

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:
студент группы 3824Б1ПМ4
Овчинников В.А.

подпись

Преподаватель:
Куклин А.Е.

подпись

Нижний Новгород
2024

Содержание:

Введение.....	2
Постановка задачи	2
Описание алгоритмов	2
Описание программной реализации	3
Результаты экспериментов	5
Заключение.....	6
Литература	6
Приложение.....	6

Введение

Сортировка массивов - это один из основных алгоритмов, который используется в программировании для упорядочивания элементов в массиве по определенному критерию. Сортировка позволяет упорядочить данные в порядке возрастания или убывания, что упрощает поиск, анализ и обработку информации. Существует множество методов сортировки, каждый из которых имеет свои особенности, эффективность и область применения. В данной работе рассмотрим несколько из них и сравним их производительность и сложность

В этом отчете рассматриваются три популярных алгоритма сортировки:

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

Постановка задачи

Задача состояла в создании программы, которая создает массив, генерируя в нем случайные числа. Затем сортирует тремя разными вариантами: Пузырьковая сортировка, Сортировка выбором и Сортировка вставками. Далее программа должна вывести время каждой сортировки, за которое она отсортировала массив, мы же в свою очередь должны выяснить какой метод сортировки является наилучшим для определенного размера массива.

Описание алгоритмов

1. Сортировка пузырьком (Bubble Sort)

Описание:

Пузырьковая сортировка – это один из самых простых методов сортировки элементов массива. Он получил свое название от способа сортировки, при котором наименьшие элементы "поднимаются" к началу массива, как пузырьки в воде.

Принцип работы:

1. Начинаем с первого элемента массива.
2. Сравниваем текущий элемент со следующим.
3. Если текущий элемент больше следующего, меняем их местами.
4. Переходим к следующему элементу и повторяем шаги 2-3.
5. После завершения прохода по массиву, повторяем процесс, пока не будет выполнен полный проход без изменений.

2. Сортировка выбором (Selection Sort)

Описание:

Сортировка выбором — это алгоритм, который сортирует массив, находя наименьший элемент в неотсортированной части массива и перемещая его в начало отсортированной части. Этот процесс повторяется для всех элементов массива.

Принцип работы:

1. Разделите массив на отсортированную и неотсортированную части.
2. На каждой итерации находите наименьший элемент в неотсортированной части.
3. Меняйте местами найденный элемент с первым элементом неотсортированной части.
4. Увеличивайте границу отсортированной части на один элемент и повторите процесс.

3. Сортировка вставками (Insertion Sort)

Описание:

Сортировка вставками - это алгоритм сортировки, который работает путем последовательного вставления каждого элемента в отсортированную часть массива.

Принцип работы:

1. Начинаем с первого элемента, который считается отсортированным.
2. Берем следующий элемент и сравниваем его с отсортированной частью.
3. Вставляем элемент в правильное положение, сдвигая все большие элементы вправо.
4. Повторяем процесс для всех элементов массива.

Описание программной реализации

В данной программе реализованы три алгоритма сортировки: сортировка пузырьком, сортировка выбором и сортировка вставками. Программа запрашивает длину массива у пользователя, а затем предлагает выбрать режим сортировки. Ниже представлено подробное описание каждой части программы.

1. Подключение библиотек

- **stdio.h:** Библиотека для ввода и вывода данных.
- **stdlib.h:** Библиотека для работы с памятью и генерации случайных чисел.
- **time.h:** Библиотека для работы с временем, используется для измерения времени выполнения сортировок.
- **cstring:** Библиотека для работы с функциями манипуляции строками и массивами (в данном случае используется для функции `memset`).

2. Алгоритмы сортировки

- **Сортировка пузырьком (bubble_sort):**
 - Проходит по массиву и сравнивает соседние элементы, меняя их местами, если они находятся в неправильном порядке. Процесс повторяется до тех пор, пока не будет выполнен полный проход без изменений.

- **Сортировка выбором (search_sort):**
 - На каждой итерации находит наименьший элемент в неотсортированной части массива и перемещает его в начало отсортированной части.
- **Сортировка вставками (insertion_sort):**
 - Строит отсортированный массив, вставляя каждый элемент в правильное положение относительно уже отсортированных элементов.

Каждый из алгоритмов реализован в отдельной функции, принимающей массив и его размер в качестве аргументов.

3. Генерация массива

Генерация массива происходит через функцию `rand()`, которая дает псевдочисла. Они постоянны, но благодаря функции `srand(time(NULL))` мы генерируем рандом числа. Массив случайным образом генерируется числами от 0 до 999.

4. Основная функция

В основной функции происходит:

- Запрос размера массива у пользователя.
- 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.
- Запрос выбора сортировки.

5. Цикл выбора сортировки

В этом блоке программа от запрошенного числа сортирует массив. В зависимости от выбора, массив `mas` копируется во временный массив `copy` с помощью функции `memcpy`, чтобы сохранить исходные данные для каждой сортировки.

6. Измерение времени выполнения

Для каждой сортировки используется функция `clock()` для измерения времени выполнения. Время выполнения каждой сортировки сохраняется в переменных `cpu_time1`, `cpu_time2` и `cpu_time3`.

7. Вывод результатов

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

8. Освобождение памяти

В конце программы освобождается память, выделенная для массива `mas`, с помощью функции `free()`, что предотвращает утечки памяти.

Результаты экспериментов

Я проводил эксперименты над массивами, содержащими минимум 10000 элементов, так как при меньшем их содержании выводиться время, которое трудно сравнивать так как оно измеряется в сотых секундах.

10000 элементов:

Bubble sort – 0.27 с

Search sort – 0.18 с

Insertion sort – 0.19 с

Результаты у Search и Insertion sort практически одинаковые.

20000 элементов:

Bubble sort – 1.33 с

Search sort – 0.73 с

Insertion sort – 0.78 с

Лучший результат в этом тесте показал Search sort.

50000 элементов:

Bubble sort – 9.52 с

Search sort – 4.58 с

Insertion sort – 4.68 с

Лучший результат в этом тесте показал Search sort.

100000 элементов:

Bubble sort – 30.85 с

Search sort – 15.6 с

Insertion sort – 16.84 с

Лучший результат в этом тесте показал Search sort.

Заключение

Из результатов экспериментов видно, что Bubble sort при 20000+ элементов значительно отстает от Search и Insertion sort. А разница между Search и Insertion sort в основном в долях секундах, я считаю, что для массивов малой длины можно использовать любую из этих сортировок, но вот для большой длины уже более сложные сортировки.

Литература

<https://stackoverflow.com/questions/3557221/how-do-i-measure-time-in-c>

Приложение

<https://github.com/cishka/god-damn>