МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Челябинский государственный университет»**

**(ФГБОУ ВО «ЧелГУ»)**

Институт информационных технологий

Кафедра информационных технологий и экономической информатики

ОТЧЕТ

по лабораторной работе №2

Выполнил:

Студент группы ПрИ-202 Приходько Даниил Александрович

Студент группы ПрИ-202 Саламатин Алексей Юрьевич

Студент группы ПрИ-202 Скоробогатов Максим Дмитриевич

Принял:

Преподаватель ИИТ Николаев Иван Евгеньевич

Отчет защищен \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата оценка

Челябинск 2024 г.

Содержание

Введение

Цель работы: понять и доступно объяснить Шестикласснику, как работают различные сортировки данных. Написать приложение с визуальной частью, благодаря которой будет видно весь процесс сортировок.

Задание 1: Реализовать демонстрацию работы 4 алгоритмов внутренней сортировки. А конкретно, нами были выбраны: Select Sort, Shell Sort, Quick Sort, Merge Sort.

Задание 2: Дана таблица, состоящая из нескольких полей. Таблица находится в файле. Реализовать программу, которая одним из методов (прямым, естественным или многопутевым слиянием) сортирует записи и записывает их в результирующий файл. Метод внешней сортировки и ключевой атрибут, по которому осуществляется сортировка выбирается пользователем. Количество атрибутов в таблица может быть произвольным.

Задание 3: Пусть имеется текст, состоящий из слов. Необходимо разбить текст на отдельные слова и провести их сортировку в лексикографическом порядке. Причем одинаковые слова будут в отсортированной последовательности идти друг за другом. Надо выбрать два алгоритма сортировки, один из которых базовый или усовершенствованный, и другой алгоритм ABC- сортировка или redix сортировка. После получения отсортированного массива необходимо пройтись по нему и для каждого слова подсчитать, сколько раз оно встречается. Результаты подсчетов вывести на экран. Далее необходимо провести эксперименты с использованием обоих алгоритмов по сортировке текстов различной длины (100, 500, 1000, 2000, 5000 слов или более). Причем для каждого эксперимента произвести замеры времени сортировки. Получившиеся

данные оформить в таблицу.

Задание 1

В рамках этой части были реализованы алгоритмы внутренней сортировки Select Sort, Shell Sort, Quick Sort, Merge Sort.

Сортировка выбором – это алгоритм, суть которого заключается в постоянном сравнении элементов неотсортированой части. Наименьший элемент в неотсортированой части меняется местами с первым ее элементом и сразу становится отсортированой частью.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 1.1.1 – 1.1.3.

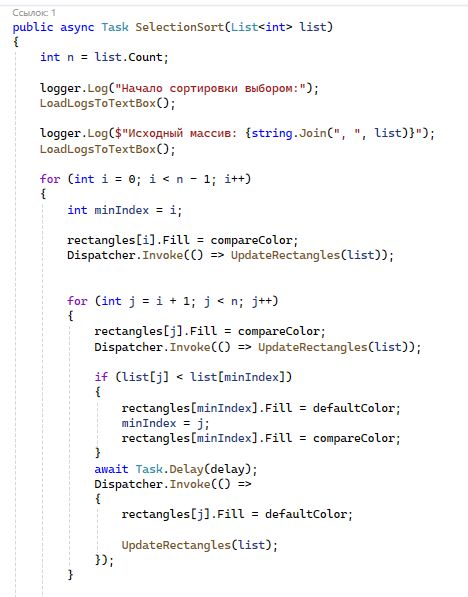


Рис.1.1.1 Код сортировки выбором

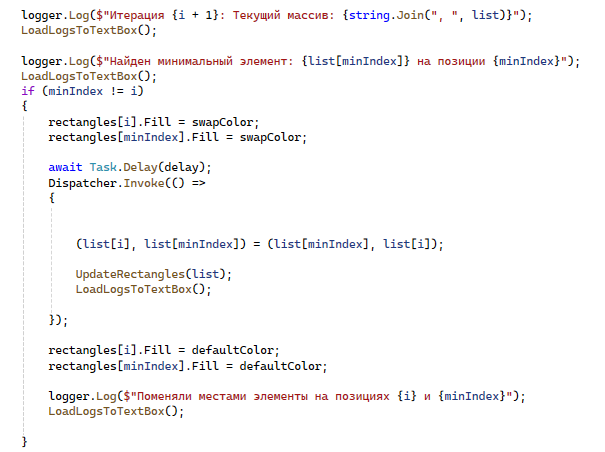


Рис.1.1.2 Код сортировки выбором

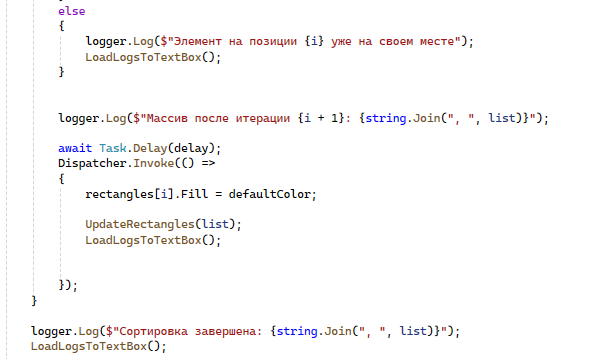


Рис.1.1.3 Код сортировки выбором

Сортировка Шелла – это модифицированая сортировка вставками. Ее суть в том, что массив данных разбивается на так называемые Gap’ы (зазоры). Каждая итерация сортировки – это сравнение элементов массива на расстоянии текущего зазора друг от друга, а затем упорядочивание этих элементов. С каждым проходом по массиву зазор уменьшается вдвое и эти действия повторяются до окончания сортировки.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 1.2.1 – 1.2.2.

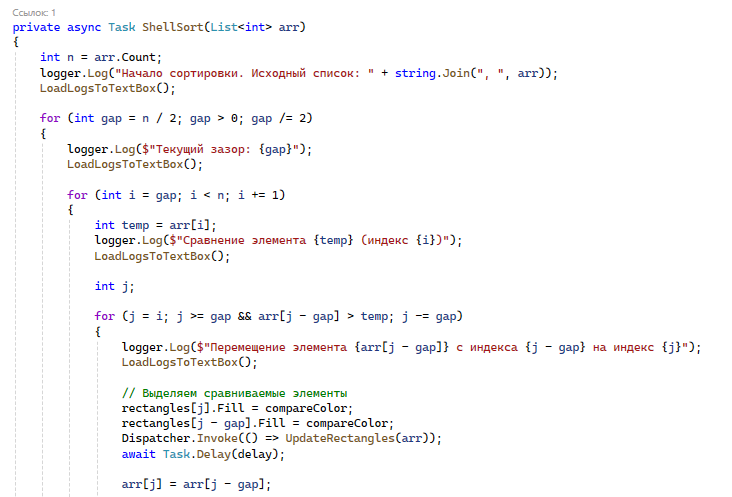


Рис.1.2.1 Код сортировки Шелла

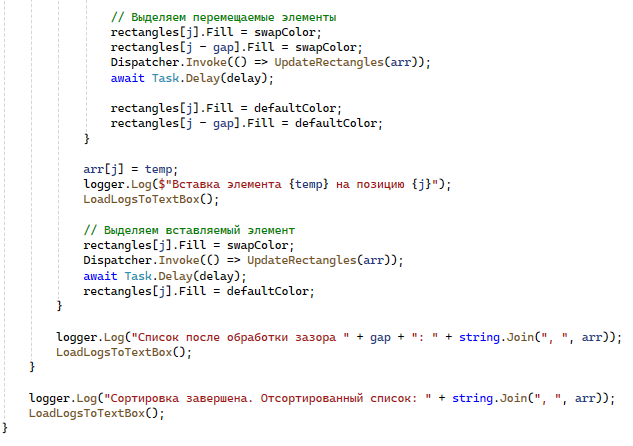


Рис.1.2.2 Код сортировки Шелла

Быстрая сортировка - это алгоритм "разделяй и властвуй", который рекурсивно сортирует массив, разделяя его на части относительно опорного элемента и объединяя отсортированные части.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 1.3.1 – 1.3.3.

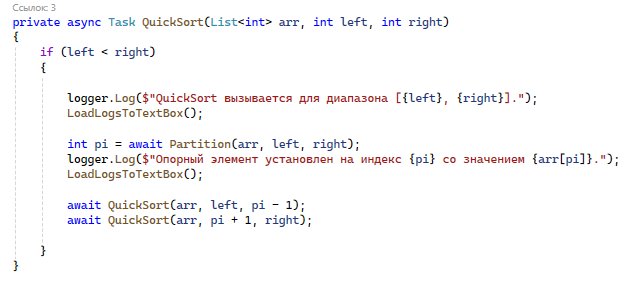


Рис.1.3.1 Код быстрой сортировки

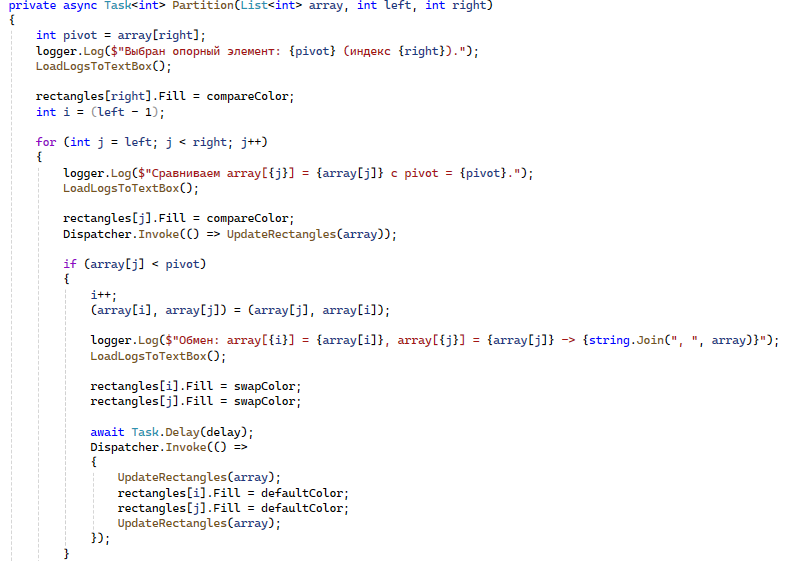


Рис.1.3.2 Код быстрой сортировки

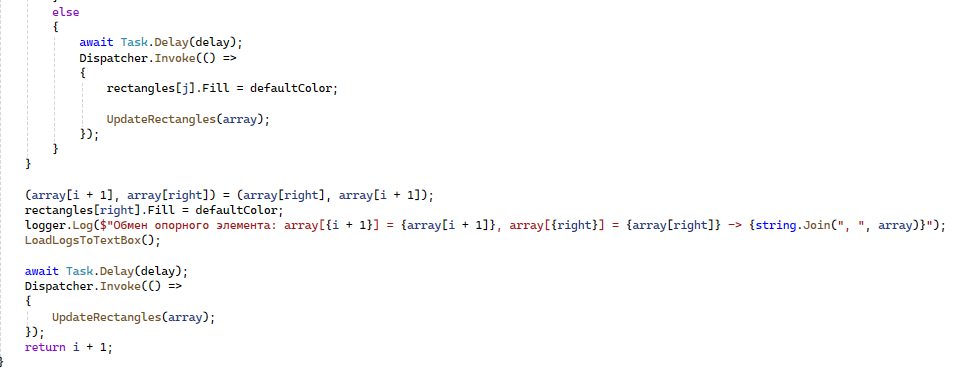


Рис.1.3.3 Код быстрой сортировки

Сортировка слиянием – это алгоритм который разделяет массив данных на две равные части, потом вызывает сам себя для каждой из этих частей и так до тех пор, пока массивы не станут единичными. Дальше эти подмассивы сравниваются со своей «половинкой» и в упорядоченом виде объединяются в новый отсортированый массив.

Данный алгоритм является внутренним, несмотря на то что есть его аналоги в виде внешних сортировок.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 1.4.1 – 1.4.4.

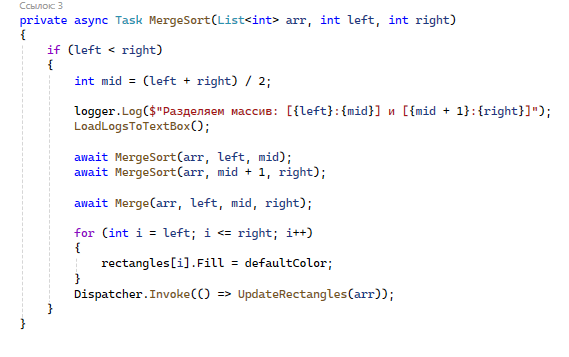


Рис.1.4.1 Код сортировки слиянием

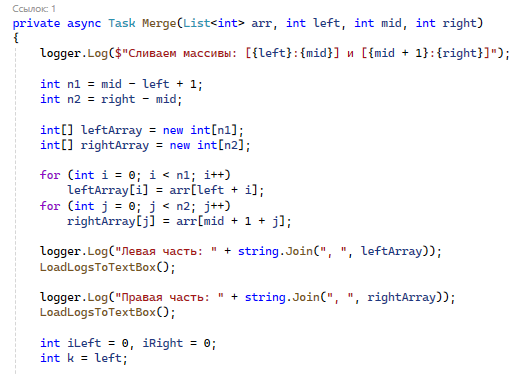


Рис.1.4.2 Код сортировки слиянием

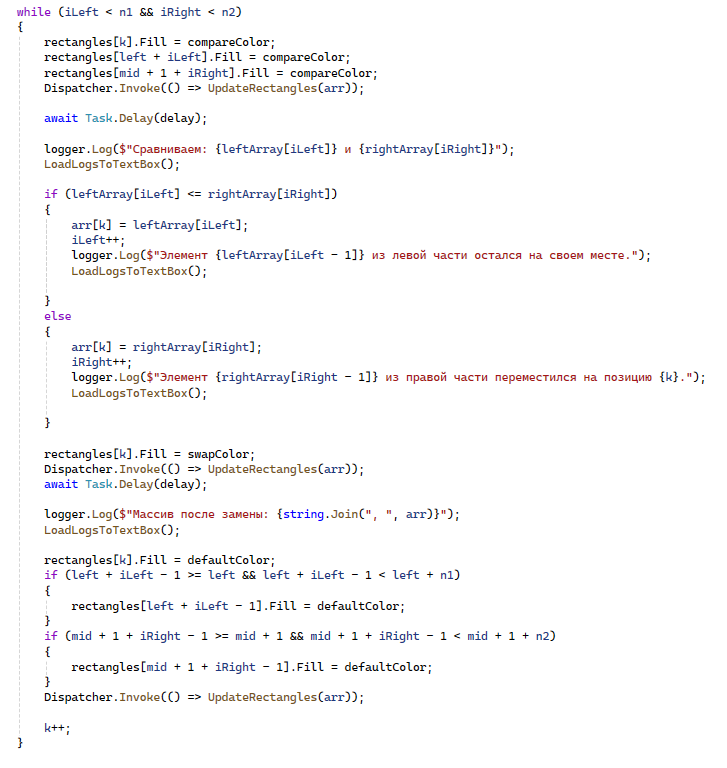


Рис.1.4.3 Код сортировки слиянием

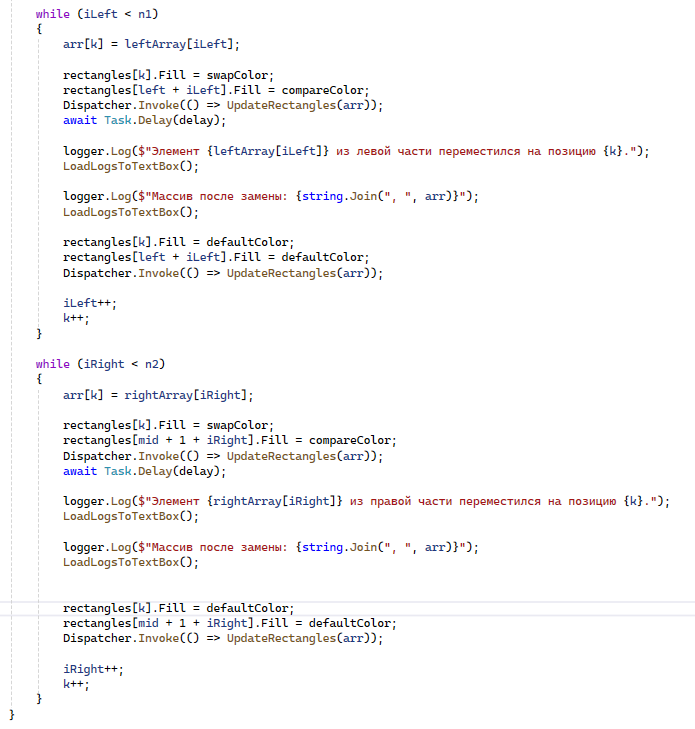


Рис.1.4.4 Код сортировки слиянием

Чтобы наглядно посмотреть на процесс сортировки был написан GUI для данной части (рис.1.5), где видно как проходит сортировка элементов, которые представлены в виде столбцов с соответствующей величиной. Так же есть окно, куда с определенной задержкой, которую выбирает пользователь, выводятся сопутствующие объяснения. Сам массив пользователь может ввести в верхнем окне ввода, через запятую или пробел. Выбрать желаемую сортировку можно в панели управления, в правой части окна.

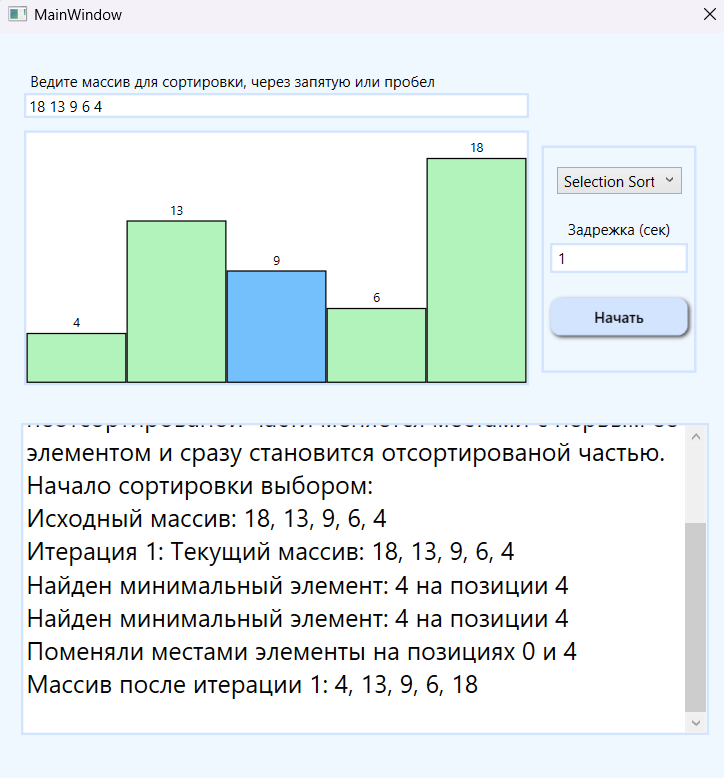


Рис.1.5 визуализация первого задания

Задание 2

Эта часть работы заключается в том, чтобы показать наглядно, и объяснить, как работают внешние сортировки, такие как прямое, естественное и многопутевое слияния.

Прямое слияние – большой массив данных разбивается на части (чанки), которые могут поместиться в оперативную память. Дальше к этим частям применяются алгоритм внутренней сортировки, в нашем случае – сортировка вставками. Отсортированные чанки сливаются в готовый массив.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 2.1.1 – 2.1.3.

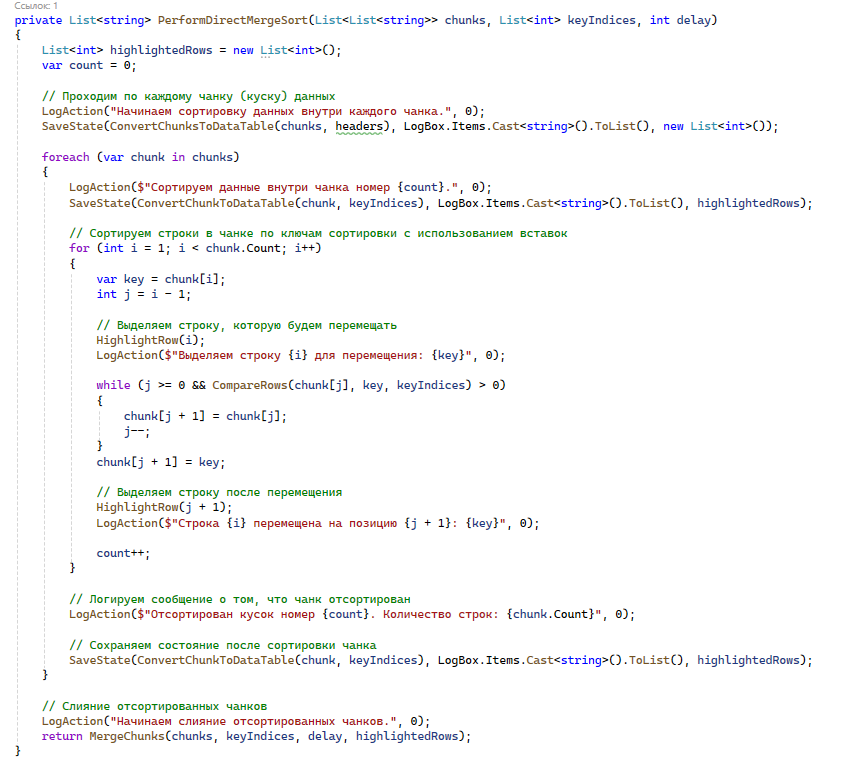


Рис.2.1.1 Код сортировки прямым слиянием



Рис.2.1.2 Код сортировки прямым слиянием



Рис.2.1.3 Код сортировки прямым слиянием

Естественное слияние

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 2.2.1 – 2.2.4.

  
Рис.2.2.1 Код сортировки естественным слиянием

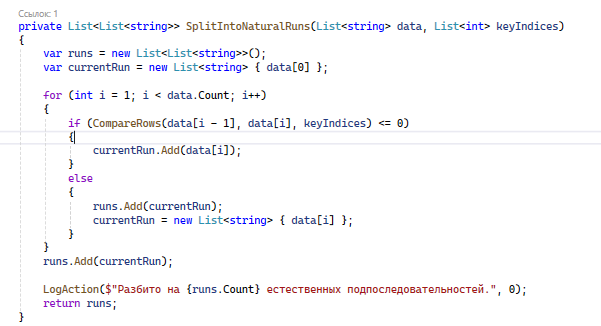


Рис.2.2.2 Код сортировки естественным слиянием



Рис.2.2.3 Код сортировки естественным слиянием



Рис.2.2.4 Код сортировки естественным слиянием

Многопутевое слияние – это метод внешней сортировки, при котором данные разбиваются на несколько файлов (так называемые «пути»), сортируются отдельно (в нашем случае сортировкой вставками), а затем объединяются в один отсортированный файл через многократное слияние

При использовании метода многопутевой внешней сортировки на каждом шаге примерно половина вспомогательных файлов используется для ввода данных и примерно столько же для вывода сливаемых серий.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 2.3.1 – 2.3.2.



Рис.2.3.1 Код сортировки многопутевым слиянием



Рис.2.3.2 Код сортировки многопутевым слиянием

Чтобы наглядно посмотреть на процесс сортировки был написан GUI для данной части (рис.2.4), где видно как проходит сортировка элементов таблицы. Пользователю представлен выбор сортировки, ключа. Так же он может пошагово отслеживать изменения во время процесса, благодаря кнопкам «Вперед» и «Назад», или включить автопроигрывание.

В отдельных окнах выводится визуализация сортировки в виде состояния таблицы в текущий момент времени, текстовая документация процесса и финальный результат.

КАРТИНКА

Задание 3

В рамках этой части были реализованы алгоритмы сортировки Modificated Buble Sort и Radix Sort.

Modificated Bubble Sort — это оптимизированная версия стандартного алгоритма сортировки пузырьком, которая позволяет завершить сортировку раньше, если массив уже отсортирован. Этот алгоритм прост в реализации и может быть эффективен для небольших массивов или частично отсортированных данных.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 3.1.1 – 3.1.2.



Рис.3.1.1 Код сортировки пузырьком.

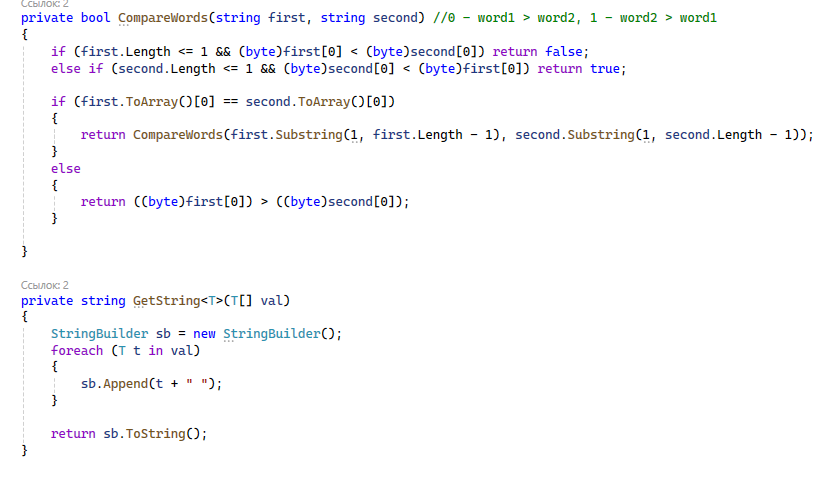


Рис.3.1.2 Код сортировки пузырьком.

Radix Sort - выполняет поразрядную сортировку, начиная с последнего символа и двигаясь к первому. Все строки выравниваются до одинаковой длины, заполняясь символом '0' (или пробелом) для корректного сравнения. На каждом этапе используется стабильная сортировка, например, CountingSort, для сортировки символов текущей позиции.

Чтобы доступно объяснить процесс сортировки, код был подвержен постоянному логированию. Так же внутри него есть множество вставок кода который отвечает за визуализацию сортировки.

Код сортировки представлен на рисунке 3.2.1 – 3.2.2.

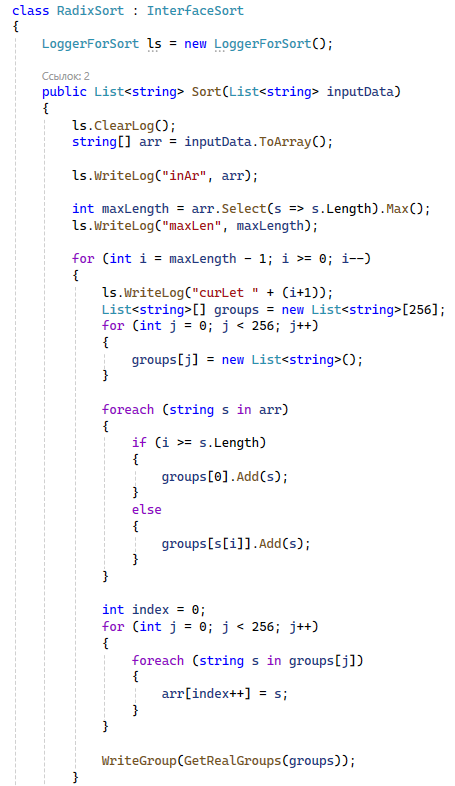


Рис.3.2.1 Код Radix Sort

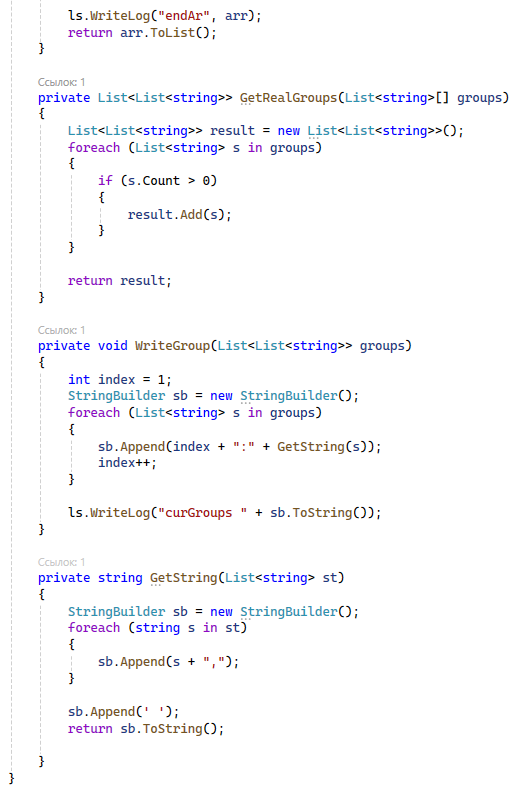


Рис.3.2.2 Код Radix Sort

Чтобы наглядно показать работу этих сортировок было создано два окна под Buble Sort (рис.3.3.1) и Radix Sort (рис.3.3.2), соответсвтенно. В каждом окне есть возможность выгрузить файл с массивом данных с компьютера. Так же пользователь может отслеживать состояния массива прямо во время сортировки, переключая шаги

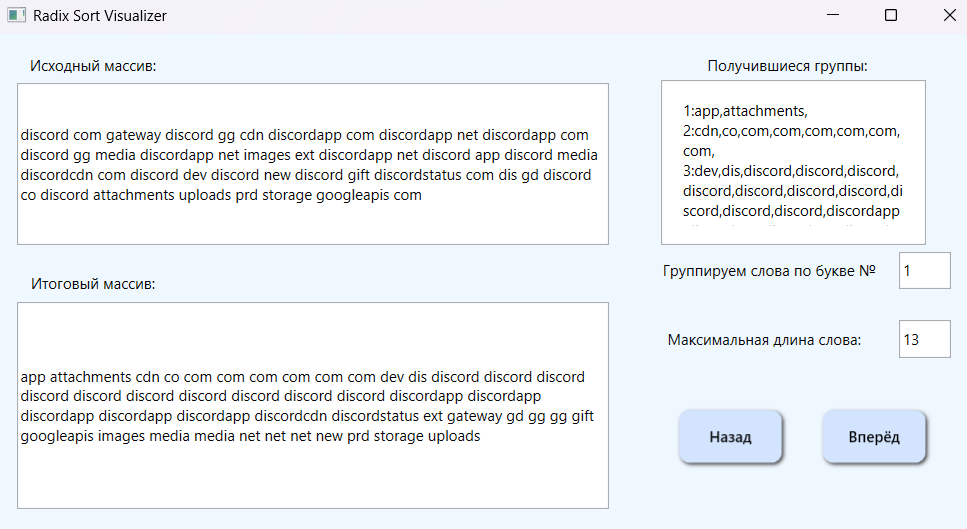


Рис.3.3.2 Визуализация Radix Sort

КАРТИНКА 2