

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

**Отчет по выполнению практического задания номер один**

**Тема: Эмпирический анализ алгоритмов сортировки**

**Дисциплина: Структуры и алгоритмы обработки данных**

Выполнил студент Михайлюк Д. С.

Группы ИНБО-07-21

Москва 2021

**СОДЕРЖАНИЕ**

[1 Алгоритм Простой вставки (Insertion sort) 3](#_Toc96562331)

[1.1 Алгоритм сортировки по методу простой вставки 3](#_Toc96562332)

[1.2 Описание процесса определения функции роста времени выполнения сортировки методом простой вставки, при увеличении объема массива n 4](#_Toc96562333)

[1.3 Сводная таблицу результатов выполнения сортировки по указанным объемам, на случайно заполненном массиве для всех указанных объемов 4](#_Toc96562334)

[1.4 Код алгоритма и основной программы 5](#_Toc96562335)

[1.5 График зависимости теоретической Тт=f(С+М)=О(f(n)) и практической Тп=(Сф+Мф) вычислительной сложности алгоритма от размера n массива 7](#_Toc96562337)

[1.6 Анализ результатов для задания 1 8](#_Toc96562338)

[2 Задание 9](#_Toc96562339)

[2.1 Таблица результатов при применении к массиву, упорядоченному по возрастанию 9](#_Toc96562340)

[2.2 Таблица результатов при применении к массиву, упорядоченному по убыванию 9](#_Toc96562341)

[2.3 Код программы, используемой для тестов, результат которых отображен в пунктах 2.1 и 2.2 10](#_Toc96562342)

[2.4 График зависимости теоретической и практической вычислительной сложности алгоритма для трех рассмотренных случаев 12](#_Toc96562343)

[2.5 Емкостная сложность алгоритма от n 13](#_Toc96562344)

[2.6 Анализ результатов и выводы из полученных данных 13](#_Toc96562345)

[3 Алгоритм простого выбора (Selection sort) 14](#_Toc96562346)

[3.1 алгоритм сортировки по методу простого выбора (Selection sort) 14](#_Toc96562347)

[3.2 Описание процесса определения функции роста времени выполнения сортировки методом простого выбора, при увеличении объема массива n 15](#_Toc96562348)

[3.3 Сводная таблицу результатов выполнения сортировки по указанным объемам, на случайно заполненном массиве для всех указанных объемов 15](#_Toc96562349)

[3.5 Код всей программы алгоритма простого выбора 16](#_Toc96562350)

[3.6 График зависимости теоретической и практической вычислительной сложности алгоритма для двух рассмотренных алгоритмов: Алгоритм простой вставки и алгоритм простого выбора 18](#_Toc96562351)

[3.7 Определение эффективности алгоритма 19](#_Toc96562352)

[3.8 Анализ результатов 19](#_Toc96562353)

[4 Выводы 20](#_Toc96562354)

[5 Список информационных источников 21](#_Toc96562355)

# Алгоритм Простой вставки (Insertion sort)

## Алгоритм сортировки по методу простой вставки

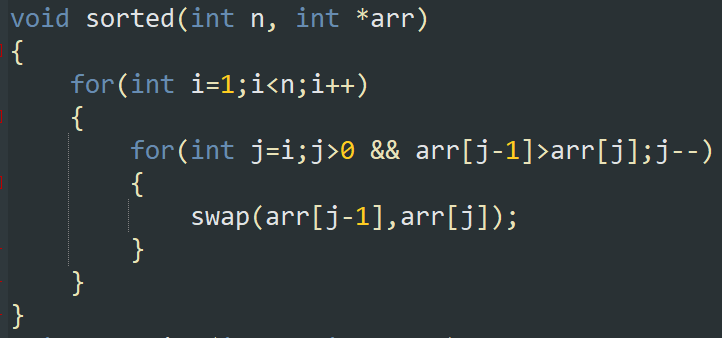


Рис. 1 – Алгоритм сортировки, реализованный через процедуру на языке C++

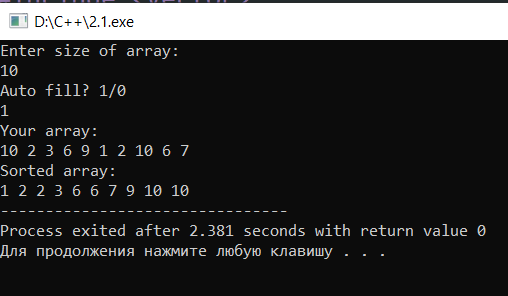


Рис. 2 – Проверка работоспособности алгоритма, на массиве состоящем из 10 элементов, заполненных случайным образом

## Описание процесса определения функции роста времени выполнения сортировки методом простой вставки, при увеличении объема массива n

Теоретическая вычислительная сложность алгоритма: – что является разновидностью квадратичной зависимости n2

График 1 – сложность алгоритма

## Сводная таблицу результатов выполнения сортировки по указанным объемам, на случайно заполненном массиве для всех указанных объемов

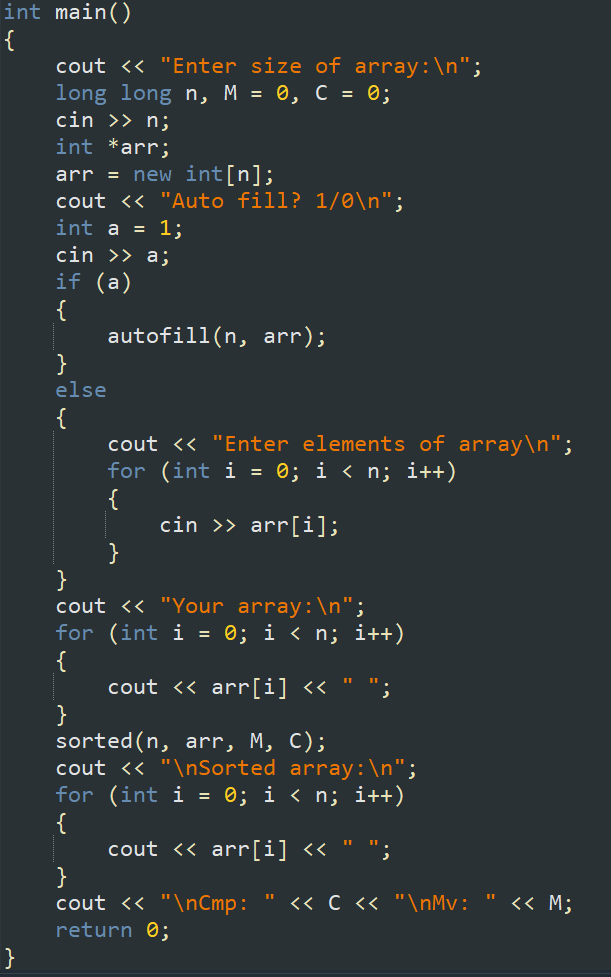
Таблица 1 - таблицу результатов для алгоритма простой вставки

|  |  |  |  |
| --- | --- | --- | --- |
| **n** | **T(n) (сек)** | **Тт=f(C+M)** | **Тп=Cф+Mф** |
| 100 | 0.07785 | 10100 | 7559 |
| 1000 | 0.6756 | 1001000 | 745012 |
| 10000 | 1.732 | 100010000 | 75187011 |
| 100000 | 11.76 | 10000100000 | 7499642861 |
| 1000000 | 114.69 | 1000001000000 | 748855647654 |

## Код алгоритма и основной программы

## 

Рисунок 3 – код алгоритма сортировки и автозаполнения массива



Рисуноу 4 – Код основной программы

## График зависимости теоретической Тт=f(С+М)=О(f(n)) и практической Тп=(Сф+Мф) вычислительной сложности алгоритма от размера n массива

График 2 – зависимость теоритической и практической сложностей алгоритма

## Анализ результатов для задания 1

В результате проделанной работы был создан алгоритм, была определена его теоретическая сложность по вычислительной и емкостной сложности, алгоритм был протестирован на различных размерах исходного, неотсортированного массива, а результаты тестирования были наглядно отображены в табличной и графической форме.

# Задание

## Таблица результатов при применении к массиву, упорядоченному по возрастанию

Таблица 2 – результаты тестирования на отсортированном по возрастанию массиве

|  |  |  |  |
| --- | --- | --- | --- |
| **n** | **T(n)** | **Тт=f(C+M)** | **Тп=Cф+Mф** |
| 100 | 0.5727 | 10100 | 4950 |
| 1000 | 0.6009 | 1001000 | 499500 |
| 10000 | 1.665 | 100010000 | 49995000 |
| 100000 | 15.46 | 10000100000 | 4999950000 |
| 1000000 | 103.54 | 1000001000000 | 499999500000 |

## Таблица результатов при применении к массиву, упорядоченному по убыванию

Таблица 3 – результаты тестирования на отсортированном по убыванию массиве

|  |  |  |  |
| --- | --- | --- | --- |
| **n** | **T(n)** | **Тт=f(C+M)** | **Тп=Cф+Mф** |
| 100 | 0.5956 | 10100 | 9900 |
| 1000 | 0.6489 | 1001000 | 999000 |
| 10000 | 1.774 | 100010000 | 99990000 |
| 100000 | 17.84 | 10000100000 | 9999900000 |
| 1000000 | 127.324 | 1000001000000 | 999999000000 |

## Код программы, используемой для тестов, результат которых отображен в пунктах 2.1 и 2.2

Поскольку упомянутые выше программы отличаются от исходных только разным способом заполнения массива, предоставляется лишь часть, отличающаяся от исходного алгоритма.

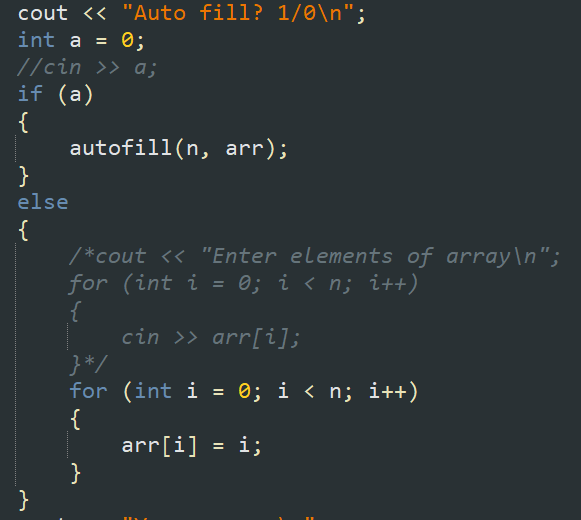


Рисунок 5 – заполнение массива для тестирования алгоритма на массиве из отсортированных по возрастанию элементов

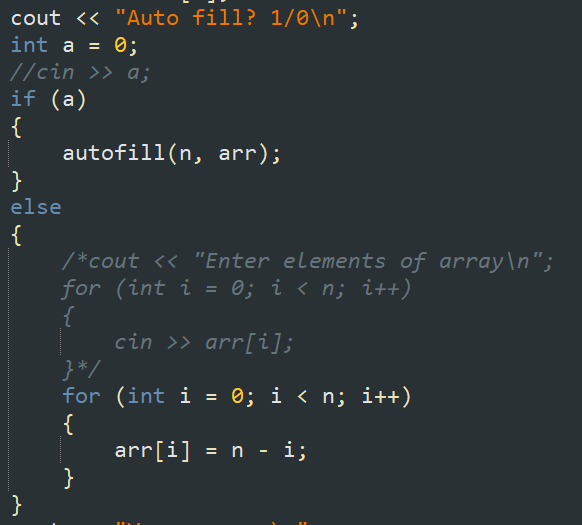


Рисунок 6 – заполнение массива для тестирования алгоритма на массиве из отсортированных по убыванию элементов

## График зависимости теоретической и практической вычислительной сложности алгоритма для трех рассмотренных случаев

Поскольку теоретическая сложность не учитывает состав массива она будет одинакова для всех 3 случаев, поэтому она приведена в единственном экземпляре.

График 3 – сравнение вычислительной сложности алгоритма при трех разных случаях задания исходного массива

## Емкостная сложность алгоритма от n

Основную часть емкостной сложности алгоритма при больших n будет занимать массив, а поэтому емкостная сложность алгоритма равна размеру массива:

## Анализ результатов и выводы из полученных данных

В ходе проделанной работы были проведены тесты алгоритма, на массиве из отсортированных по убыванию и по возрастанию значений. Так же полученные данные были сравнены между собой и с результатом тестирования алгоритма на массиве из случайных данных, на основании чего можно сделать вывод о том что вычислительная сложность алгоритма обрабатывающая массив, отсортированный по убыванию максимальна, и почти идентична теоретической вычислительной сложности алгоритма, вычислительная сложность алгоритма обрабатывающая массив, отсортированный по возрастанию минимальна, ибо в ней исключены перестановки элементов, средней вычислительная сложность алгоритма обладает алгоритм, сортирующий массив сгенерированный случайно.

# Алгоритм простого выбора (Selection sort)

## алгоритм сортировки по методу простого выбора (Selection sort)

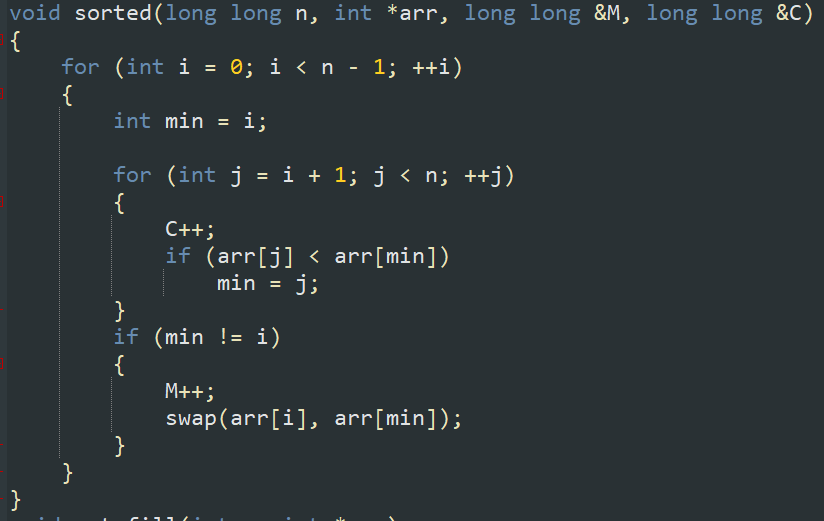


Рисунок 7 - Алгоритм простого выбора реализованный на языке C++

## Описание процесса определения функции роста времени выполнения сортировки методом простого выбора, при увеличении объема массива n

Теоретическая вычислительная сложность алгоритма: – что является разновидностью квадратичной зависимости n2

График 4 – сложность алгоритма

## Сводная таблицу результатов выполнения сортировки по указанным объемам, на случайно заполненном массиве для всех указанных объемов

Таблица 4 - таблицу результатов для алгоритма простого выбора

|  |  |  |  |
| --- | --- | --- | --- |
| **n** | **T(n) (сек)** | **Тт=f(C+M)** | **Тп=Cф+Mф** |
| 100 | 0.08866 | 5050 | 5046 |
| 1000 | 0.5068 | 500500 | 500494 |
| 10000 | 0.6098 | 50005000 | 50004990 |
| 100000 | 5.114 | 5000050000 | 5000049981 |
| 1000000 | 114.69 | 500000500000 | 500000499924 |

## Код всей программы алгоритма простого выбора

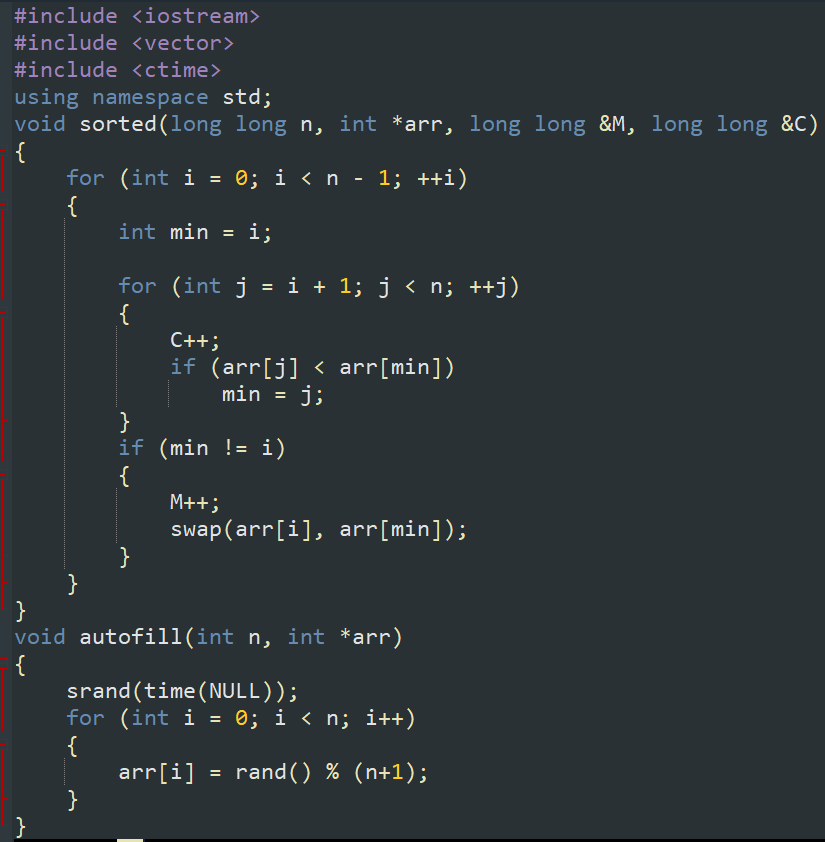


Рисунок 8 – алгоритм простого выбора и функция заполнения массива случайными числами

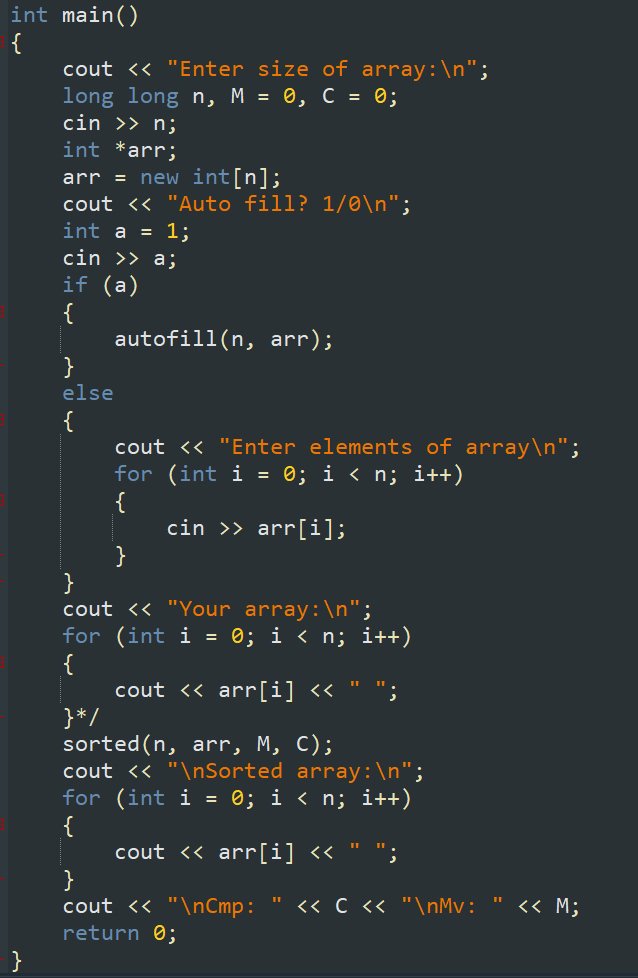


Рисунок 9 – Код основной программы

## График зависимости теоретической и практической вычислительной сложности алгоритма для двух рассмотренных алгоритмов: Алгоритм простой вставки и алгоритм простого выбора

График 5 - График зависимости теоретической и практической вычислительной сложности алгоритма для двух рассмотренных алгоритмов

## Определение эффективности алгоритма

Эффективность алгоритма — это свойство алгоритма, которое связано с вычислительными ресурсами, используемыми алгоритмом.

## Анализ результатов

Опираясь на вычисленную теоретическую вычислительную сложность алгоритмов, а также полученную практическую вычислительную сложность для каждого алгоритма, их подробного сравнения можно утверждать, что алгоритм простого выбора эффективнее алгоритма простой вставки, причем при возникновении наихудшего случая алгоритм простого выбора почти вдвое эффективней алгоритма простой вставки, при возникновении наилучшего случая же эти алгоритмы равны по вычислительной сложности.

# Выводы

В ходе проделанной работы были получены практические знания по написанию некоторых видов простых алгоритмов сортировки, практические знания по методологии тестирования и определению вычислительной сложности различных алгоритмов сортировки, помимо этого были получены теоретические знания по классификации и терминологии в области сортировок.

# Список информационных источников

1. Васильев А.Н. самоучитель C++ 3е изд. 2016г.
2. Занятие 2 (простые сортировки) – методический материал
3. СиАОД лекции Красников С.А. 2021г.