ФЕДЕРАЛЬНОЕ АГЕНСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ Императора Александра I»

Кафедра «Информационные и вычислительные системы»

Дисциплина «Структуры и алгоритмы обработки данных»

ОТЧЁТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1
«Реализация и анализ алгоритмов сортировки»

Выполнил студент Фадхутдинов К.Э.
Факультет: АИТ
Группа: ИВБ-417

Проверил: Забродин А. В.

Санкт-Петербург

2025

Цель:

Разработать и реализовать алгоритмы сортировки данных, провести их сравнительный анализ по времени выполнения и оценить сложность каждого алгоритма в терминах О-символики.

Задания:

1. **Реализовать следующие алгоритмы сортировки:**
   - Сортировка выбором
   - Пузырьковая сортировка
   - Сортировка вставками
   - Сортировка слиянием
   - Пирамидальная сортировка
   - Быстрая сортировка: сайт Архив погоды с 1929 года pogoda-service.ru
   - Лексикографическая сортировка (входными данными может быть журнал, где сделать сортировку по имени, отчеству, n-ой букве фамилии и т.д.)
2. **Анализ:**
   - Время выполнения: для каждого алгоритма измерьте время выполнения сортировки на полученных данных.
   - О-символика: оцените и укажите сложность каждого алгоритма в терминах О-символики.
   - Диаграмма: Постройте диаграмму, отображающую скорость выполнения каждой сортировки для разных наборов данных.

Описание алгоритмов сортировки

Сортировка выбором

На каждой итерации алгоритм находит наименьший элемент в неотсортированной части массива и помещает его в начало этой части. После каждой итерации неотсортированная часть уменьшается на один элемент, а отсортированная часть увеличивается. Алгоритм продолжает этот процесс, пока весь массив не будет отсортирован. **Сложность:** $O(n^2)$ в худшем и среднем случае, где n — длина массива.

Пузырьковая сортировка

В пузырьковой сортировке каждый элемент массива сравнивается с соседним элементом. Если элементы находятся в неправильном порядке, они меняются местами. На каждой итерации (проходе) наибольший элемент из неотсортированной части перемещается в конец. Проходы повторяются, пока массив не станет отсортированным. Если на очередной итерации не

происходит обменов, сортировка завершается. **Сложность:** $O(n^2)$ в худшем и среднем случае, O(n) в лучшем случае (почти отсортированный массив).

Сортировка вставками

Алгоритм сортировки вставками работает с последовательностью чисел, постепенно формируя отсортированную часть массива. На каждом шаге берется очередной элемент из неотсортированной части и вставляется в правильную позицию в отсортированной части, сдвигая элементы при необходимости. Отсортированная часть растет, пока не охватит весь массив. **Сложность:** $O(n^2)$ в худшем и среднем случае, O(n) в лучшем случае (почти отсортированный массив).

Сортировка слиянием

Алгоритм использует стратегию «разделяй и властвуй».

1. Массив делится пополам, пока подмассивы не станут длиной 1 (такие подмассивы считаются отсортированными).
2. Рекурсивно сортируются обе половины массива.
3. Отсортированные половины объединяются в один отсортированный массив: сравниваются первые элементы каждой половины, меньший добавляется в результирующий массив, указатель сдвигается, и процесс повторяется до объединения всех элементов. **Сложность:** O(n log n) в худшем, среднем и лучшем случае.

Пирамидальная сортировка

Пирамидальная сортировка (или сортировка кучей) использует структуру данных «куча» (бинарное дерево, где значение каждого узла больше или равно значений его потомков). Алгоритм:

1. Массив преобразуется в максимальную кучу.
2. Наибольший элемент (вершина кучи) помещается в конец массива.
3. Размер кучи уменьшается, и куча восстанавливается для оставшихся элементов.
4. Процесс повторяется, пока все элементы не будут отсортированы. **Сложность:** O(n log n) в худшем, среднем и лучшем случае.

Быстрая сортировка

Быстрая сортировка также использует стратегию «разделяй и властвуй».

1. Выбирается опорный элемент (в коде — последний элемент подмассива).

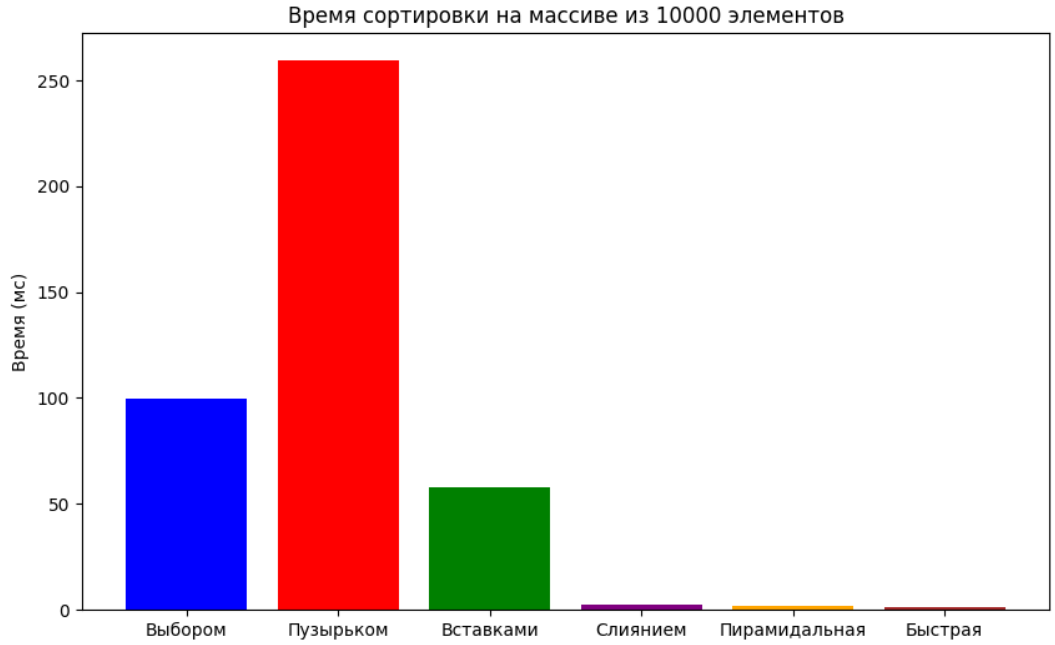
2. Массив разделяется на две части: элементы, меньшие опорного, и элементы, большие или равные опорному.
3. Рекурсивно сортируются обе части массива.
4. После сортировки подмассивов массив оказывается полностью отсортированным. **Сложность:** O(n log n) в среднем, O(n^2) в худшем случае (например, при уже отсортированном массиве и неудачном выборе опорного элемента), O(n log n) в лучшем случае.

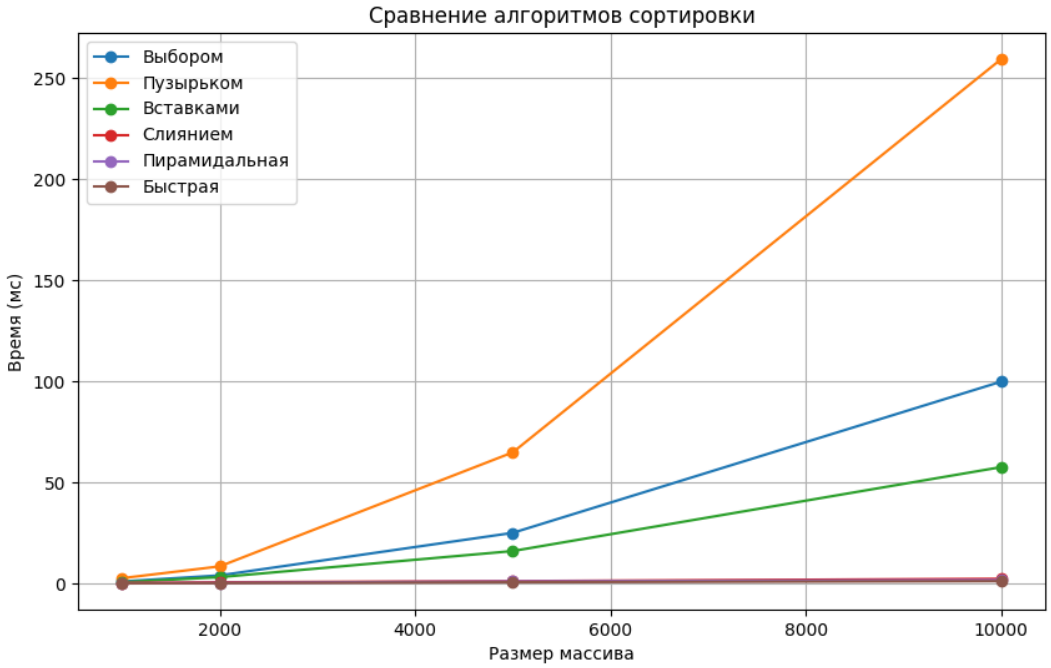
Лексикографическая сортировка (на основе быстрой сортировки)

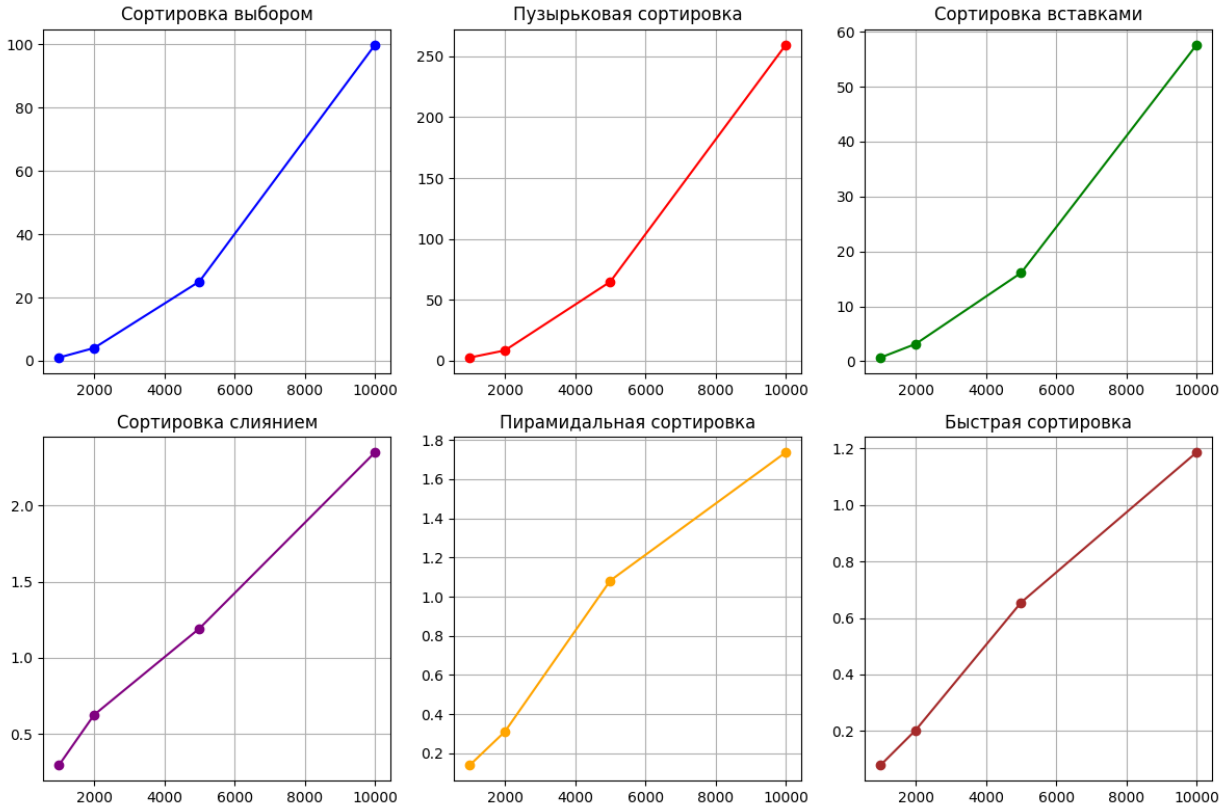
Алгоритм, реализованный в коде для структуры Person, является адаптацией быстрой сортировки для лексикографической сортировки массива объектов, содержащих фамилию, имя и отчество.

1. Сравнение объектов Person происходит по полям: сначала по имени, затем по отчеству, затем по второй букве фамилии, и, наконец, по всей фамилии лексикографически.
2. Как и в обычной быстрой сортировке, выбирается опорный элемент, массив разделяется на две части (элементы, меньшие и большие опорного по заданному порядку), и рекурсивно сортируются подмассивы.
3. Элементы меняются местами с помощью специальной функции manual_swap_person, учитывающей структуру Person. **Сложность:** O(n log n) в среднем, O(n^2) в худшем случае, где n — количество записей.

Диаграммы.







Заключение.

В ходе выполнения лабораторной работы были реализованы алгоритмы сортировки: выбором, пузырьковая, вставками, слиянием, пирамидальная и быстрая, а также адаптированная быстрая сортировка для лексикографической сортировки структуры Person. Проведен сравнительный анализ их производительности по времени выполнения на основе входных

данных из файлов. Каждый алгоритм был оценен с точки зрения временной сложности в терминах O-символики, что позволило выявить их эффективность в различных сценариях (худший, средний и лучший случаи). Результаты демонстрируют, что алгоритмы слияния, пирамидальная и быстрая сортировки обладают лучшей средней сложностью O(n log n), тогда как сортировки выбором, пузырьковая и вставками имеют квадратичную сложность $O(n^2)$, что делает их менее эффективными для больших массивов.