МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Национальный исследовательский**

**Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки: «Прикладная математика и информатика»

**ОТЧЕТ**

по практике

**Тема:**

«Сравнение различных способов сортировки массива»

**Выполнил:**

студент группы 3824Б1ПМ4

Рюмшин.А.А

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

**Научный руководитель:**

профессор

института ИТММ,

Мееров.И.Б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

**Нижний Новгород**

**2023**

[**Введение** 2](#_Toc182258403)

[**Постановка задач** 2](#_Toc182258404)

[**Описание алгоритмов** 3](#_Toc182258405)

[**Описание программной реализации** 6](#_Toc182258406)

[**Результаты экспериментов** 7](#_Toc182258407)

[**Заключение** 7](#_Toc182258408)

[**Литература** 8](#_Toc182258409)

[**Приложение** 8](#_Toc182258410)

# **Введение**

В современном мире обработка и анализ данных являются важнейшими аспектами в различных областях науки. Одним из важнейших этапов обработки данных является их сортировка. Это есть, ничто иное, как распределение элементов в определённом порядке. Сортировка помогает улучшить поиск данных и облегчает анализ информации.

В этой лабораторной работе я хочу рассмотреть несколько разных видов сортировки данных в массиве. Я постараюсь рассказать о трёх видах сортировки массива, выделю их преимущества и недостатки, а также особенности применения в зависимости от требований к скорости, объёму используемой памяти и другим параметрам.

Я буду рассматривать три наиболее популярных метода сортировки данных в массиве:

1. Bubble Sort (сортировка пузырьком);
2. Selection Sort (сортировка выбором);
3. Insertion Sort (сортировка вставками);

# **Постановка задач**

* Описать основные понятия и определения, связанные с сортировкой массивов;
* Изучить и реализовать алгоритмы сортировки: пузырьком, выбором, вставками;
* Провести анализ полученных результатов и сравнить эффективность различных алгоритмов сортировки;

# **Описание алгоритмов**

1. Сортировка пузырьком **(Bubble Sort)**

*Описание:*

Сортировка массива пузырьком – это простой понятный алгоритм сортировки, который работает путём последовательного сравнения и замены соседних элементов массива. Название метода происходит от того, что более «лёгкие» (меньшие по значению) элементы, как бы «всплывают» в массиве, постепенно перемещаются в начало.

*Принцип работы:*

Алгоритм сравнивает каждую пару соседних элементов массива и если, они стоят в неправильном порядке, то меняются местами. Процесс повторяется до тех пор, пока массив не будет полностью отсортирован. На каждой итерации самые «лёгкие» элементы перемещаются в начало массива.

*Программный код:*

void BubbleSort (int mas[], int size) {

for (int i = 0; i < size - 1; ++i) {

for (int j = (size - 1); j > i; j--) {

if (mas[j - 1] >= mas[j]) {

int tmp = mas [j - 1];

mas [j - 1] = mas[j];

mas [j] = tmp;

}

}

}

}

*Преимущества и недостатки:*

+) Главное преимущество этого алгоритма в простоте реализации;

+) Понимание логики;

\*) Метод эффективен для массивов небольшой размерности;

-) Низкая скорость работы для больших массивов;

-) Неэффективное использование памяти;

*Итог:*

Сортировка массива пузырьком – просто и понятный алгоритм, который поможет разобраться с основным принципом сортировки элементов. Для больших массивов существуют более эффективные методы сортировки.

1. Сортировка выбором **(Selection Sort)**

*Описание:*

Сортировка выбором – это алгоритм, который находит минимальный элемент в массиве и перемещает его в начало, затем находит следующий минимальный элемент и перемещает следом за первым.

*Принцип работы:*

Начинаем с того, что просматриваем массив от начала и до конца в поиске минимального элемента. Затем найденный минимальный элемент помещаем на первую позицию, а все остальные «сдвигаем» на одну позицию вправо. Снова просматриваем массив, но начиная со второго элемента (так как первый элемент уже отсортирован). Повторяем этот алгоритм, до тех пор, пока не отсортируется весь массив.

*Программный код:*

void SelectionSort(int mas[], int size) {

for (int i = 0; i < size - 1; i++) {

int minind = i;

for (int j = i + 1; j < size; ++j) {

if (mas[j] < mas[minind]) {

minind = j;

}

}

int tmp = mas[minind];

mas[minind] = mas[i];

mas[i] = tmp;

}

}

*Преимущества и недостатки:*

+) Простота реализации;

+) Возможность сортировки больших массивов;

-) Невысокая скорость работы для небольших массивов;

-) Использование дополнительной памяти для хранения промежуточных итогов;

*Итог:*

Сортировка выбором более эффективна чем пузырьковая сортировка, потому что первый алгоритм находит минимальный элемент за один проход, в то время как сортировка пузырьком выполняет много проходов по массиву. Однако для небольших массивов будет выгоднее использовать пузырьковую сортировку. Для больших массивов лучше использовать более эффективные алгоритмы, такие как сортировка выбором.

1. Сортировка массива вставками **(Insertion Sort)**

*Описание:*

Сортировка вставками – это алгоритм сортировки, который на каждом шаге берёт один элемент обрабатываемого массива и вставляет его в нужное место в уже отсортированной части.

*Принцип работы:*

Основной принцип заключается в том, что мы берём первый элемент массива и вставляем его в нужное место в уже отсортированном подмассиве. Если он меньше, чем элемент в отсортированной части массива, то вставляем его на нужное место в подмассиве. Иначе продолжаем рассмотрение следующего элемента. Повторяем это для всего оставшегося массива.

*Программный код:*

void InsertionSort(int mas[], int size) {

for (int i = 1; i < size; ++i) {

for (int j = i - 1; j >= 0 && j < size; j--) {

if (mas[j] >= mas[j + 1]) {

int tmp = mas[j];

mas[j] = mas[j + 1];

mas[j + 1] = tmp;

}

}

}

}

*Преимущества и недостатки:*

+) Простота реализации алгоритма;

+) Не требует дополнительной памяти;

-) Низкая скорость работы для небольших массивов;

*Итог:*

Алгоритм сортировки массива методом вставок эффективен для больших массивов и для массивов, которые частично отсортированы. Для небольших массивов этот метод тоже подходит.

# **Описание программной реализации**

1. Подключение библиотек

* stdio.h – библиотека для ввода и вывода данных;
* stdlib.h – библиотека для работы с памятью и генерацией случайных чисел;
* time.h – библиотека для работы со временем. В этой программе используется для подсчёта времени сортировки массива каждым из методов;
* cstring – библиотека для работы с массивами, манипуляции строками и массивами;

1. Алгоритмы сортировки

* Сортировка пузырьком;
* Сортировка выбором;
* Сортировка вставками;

Каждый метод сортировки реализован в отдельной функции и принимает в себя массив и его размер.

1. Генерация массива

Массив генерируется с помощью цикла for и заполняется случайными числами в диапозоне от 0 до 50. Случайные числа генерируются с помощью функции rand().

1. Основная функция (main)

В основной функции реализовано:

* Запрос пользователя ввести размерность массива;
* Выделение памяти для массива mas, размерностью length с помощью malloc;
* Создание массива copy и заполнение копии массива элементами из начального массива;
* Запрос пользователя ввести номер метода сортировки;
* Замер времени работы программы проводится через создание переменных t1 и t2, означающих начало и конец работы соответственно;
* На вывод подаётся время которые было потрачено на данный тип сортировки;
* В конце массива освобождается память, выделенная для массива mas[], с помощью функции free();

# **Результаты экспериментов**

Я проводил эксперимент над массивами, в которых минимум 5000 элементов, потому что при меньшей размерности выводится время, которое трудно сравнивать. Время рассчитывается в секундах. 0.15400

|  |  |  |  |
| --- | --- | --- | --- |
| *Тип сортировки*  *Кол-во элементов* | **Bubble Sort** | **Selection Sort** | **Insertion Sort** |
| 5000 | 0.03900 | 0.02800 | 0.04100 |
| 10000 | 0.15400 | 0.10600 | 0.16200 |
| 50000 | 3.87600 | 2.64700 | 4.03600 |
| 100000 | 15.40100 | 10.65900 | 16.31200 |

В результате эксперимента, можно сказать, что самым быстрым методом сортировки является метод сортировки выбором. На любом количестве элементов сортировка пузырьков оказалась быстрее даже сортировки вставками.

# **Заключение**

В результате экспериментов видно, что самым быстрым методом оказался метод сортировки выбором. При увеличении количества элементов, этот метод сохранял лидерство по времени среди других. При сравнении этих алгоритмов, в любом случае нужно делать поправку на то, что массив генерировался случайно и в разных ситуациях массивы могли быть уже отсортированы в какой-то степени.

# **Литература**

<https://translated.turbopages.org/proxy_u/en-ru.ru.5db45388-671a9e53-9bd8420a-74722d776562/https/stackoverflow.com/questions/3557221/how-do-i-measure-time-in-c>

# **Приложение**