МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по учебной практике

ТЕМА: ГРАФИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ МОДИФИКАЦИЙ СОРТИРОВКИ ПУЗЫРЬКОМ(ШЕЙКЕРНАЯ, ЧЁТ-НЕЧЁТ, РАСЧЁСКОЙ)

Студентка гр. 9382 _____ Балаева М.О.

Студентка гр. 9382 _____ Голубева В.П.

Студентка гр. 9382 _____ Пя С.

Руководитель _____ Ефремов М.А.

Санкт-Петербург

2021

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студентка Балаева М.О. группы 9382

Студентка Голубева В.П. группы 9382

Студентка Пя С. группы 9382

Тема практики: графическое представление модификаций сортировки пузырьком(шейкерная, чёт-нечёт, расчёской)

Задание на практику:

Командная итеративная разработка визуализатора алгоритма(ов) на Java с графическим интерфейсом.

Алгоритм: сортировки: шейкерная, чёт-нечёт, расчёской.

Сроки прохождения практики: 01.06.2021 – 14.07.2021

Дата сдачи отчета: 12.07.2021

Дата защиты отчета: 12.07.2021

Студентка	_____	Балаева М.О.
Студентка	_____	Голубева В.П.
Студентка	_____	Пя С.
Руководитель	_____	Ефремов М.А.

АННОТАЦИЯ

Целью данной практики является изучение нового языка программирования — Java, получение навыков создания графического интерфейса приложения с помощью библитоеки Swing, а также освоение и применение функционала системы сборки Maven.

SUMMARY

The purpose of this practice is to learn a new programming language - Java, gain skills in creating a graphical application interface using the Swing library, as well as mastering and using the functionality of the Maven build system.

СОДЕРЖАНИЕ

	Введение	5
1.	Требования к программе	6
1.1.	Формат входных и выходных данных	6
1.2.	Функционал приложения	6
1.3.	Эскиз интерфейса приложения	7
1.4	Диаграмма классов	
2.	План разработки и распределение обязанностей в бригаде	
2.1.	План разработки	
2.2.	Распределение обязанностей в бригаде	
3.	Особенности реализации	
3.1.	Описание используемых алгоритмов сортировки	
3.1.1	Шейкерная сортировка	
3.1.2	Чётно — нечётная сортировка	
3.1.3	Сортировка расчёской	
3.2.	Описание реализованных классов	
3.2.1	Описание классов графического интерфейса	
3.2.2	Описание классов, отвечающих за логику программы	
4.	Тестирование	
4.1	Тестирование кода алгоритма	
4.2	Тестирование графического интерфейса	

ВВЕДЕНИЕ

Целью практики является освоение нового языка программирования - Java и создания с его помощью визуализации модификации алгоритмов сортировки пузырьком(шейкерная, расчёской, чёт-нечёт). Алгоритмы служат для ознакомления с понятием сортировки массива и наглядно визуализируются.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1. Формат входных и выходных данных

В приложении при генерации:

- Три числа — количество элементов (не более ста), нижняя и верхняя граница

В приложении при вводе вручную:

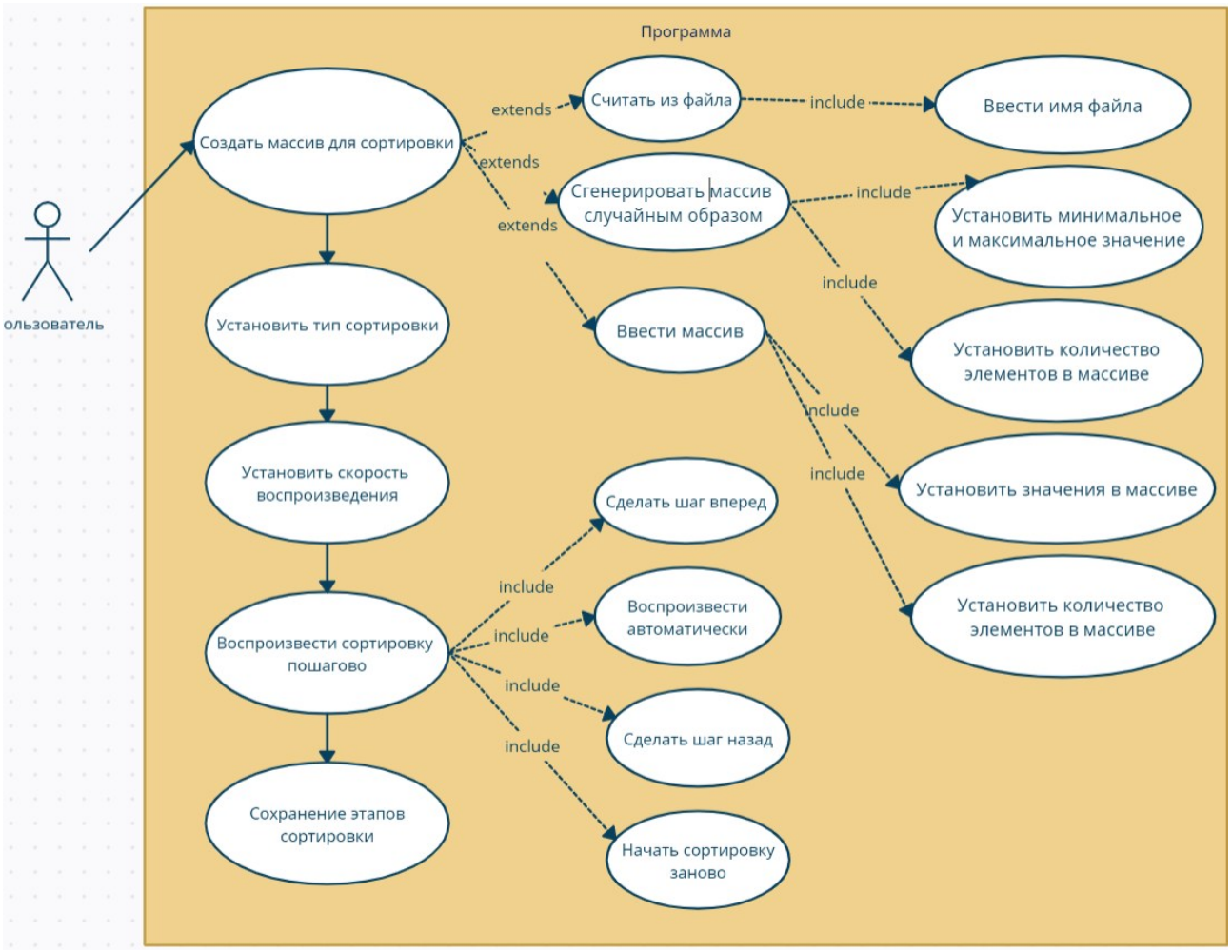
- Массив целых чисел, записанный в строку, разделитель чисел — пробел, не более 100-та чисел.

При вводе из файла:

- Файл формата txt, числа записаны с каждое с новой строки.

1.2. Функционал приложения

Use-case диаграмма:



Пользователю будет предоставлен графический интерфейс для настройки и регулирования визуализации алгоритмов сортировки, а именно: создание массива, выбор типа сортировки, пошаговая/автоматическая итерация сортировки, сохранение этапов сортировки.

Вначале пользователь задает параметры, необходимые для визуализации.

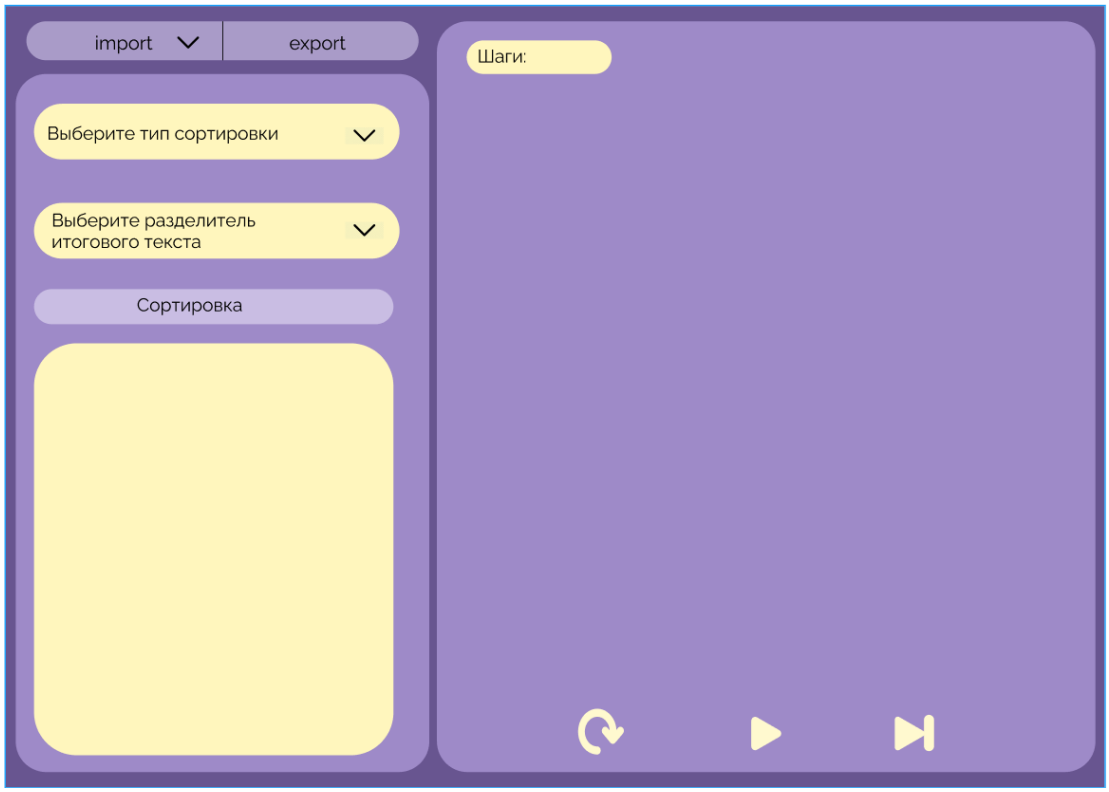
1. Создать массив он может тремя способами: чтения из файла, генерации массива случайным образом, ввода массива.

2. Пользователь может выбрать тип сортировки из предложенных: расческой, чет-нечет, шейкерная.

3. Далее он устанавливает скорость воспроизведения визуализации.

После установки параметров пользователь может начать визуализацию. Он способен запустить ее пошагово, осуществляя шаг вперед, шаг назад, или автоматически. Так же доступна возможность начать сортировку заново.

1.3. Эскиз интерфейса приложения



1.4. Диаграмма классов

2. ПЛАН РАЗРАБОТКИ И РАСПРЕДЕЛЕНИЕ ОБЯЗАННОСТЕЙ В БРИГАДЕ

2.1. План разработки

0) 6/7 июля - создание прототипа, научиться вводить массив чисел для сортировки из файла и в приложении, вывод чисел в файл, реализация алгоритмов и покрытие их юнит-тестами;

1) 8 июля - (1-я версия) реализовать выполнение в автоматическом режиме;

2) 10 - (2-я версия) реализовать выполнение в пошаговом режиме;

3) 12 июля - (финальная версия) отчёт, исправление возникших недочётов.

2.2. Распределение обязанностей в бригаде

Балаева М.О. - создание схемы графического интерфейса, реализация сортировки расчёской, создание прототипа графического интерфейса, графическая реализация алгоритма в автоматическом режиме.

Голубева В.П. - создание плана разработки, реализация шейкерной сортировки, юнит-тестов к сортировкам, настройка pom.xml, ввод-вывод массива для сортировки, раздел с описанием алгоритмов сортировок, общее оформление отчёта, реализация логгирования.

Пя С. - создание диаграммы последовательностей, реализация сортировки чёт-нечёт, создание прототипа графического интерфейса, графическая реализация алгоритма в автоматическом режиме.

3. ОСОБЕННОСТИ РЕАЛИЗАЦИИ

3.1. Описание используемых алгоритмов сортировки

3.1.1. Шейкерная сортировка

Как в «пузырьке» проходим по массиву, сравниваем соседние элементы, выдавливаем максимальный элемент в конец массива. После этого разворачиваемся на 180^0 и идём в обратную сторону, при этом уже перекатывая в начало не максимум, а минимум. Обойдя туда-обратно несколько раз, в итоге заканчиваем процесс, оказавшись в середине списка.

Сложность алгоритма:

$O(n^2)$ - наихудшая сложность, O(n) — наилучшая сложность.

3.1.2. Чётно — нечётная сортировка

Проходим по массиву слева-направо. На первом проходе элементы с нечётным номером сравниваем с соседями на чётных местах. На втором проходе - «чётные по счёту» элементы сравниваем/меняем с «нечётными». Затем снова «нечёт-чёт», потом опять «чёт-нечет». Процесс останавливается тогда, когда после подряд двух проходов по массиву («нечётно-чётному» и «чётно-нечётному») не произошло ни одного обмена.

Сложность алгоритма:

$O(n^2)$ – его наихудшая сложность, в среднем по времени - $O(n \log n)$, O(n) — наилучшая сложность.

3.1.3. Сортировка расчёской

Модификация пузырьковой сортировки с весьма высокой временной сложностью. На массивах до нескольких тысяч элементов показывает скорость выше чем у быстрой сортировки.

Производятся неоднократные прогоны по массиву, при которых сравниваются пары элементов. Если они не отсортированы друг относительно

друга - то производится обмен. В результате крупные элементы мигрируют в конец массива, а небольшие по значению - в начало.

В «пузырьковой сортировке» при каждом прогоне по массиву сравниваются соседние элементы. Здесь же сравниваются элементы, между которыми некоторое фиксированное расстояние. При каждом следующем прохождении расстояние уменьшается, пока не достигнет минимальной величины.

Уменьшающееся расстояние между сравниваемыми элементами рассчитывается с помощью специальной величины, называемой фактором уменьшения. Длина массива делится на этот фактор, это и есть разрыв между индексами. После каждого прохода расстояние делится на фактор уменьшения и таким образом получается новое значение. В конце концов оно сужается до минимального значения - единицы, и массив просто досортировывается обычным "пузырьком".

В результате практических тестов и теоретических исследований оптимальное значение для фактора уменьшения определено следующее:

$$\frac{1}{1-\frac{1}{e^{\varphi}}} \approx 1{,}247330950103979$$

Сложность алгоритма:

$O(n^2)$ – его наихудшая сложность, наилучшая сложность - $O(n \log n)$.

3.2. Описание реализованных классов

3.2.1. Описание классов графического интерфейса

3.2.2. Описание классов, отвечающих за логику программы

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ А
НАЗВАНИЕ ПРИЛОЖЕНИЯ