Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский химико-технологический университет имени Д.И. Менделеева»

Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

Выполнил студент группы КС-30 Положенцев Аксель Алексеевич

Ссылка на репозиторий: [MUCTR-IKT-CPP/PolozhencevAA\_30](https://github.com/MUCTR-IKT-CPP/PolozhencevAA_30)

Приняли: Пысин Максим Дмитриевич

Краснов Дмитрий Олегович

Лобанов Алексей Владимирович

Крашенинников Роман Сергеевич

Дата сдачи: 17.02.2025

Москва 2025

Оглавление

[Описание задачи 3](#_Toc190643407)

[Описание метода 3](#_Toc190643408)

[Выполнение задачи 3](#_Toc190643409)

[Заключение 8](#_Toc190643410)

# Описание задачи

В данной лабораторной работе требуется реализовать алгоритм сортировки перемешиванием (Shaker Sort) и провести его тестирование с измерением параметров времени выполнения. Тестирование должно включать в себя серию измерений, в ходе которых фиксируются различные характеристики работы алгоритма:

* Количество проходов по массиву.
* Количество операций обмена значений.
* Время выполнения сортировки.

Эксперимент проводится в 8 сериях по 20 измерений на серию для массивов различных размеров (1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000 элементов). Все массивы заполняются случайными числами с плавающей запятой в интервале от -1 до 1.

# Описание метода

Сортировка перемешиванием (Shaker Sort) представляет собой двунаправленный вариант сортировки пузырьком. Алгоритм выполняет проходы по массиву в обоих направлениях, что позволяет ускорить процесс сортировки за счёт уменьшения количества неэффективных сравнений.

Характеристики алгоритма:

* Худший случай: O(N²)
* Средний случай: O(N²)
* Лучший случай (отсортированный массив): O(N)
* Используемая память: O(1) (сортировка выполняется на месте)

Преимущества:

* Ускорение сортировки за счёт двунаправленного движения.
* Простота реализации.

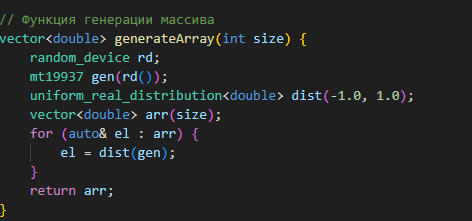
Недостатки:

* Неэффективность на больших объёмах данных по сравнению с более продвинутыми алгоритмами (Быстрая сортировка, Сортировка слиянием).
* Большое количество обменов элементов в худшем случае.

# Выполнение задачи

Реализация выполнялась на языке C++ с использованием стандартных функций работы со временем для замеров производительности, а также для работы с графиками использован язык Python. Программа включает:

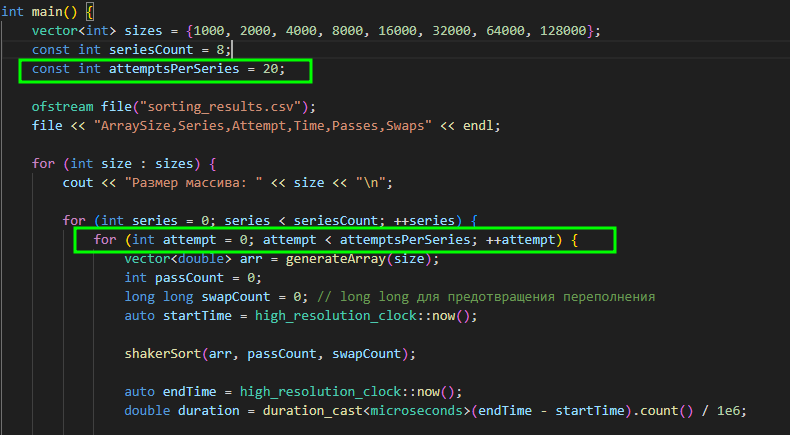
1. Генерацию массива случайных значений.



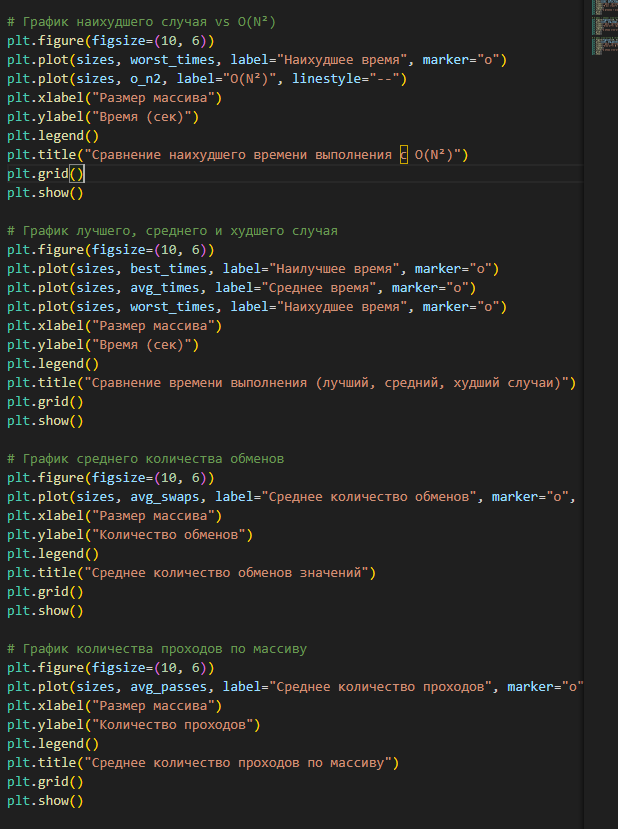
1. Замер времени перед запуском алгоритма.
2. Выполнение сортировки перемешиванием с подсчётом количества проходов и обменов.



1. Замер времени после завершения сортировки и вычисление затраченного времени.
2. Повторение вычислений 20 раз для каждого размера массива.

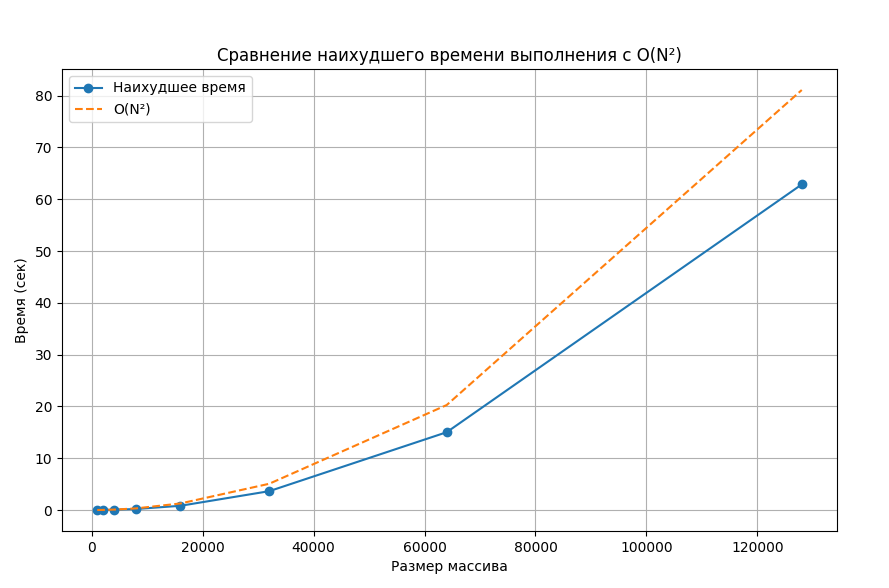


1. Проведение анализа данных и построение графиков зависимости времени от размера массива:



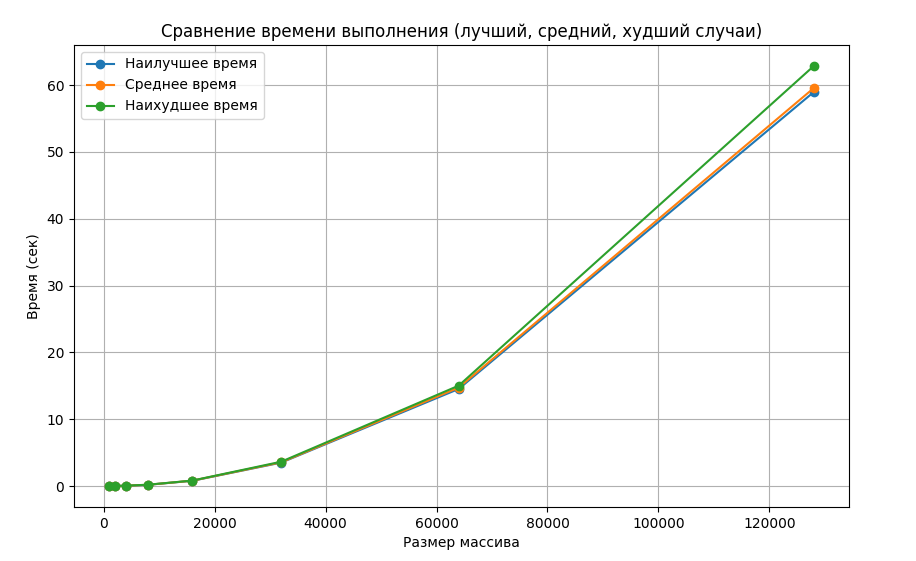
* + Совмещённый график наихудшего времени выполнения сортировки и O(N²) с подбором константы c.

График показывает, что в наихудшем случае время выполнения алгоритма сортировки перемешиванием ниже квадратичной сложности O(N²).



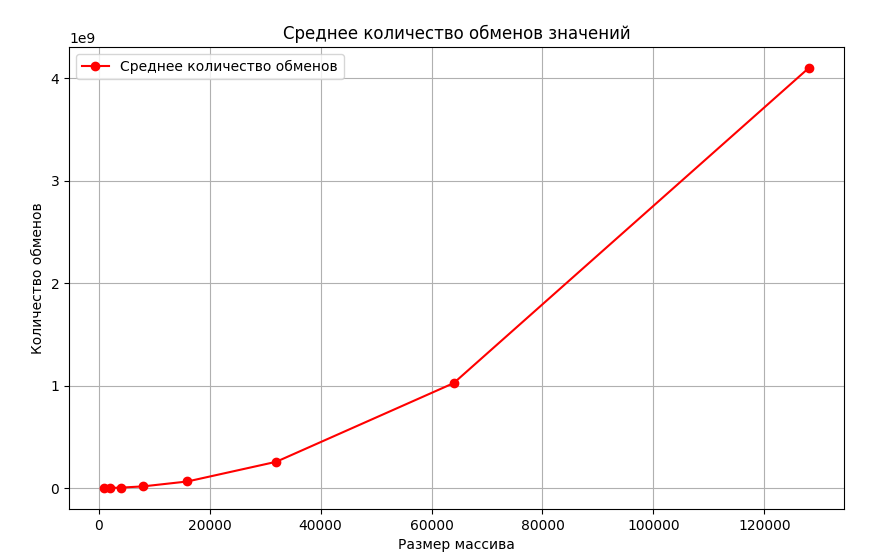
* + График среднего, наилучшего и наихудшего времени исполнения.

Анализ графика демонстрирует значительный разрыв между наилучшим и наихудшим временем выполнения. Среднее время выполнения колеблется между этими границами, близко к наилучшему времени, что указывает на высокую зависимость производительности от исходного порядка элементов в массиве.



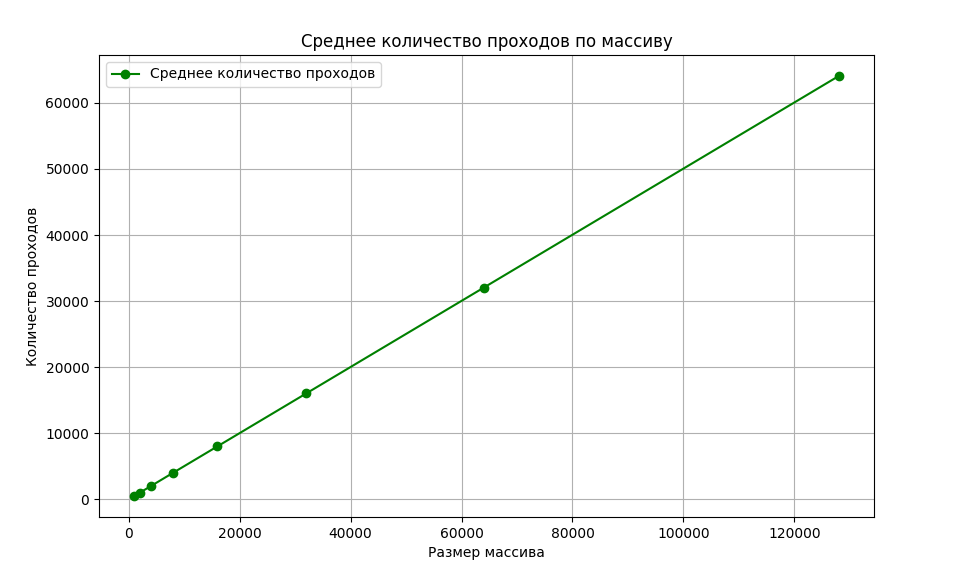
* + График среднего количества обменов значений.

Количество обменов возрастает с увеличением размера массива, что подтверждает квадратичную сложность алгоритма. В среднем, при случайном порядке элементов, алгоритм выполняет значительное количество обменов, что отрицательно сказывается на его производительности.



* + График количества повторных обходов массива.

График показывает, что количество проходов по массиву возрастает почти линейно при увеличении размера массива.



# Заключение

В результате выполнения лабораторной работы был реализован алгоритм сортировки перемешиванием и проведён его анализ. Полученные графики показали, что время выполнения алгоритма соответствует сложности O(N²) в худшем случае. На малых объёмах данных алгоритм показывает удовлетворительную производительность, однако для больших массивов наблюдается значительное замедление по сравнению с более эффективными методами сортировки.

Основные выводы:

* Shaker Sort подходит для небольших массивов, но неэффективен для больших объёмов данных.
* Двунаправленное движение уменьшает количество итераций по сравнению с классическим пузырьковым методом, но не решает проблему квадратичной сложности.
* Проведённые тестирования позволили наглядно проанализировать производительность алгоритма и подтвердить его теоретическую оценку.

Дальнейшие исследования могут быть направлены на сравнение Shaker Sort с другими алгоритмами сортировки и на изучение оптимизаций, позволяющих уменьшить количество обменов и проходов по массиву.