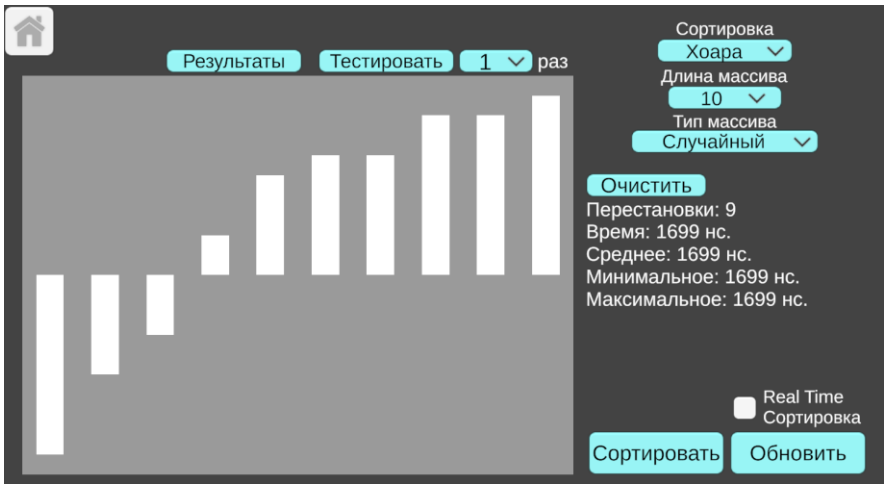

Рисунок 2.5 – Результат первой сортировки массива

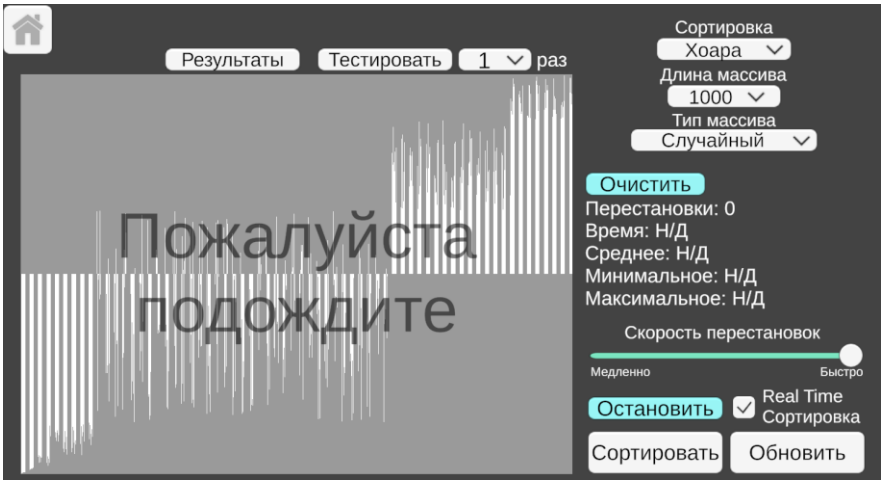

Рисунок 2.6 – Диаграмма массива из 1000 случайных элементов во время сортировки методом Хоара

На рисунке 2.7 представлена диаграмма массива из 100 убывающих элементов во время сортировки методом вставками с включенным переключателем наглядной сортировки. На рисунке 2.8 представлен результат тестирования из 10 тестов методом Хоара.

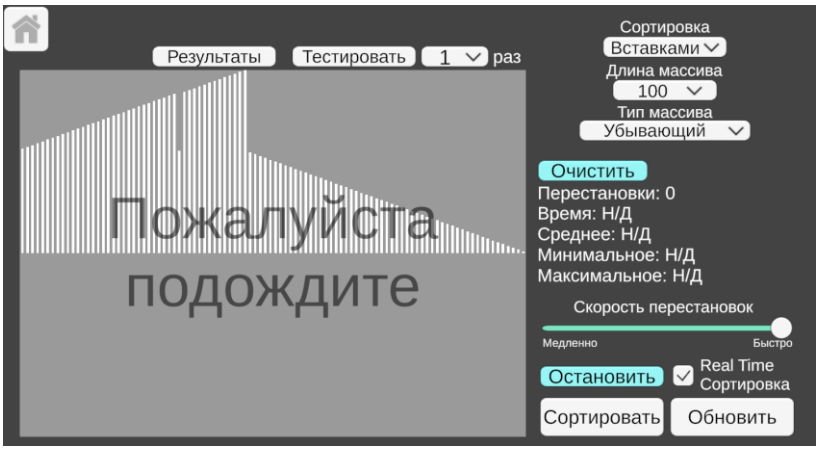

Рисунок 2.7 – Диаграмма массива из 1000 случайных элементов во время сортировки методом вставками

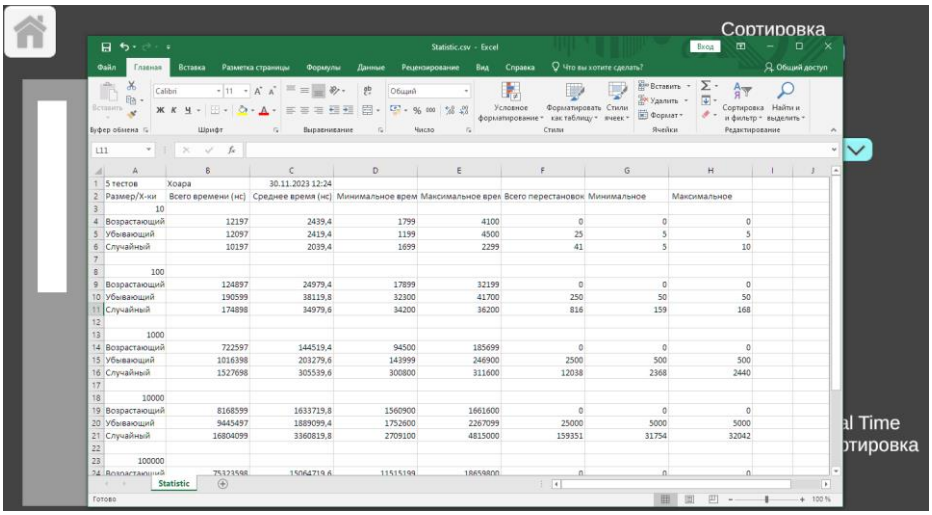

Рисунок 2.8 – Результат тестирования из 10 тестов методом Хоара

2.6 Сообщение системы

В данной программе отсутствуют сообщения системы. В случае появления сообщений следует обратиться к разработчику.

3 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

3.1 Постановка задачи

Осуществить исследование методов сортировки:

- метод Хоара;
- метод вставками.

Исследование осуществить, используя массивы упорядоченных и неупорядоченных чисел по 10, 10^2, 10^3 и 10^4 элементов для метода вставками и по 10, 10^2, 10^3, 10^4, 10^5 и 10^6 элементов для метода Хоара, так как в теории данный метод быстрее.

3.2 Краткая теория

При обработке данных важно знать информационное поле данных и размещение их в машине. Различают внутреннюю и внешнюю сортировки:

- внутренняя сортировка - сортировка в оперативной памяти;
- внешняя сортировка - сортировка во внешней памяти.

Сортировка – это расположение данных в памяти в регулярном виде по их ключам. Регулярность – возрастание (убывание) значения ключа от начала к концу в массиве.

Если сортируемые записи занимают большой объем памяти, то их перемещение требует больших затрат. Для того, чтобы их уменьшить, сортировку производят в таблице адресов ключей, делают перестановку указателей, т.е. сам массив не перемещается. Это метод сортировки таблицы адресов.

При сортировке могут встретиться одинаковые ключи. В этом случае после сортировки желательно расположить одинаковые ключи в том же порядке, что и в исходном файле. Это устойчивая сортировка.

Эффективность сортировки можно рассматривать с нескольких критериев:

- время, затрачиваемое на сортировку;
- объем оперативной памяти, требуемой для сортировки;
- время, затраченное программистом на написание программы.

Выделяем первый критерий. Можно подсчитать количество перестановок при выполнении сортировки и его время, вместо количества сравнений.

3.2.1 Метод сортировки Хоара

Программа на C#

```
int PartOfSortHoara(List<int> arr, int left, int right)
   {
      int pivot = arr[(left + right) / 2];    // Находим опорный элемент в середине
      while (left <= right)
      {
         while (arr[left] < pivot) ++left;    // Слева ищем элемент больше опорного
         while (arr[right] > pivot) --right; // Справа ищем элемент меньше опорного

         if (left <= right)
         {
            (arr[right], arr[left]) = (arr[left], arr[right]); // Меняем местами элементы
            ++left;
            --right; // Идем дальше
         }
      }
      return left; // Возвращаем левую границу правого пполучившегося подмассива
   }
   void QuickSortHoara(List<int> arr, int start, int end)
   {
      if (start >= end) return;

      int rightStart = PartOfSortHoara(arr, start, end); // Сортировка и получение номера границы
      QuickSortHoara(arr, start, rightStart - 1); // Рекурсия левого подмассива
      QuickSortHoara(arr, rightStart, end); // Рекурсия правого подмассива
   }
   void QuickSortHoara(List<int> arr)
   {
      QuickSortHoaraForTests(arr); // Вызов сортировки
   }
```

3.2.2 Метод сортировки вставками

Программа на C#

```
void InsertionSort(List<int> arr)
   {
      for (int i = 1; i < arr.Count; ++i)
      {
         int k = arr[i]; // Сохраняем значение элемента
         int j = i - 1;  // Сохраняем предыдущий индекс
         while (j >= 0 && arr[j] > k) // Пока индекс больше нуля и элемент больше сохраненного
         {
            arr[j + 1] = arr[j]; // \
            arr[j] = k;            //  Меняем местами элементы
            --j; // Идем назад, по отсортированным элементам
         }
      }
   }
```

3.3 Метод исследования

Исследование заключается в следующем: берется три массива с одинаковым количеством элементов, но один из них упорядоченный по возрастанию, второй - по убыванию, а третий - случайный. Осуществляется сортировка данных массивов 100 раз и сравнивается количество перестановок элементов при сортировке первого, второго и третьего массивов, а также сравнивается затраченное время при сортировке.

Вышеописанный способ применяется для массивов, состоящих из упорядоченных и неупорядоченных элементов в количестве 10, 10^2, 10^3 и 10^4 для метода вставками и 10, 10^2, 10^3, 10^4, 10^5 и 10^6 для метода Хоара.

3.4 Результаты исследования

Для удобства восприятия наносекунды в некоторых случаях (большие числа) округлены в секунды.

Количество наносекунд в секундах:

- 100.000 – 0.0001 сек.
- 1.000.000 – 0.001 сек.
- 10.000.000 – 0.01 сек.
- 100.000.000 – 0.1 сек.

Сокращения:

- метод Хоара – Х;
- метод вставками – В;
- наносекунды – нс;
- минимальное – мин;
- максимальное - макс.

В результате сортировки массивов с возрастающими элементами (таблица 3.1) общее количество перестановок равно нулю (что естественно). Время у сортировки Хоара, начиная с 10 элементов, возрастало с каждым тестом в ***чуть больше 10 раз***, когда время у сортировки вставками гораздо меньше и, начиная с тех же 10 элементов, время с каждым тестом возрастало приблизительно в ***9 раз***. Минимальное и максимальное время в 10 и 100 элементах метод вставками лучше в несколько раз[*], в последующих тестах минимальное время метода вставок лучше в ***10 раз***, максимальное время в 1.000 элементах лучше приблизительно в ***100 раз***, а в 10.000 элементах – приблизительно в ***15 раз***. По предварительным данным можно спрогнозировать, что метод вставками в последующих тестах уходил бы в отрыв от метода Хоара всё дальше и дальше, сильно опережая.

Таблица 3.1 – Массив с возрастающими элементами

Количество элементов	**Метод**	**Количество перестановок**			**Время (нс – 10^{-9} сек)**		
		Всего	**Мин.**	**Макс.**	**Всего**	**Мин.**	**Макс.**
10	X	0	0	0	61.055	399	2.799
	B	0	0	0	18.736	99	5.699
100	X	0	0	0	846.344	6.099	11.900
	B	0	0	0	95.944	799	4.200
1.000	X	0	0	0	0,01 сек.	54.000	1.170.100
	B	0	0	0	648.861	5.199	14.200
10.000	X	0	0	0	0,08 сек.	685.599	1.483.300
	B	0	0	0	0,005 сек.	51.999	83.199
100.000	X	0	0	0	0,84 сек.	7.201.299	0,01 сек.
1.000.000	X	0	0	0	9,4 сек.	0,08 сек.	0,11 сек.

[*] Во времени максимума в 10 элементах наблюдается, что метод Хоара лучше метода вставками. Скорее всего, это погрешность программы, которая обычно происходит при начале каждого теста каждого размера и типа массива, что очень напоминает на «разогрев» методов в начале каждой серии тестов.

Результат сортировки массивов с убывающими элементами (таблица 3.2) показывает, что количество перестановок (смотреть на минимум и максимум) метода Хоара растёт по формуле – ***n / 2***, где ***n*** – количество элементов массива, что делает его лучше метода вставок, когда у метода вставок количество с каждым тестом возрастает приблизительно в ***100 раз***. По времени метод Хоара везде показывает своё превосходство и, начиная с 1.000 элементов, становится в ***100 раз*** быстрее метода вставками. По предварительным данным можно спрогнозировать, что метод вставками в последующих тестах уходил бы в отрыв от метода Хоара всё дальше и дальше, очень сильно отставая.

Таблица 3.2 – Массив с убывающими элементами

Количество элементов	**Метод**	**Количество перестановок**			**Время (нс – 10^{-9} сек)**		
		Всего	**Мин.**	**Макс.**	**Всего**	**Мин.**	**Макс.**
10	X	500	5	5	67.058	499	2.900
	B	4500	45	45	104.737	799	15.599
100	X	5000	50	50	1.535.648	8.099	301.300
	B	495.000	4950	4950	8.932.850	60.700	271.299
1.000	X	50000	500	500	7.936.050	59.699	173.899
	B	$4.995*10^4$	499.500	499.500	0,75 сек.	6.546.000	0,01 сек.
10.000	X	$5*10^5$	5000	5000	0,08 сек.	688.699	0,01 сек.
	B	$49995*10^5$	49995000	49995000	7.6 сек.	0,72 сек.	0,87 сек.
100.000	X	$5*10^6$	50000	50000	0,92 сек.	7.798.500	0,03 сек.
1.000.000	X	$5*10^7$	$5*10^5$	$5*10^5$	9,9 сек.	0,08 сек.	0,11 сек.

Результат сортировки массивов со случайными элементами (таблица 3.3) показывает, что по количеству сортировок метод Хоара в 10 элементах не сильно опережает метод вставками, но в последующих тестах он опережает метод вставками приблизительно в $10^{n/100}$ раз, где n – количество элементов массива. Можно заметить что общее количество сортировок метода вставками в 10.000 элементах превышает число 2^{31}. По времени метод Хоара, несмотря на количество перестановок, в 10 элементах проигрывает методу вставок, а в последующих тестах опережает, с каждым разом увеличивая отрыв. По предварительным данным можно спрогнозировать, что метод вставками в последующих тестах уходил бы в отрыв от метода Хоара всё дальше и дальше, очень сильно отставая.

Таблица 3.3 – Массив со случайными элементами

Количество элементов	**Метод**	**Количество перестановок**			**Время (нс – 10^{-9} сек)**		
		Всего	**Мин.**	**Макс.**	**Всего**	**Мин.**	**Макс.**
10	X	859	5	12	114.959	699	4.599
	B	2195	10	35	99.445	499	4.200
100	X	16.450	149	180	1.773.046	11.299	73.500
	B	246.237	2.033	2.767	5.970.847	28.799	171.599
1.000	X	242.140	2.342	2.492	0,02 сек.	124.000	367.700
	B	25.040.510	236.140	266.061	0,41 сек.	3.221.299	0,01 сек.
10.000	X	3.183.612	31.255	32.381	0,18 сек.	1.537.300	3.064.200
	B	2.499.054.291	24525076	25479020	37,7 сек.	0,34 сек.	0,45 сек.
100.000	X	39.525.633	389.833	399.745	2,2 сек.	0,02 сек.	0,04 сек.
1.000.000	X	471.820.622	4.665.647	4.761.928	26,9 сек.	0,2 сек.	0,5 сек.

3.5 Итог

По результатам исследования видно, что *в возрастающем* массиве очень хорошо показывает себя ***метод вставками***: чем больше элементов, тем быстрее метод; *в убывающем* массиве – ***метод Хоара***: выигрывает по времени и количеству перестановок; *в случайном* массиве – ***метод Хоара***: в массиве из 10 элементов он незначительно проигрывает методу вставками, но дальше с каждым разом имеет настолько больший отрыв, что сортировка 1 млн. элементов быстрее сортировки 10 тыс. элементов методом вставками. Данное преимущество обусловлено очень малым количеством перестановок, по сравнению с методом вставками. Из чего можно сделать вывод, что метод Хоара является самой быстрой сортировкой в мире на данный момент времени, учитывая теоретические данные о других методах сортировки.

ЗАКЛЮЧЕНИЕ

В результате курсового проектирования разработана программа для исследования метода сортировки Хоара. Программа позволяет провести исследование и получить точную, понятную и удобную статистику работы сортировки. Исследование показало, что метод сортировки Хоара проигрывает методу сортировки вставками в массиве из возрастающих элементов, но выигрывает в массивах с остальными элементами, убывающими и случайными. В реальной жизни массив из возрастающих элементов не имеет смысл сортировать, поэтому на практике метод сортировки Хоара является самой быстрой на данной момент сортировкой в мире, учитывая теоретические данные о других методах сортировки.

Разработанная программа помогает в исследовании данного метода. Инструменты и тестирование предоставляют все необходимые функции в изучении сортировки, составлении итогов и анализе результатов.

Исследовательская программа отвечает поставленным требованиям и может быть использована для исследований.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Unity User Manual 2022.3 (LTS) – [Электронный ресурс] режим доступа: https://docs.unity3d.com/Manual/ (02.09.2023).
2. Мамлеева А.Р., Евсина Е.М., Кравченкова Е.П. Требования к оформлению пояснительной записки при выполнении отчетов по различным видам практик. – Астрахань.: Астраханский государственный технический университет, 2022. 50 с.
3. Дасгупта С. и др. Алгоритмы / С. Дасгупта, Х. Пападимитриу, У. Вазирани; Пер. с англ. под ред. А. Шеня. — М.: МЦНМО, 2014. 320 с.
4. Быстрая сортировка. Википедия – [Электронный ресурс] режим доступа: https://ru.wikipedia.org/wiki/Быстрая_сортировка#Общий_механизм_сортировки (25.11.2023).
5. Устойчивая сортировка. Википедия – [Электронный ресурс] режим доступа: https://ru.wikipedia.org/wiki/Устойчивая_сортировка (25.11.2023).
6. Алгоритм сортировки. Википедия – [Электронный ресурс] режим доступа: https://ru.wikipedia.org/wiki/Алгоритм_сортировки (25.11.2023).
7. Описание алгоритмов сортировки и сравнение их производительности. Хабр – [Электронный ресурс] режим доступа: https://habr.com/ru/articles/335920/ (23.11.2023).
8. Алгоритмы сортировки. Bimlibik – [Электронный ресурс] режим доступа: https://bimlibik.github.io/posts/sorting-algorithm/ (23.11.2023).

ПРИЛОЖЕНИЕ 1

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

на выполнение курсового проекта

по дисциплине «Алгоритмы и структуры данных»

Направление 090301 – Информатика и вычислительная техника

Исполнитель: обучающийся гр. ДИНРб-21 **Самодуров В.А.**

Тема: **Исследование метода сортировки Хоара**

1 Назначение, цели и задачи разработки

Цель разработки – наглядное и понятное изучение положительных и отрицательных сторон метода сортировки Хоара.

Назначение разработки:

- получение точной и подробной статистики работы метода сортировки Хоара.

Основные задачи, решаемые разработчиком в процессе выполнения курсового проекта:

- анализ предметной области;
- разработка программного продукта в соответствии с требованиями;
- документирование проекта в соответствии с установленными требованиями.

2 Характер разработки: прикладная квалификационная работа.

3 Основания для разработки

- Учебный план направления 09.03.01 «Информатика и вычислительная техника» 2022 года набора.
- Рабочая программа дисциплины «Информатика и вычислительная техника».
- Распоряжение по кафедре АСОИУ №____ от «___»_____ 2022 г.

4 Плановые сроки выполнения – осенний семестр 2023 учебного года:

Начало «04» октября 2023 г.

Окончание «12» декабря 2023 г.

5 Требования к проектируемой системе

5.1 Требования к функциональным характеристикам

Проектируемая система должна обеспечивать выполнение следующих основных функций:

- предоставлять пользователю визуализацию сортировки;
- предоставлять инструменты для исследования;
- прекращать визуализацию по желанию пользователя.

Система должна предусматривать функциональные ограничения:

- входные данные осуществляются только через левую кнопку мыши;
- программа не может использоваться в качестве сортирующей программы.

5.3 Требования к эксплуатационным характеристикам

Программа не должна аварийно завершаться при любых действиях пользователя.

Время реакции программы на действия пользователя не должно превышать 10 секунд и на тестирование - 10 минут.

5.4 Требования к программному обеспечению:

Среда разработки – Unity, Microsoft Visual Studio (C#) (Версия 17.4.1).

Операционная система: Windows 10 (x64), версия 21H2, сборка ОС 19044.2486.

5.5 Требования к аппаратному обеспечению:

Рекомендуемая конфигурация:

- Intel-совместимый процессор с частотой не менее 1,6 ГГц;
- не менее 512 МБ ОЗУ;
- не менее 100 МБ свободного места на диске;
- Дисковод CD-ROM/DVD-ROM.

6 Стадии и этапы разработки

6.1 Эскизный проект (ЭП)

- Анализ предметной области.
- Подготовка проектной документации.

6.2 Технический проект (ТП)

- Разработка структур и форм представления данных.
- Разработка структуры программного комплекса.
- Подготовка пояснительной записки.

6.3 Рабочий проект (РП)

- Программная реализация.
- Тестирование и отладка программы.
- Подготовка программной и эксплуатационной документации.

6.4 Эксплуатация (Э)

Описание и анализ результатов проведенного исследования.

7 Требования к документированию проекта

К защите курсового проекта должны быть представлены следующие документы:

- Пояснительная записка к курсовому проекту.
- Презентация доклада.

- Программа, презентация и пояснительная записка к курсовому проекту на оптическом носителе.

Требования к структуре документов определены соответствующими стандартами ЕСПД.

Требования к оформлению определены соответствующими методическими указаниями.

8 Порядок контроля и приемки

Контроль выполнения курсового проекта проводится руководителем поэтапно в соответствии с утвержденным графиком проведения практики.

На завершающем этапе руководитель осуществляет нормоконтроль представленной исполнителем документации и принимает решение о допуске (недопуске) проекта к защите.

Защита курсового проекта проводится комиссией в составе не менее двух человек, включая руководителя практики.

В процессе защиты проекта исполнитель представляет документацию, делает краткое сообщение по теме разработки и демонстрирует ее программную реализацию.

При выставлении оценки учитывается:

- степень соответствия представленной разработки требованиям технического задания;
- качество программной реализации, документации и доклада по теме проекта;
- соблюдение исполнителем графика выполнения курсового проекта.

9 Литература

1. Environment.UserName Свойство. Microsoft Learn – [Электронный ресурс] режим доступа: https://learn.microsoft.com/ru-ru/dotnet/api/system.environment.username?view=net-7.0 (20.11.2023)
2. Как создать текстовый файл? – C#. Киберфорум – [Электронный ресурс] режим доступа: https://www.cyberforum.ru/csharp-beginners/thread111134.html (19.11.2023)
3. Как сделать график в unity. Youtube – [Электронный ресурс] режим доступа: https://youtu.be/nTKivkiYbK8?si=XaLnXQS4BFlmtN9A (19.11.2023)
4. Coroutines. Unity – [Электронный ресурс] режим доступа: https://docs.unity3d.com/Manual/Coroutines.html (18.11.2023)
5. How do I launch files in C#. StackOverflow – [Электронный ресурс] режим доступа: https://stackoverflow.com/questions/1283584/how-do-i-launch-files-in-c-sharp (19.11.2023)