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 лабораторной работе

Тема:

**«Сортировка массивов разными способами»**

**Выполнил:**студентка группы 3824Б1ПМ4

Миронова С.И

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

подпись

**Преподаватель: Мееров И.Б**

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

подпись

Нижний Новгород  
2024

**Содержание:**

[**Введение**](#_Toc106028628)  2

[**Постановка задачи** 3](#_Toc106028629)

[**Описание алгоритмов** 4](#_Toc106028630)

[**Описание программной реализации** 6](#_Toc106028631)

[**Результаты экспериментов** 8](#_Toc106028632)

[**Заключение** 9](#_Toc106028633)

Сортировка – это процесс упорядочивания элементов в наборе данных по заданному критерию. Она является одной из важнейших задач в программировании, применяющейся в самых разных областях, от обработки данных до алгоритмов поиска.

В отчёте рассматривается три алгоритма сортировки:

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

**Постановка задачи**

Задача данной работы состояла в том, чтобы создать программу, которая выполняла бы сортировку массива тремя разными способами. А также программа должна выводить время за которое был отсортирован массив. Главной задачей является, выявление наиболее действенного метода сортировки.

**Описание Алгоритма**

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

Сортировка пузырьком — это простейший алгоритм сортировки, который работает путем многократной перестановки соседних элементов, если они находятся в неправильном порядке. Он часто используется для введения концепции сортировки и особенно подходит для сортировки небольших наборов данных.

**Алгоритм пузырьковой сортировки:**

* Нужно начать со сравнения первых двух элементов. Если они в неправильном порядке, поменять их местами.(т.е если первый элемент больше следующего)
* Продолжить этот процесс для всех элементов, двигаясь слева направо. После первого прохода самый большой элемент будет в конце.
* В следующем проходе пропустите последний элемент, так как он уже отсортирован, и повторить шаги выше. Второй по величине элемент переместится на предпоследнюю позицию.
* Повторять шаги, пока весь массив не будет отсортирован.

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

Сортировка выбором — это алгоритм сортировки, основанный на сравнении. Он сортирует массив путем многократного выбора самого маленького (или самого большого) элемента из не отсортированной части и заменяет его первым не отсортированным элементом. Этот процесс продолжается до тех пор, пока не будет отсортирован весь массив.

**Алгоритм сортировки выбором:**

1. Сначала нужно найти самый маленький элемент и поменять его местами с первым элементом, т.е. мы получим самый маленький элемент в его правильном положении.
2. Затем нужно найти самый маленький среди оставшихся элементов (или второй по маленькому) и переместить его в правильное положение, меняя местами.
3. Нужно продолжать делать это до тех пор, пока не переместим все элементы в правильное положение.
4. **Сортировка вставками (Insertion Sort)**

Сортировка вставкой — это простой алгоритм сортировки, который работает путем многократной вставки каждого элемента не отсортированного списка в его правильное положение в отсортированной части списка.

**Алгоритм сортировки вставками:**

* Нужно начать со второго элемента массива, так как предполагается, что первый элемент в массиве отсортирован.
* Сравнить второй элемент с первым и проверить, меньше ли второй элемент, а затем поменять их местами.
* Переходим к третьему элементу и сравниваем его с первыми двумя элементами и ставим на его правильное место
* Повторять до тех пор, пока не будет отсортирован весь массив.

**Описание программной реализации**

В программе было реализовано три сортировки: сортировка пузырьком (Bubble Sort); сортировка выбором (Selection Sort); сортировка вставками (Insertion Sort). Пользователь выбирает одну из сортировок для реализации массива случайных целых чисел. Далее расписано описание каждой части программы.

1. **Подключение библиотек:**

* «<stdio.h>» - заголовочный файл для стандартного ввода – вывода.
* <stdlib.h> - содержит в себе функции, занимающиеся выделением памяти, контролем процесса выполнения программы и генерации случайных чисел.
* <time.h> - определяет функции даты и времени.

1. **Алгоритмы сортировок.**

* Сортировка пузырьком (Bubble Sort)

Проходит по массиву и сравнивает соседние элементы, меняет их местами, если они находятся в неправильном порядке. Процесс повторяется до тех пор, пока массив не будет отсортирован.

* Сортировка выбором (Search Sort)

Находит минимальный (или максимальный) элемент в несортированной части массива и меняет его с первым элементом.

* Сортировка вставкой (Insertion Sort)

Строит отсортированное подмножество в массиве, постепенно добавляя элементы из неотсортированной части. Каждый элемент вставляется в правильное положение в отсортированном подмножестве.

1. **Генерация массива**

Функция array\_generating заполняет массив случайными целыми числами в диапазоне от 0 до 999. Для генерации случайных чисел используется функция rand(), инициализированная с помощью srand(time(NULL)), что обеспечивает разнообразие значений при каждом запуске программы.

1. **Основная функция**

* Запрос размера.
* Выделение памяти для массива mas с помощью функции malloc. Здесь mas = (int\*)malloc(size \* sizeof(int)); выделяет память для массива целых чисел размером size. Использование sizeof(int) позволяет определить, сколько байт нужно выделить для массива целых чисел.
* Проверка успешности выделения памяти. Если mas равно NULL, программа выводит сообщение об ошибке и завершает выполнение.

1. **Цикл выбора сортировки**

Программа предлагает выбрать тип сортировки. В зависимости от выбора, массив mas копируется во временный массив copy с помощью функции memcpy, чтобы сохранить исходные данные для каждой сортировки.

1. **Измерение времени выполнения**

Для каждой сортировки используется функция clock() для измерения времени выполнения. Время выполнения каждой сортировки сохраняется в переменных time1, time2 и time3.

1. **Вывод результатов**

После завершения сортировок программа выводит время выполнения каждого алгоритма на экран.

## **Освобождение памяти**

В конце программы освобождается память, выделенная для массива mas, с помощью функции free().

**Результаты экспериментов**

В своих экспериментах я использовала массивы, содержащие не менее 10 000 элементов. Это связано с тем, что при меньшем количестве элементов время выполнения алгоритмов сортировки слишком мало, чтобы его можно было достоверно сравнивать, так как оно измеряется в сотых долях секунды.

10000 элементов:

Bubble Sort – 0.156 с

Search Sort – 0.036 с

Insertion Sort – 0.034 с

Результаты: сортировка ставками – лучше.

50000 элементов:

Bubble Sort – 5.079 с

Search Sort – 0.954с

Insertion Sort – 0.882 с

Результаты: сортировка ставками – лучше.

100000 элементов:

Bubble Sort – 16.717с

Search Sort – 3.61 с

Insertion Sort – 2.341 с

Результаты: сортировка ставками – лучше.

**Заключение**

Результаты эксперимента показывают, что наиболее эффективной из рассмотренных является сортировка выбором (Insertionsort). На небольших массивах разница во времени выполнения различных алгоритмов незначительна, поэтому можно использовать любой из них. Однако, при работе с массивами большой длины сортировка выбором демонстрирует наилучшую производительность. Сортировка выбором также работает достаточно быстро и эффективно, но все же уступает сортировке ставками по скорости. Сортировка пузырьком, напротив, оказывается крайне неэффективной при работе с большими массивами, требуя значительного времени для выполнения.