Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное   
учреждение высшего образования

Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

**Отчет по лабораторной работе**

**«Сортировки»**

**Выполнил**:

студент группы 3822Б1ПМ1

Шляков Максим Сергеевич

**Проверил**:

преподаватель каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2022

**Содержание**

[Постановка задачи 3](#_Toc26962562)

[Метод решения 4](#_Toc26962563)

[Руководство пользователя 5](#_Toc26962564)

[Описание программной реализации 6](#_Toc26962565)

[Подтверждение корректности 7](#_Toc26962566)

[Результаты экспериментов 8](#_Toc26962567)

[Заключение 9](#_Toc26962568)

[Приложение 10](#_Toc26962569)

# Постановка задачи

Итоговая версия программы должна давать возможность пользователю, при помощи примитивного интерфейса (меню), создавать массив определенной длины, сортировать его одной из четырех представленных сортировок: “пузырьком”, Хоара, вставками, Шелла (по выбору пользователя). Также необходимо провести оценку зависимости количества операций, выполняемых каждым алгоритмом, от размера входных данных (сортируемого массива).

# Метод решения

**Пузырьковая сортировка**

Идея метода состоит в том, чтобы попарно сравнивать соседние элементы. Каждый проход начинается с начала последовательности. Сравнивается последний элемент с предпоследним: если порядок между ними нарушен, то они меняются местами. И так далее; элементы с неправильным порядком в паре меняются местами. В итоге, после первого прохода, минимальный элемент будет находится на первом месте в массиве, он как бы «всплывет» наверх. Именно поэтому этот метод называется **сортировка пузырьком**. На следующем проходе рассматривается последовательность от 1 до N, затем от 2 до N-1, и так до конца. После каждого прохода можно делать проверку: выполнялись ли перестановки элементов. Если не выполнялись, то сортировка завершена.

**Быстрая сортировка (сортировка Хоара)**

Выбрать опорный элемент в массиве — часто встречается вариант с центральным элементом.Разделить массив на две части следующим образом: все элементы из левой части, которые больше или равны опорному, перекладываем в правую, аналогично, все элементы из правой, которые меньше или равны опорному кладем в левую часть.В результате предыдущего шага в левой части массива останутся элементы, которые меньше или равны центральному, а в правой — больше либо равны.

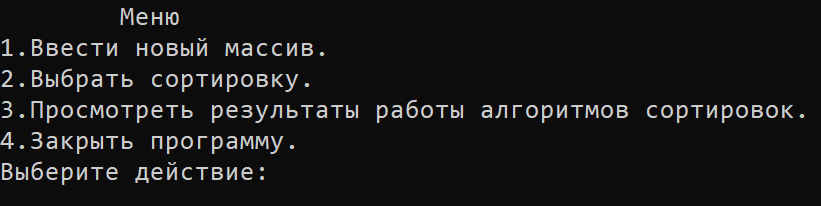
**Сортировка вставками**

Сортировка вставками — это алгоритм, в котором элементы массива просматриваются по одному, и каждый элемент элемент размещается в подходящее место среди ранее упорядоченных элементов.

**Сортировка Шелла**

Идея метода заключается в сравнение разделенных на группы элементов последовательности, находящихся друг от друга на некотором расстоянии. Изначально это расстояние равно d или N/2, где N — общее число элементов. На первом шаге каждая группа включает в себя два элемента расположенных друг от друга на расстоянии N/2, они сравниваются между собой, и, в случае необходимости, меняются местами. На последующих шагах также происходят проверка и обмен, но расстояние d сокращается на d/2, и количество групп, соответственно, уменьшается. Постепенно расстояние между элементами уменьшается, и на d=1 проход по массиву происходит в последний раз.

# Руководство пользователя

1. Пользователь запускает программу, перед ним появляется главное меню (Рис. 1) .Рис. 1

При помощи цифр на клавиатуре, пользователю необходимо выбрать один из 4 пунктов, представленных в меню (Рис. 1).

1. Команда “1” позволяет ввести размер массива, который необходимо отсортировать (Рис. 2) и его элементы (Рис. 3).



Рис. 2



Рис. 3

После окончания ввода элементов массива, на экран выводится сообщение об успешном завершении операции и набор введенных чисел. ( Пример представлен на Рис. 4)

Рис. 4

Далее программа возвращает пользователя в меню, сохраняя все введенные значения.

1. Команда “2” позволяет пользователю выбрать одну из четырех реализованных сортировок для упорядочения введенного массива.

После ввода “2” с клавиатуры, программа направляет пользователя в меню сортировок (Рис. 5) и позволяет выбрать необходимую сортировку.

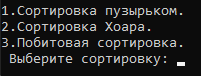


Рис. 5

Примечание: если пользователь не ввел массив для сортировки, программа не позволит перейти в меню сортировок, предупредив пользователя о необходимости создать массив. (Рис. 6)



Рис 6.

1. Команда “3” позволяет посмотреть результаты работы алгоритмов сортировки.
2. Команда “4” завершает работу программы.

# Описание программной реализации

1. Функция int main();

В данной функции, в первую очередь, инициализируются основные переменные программы (Рис.6)

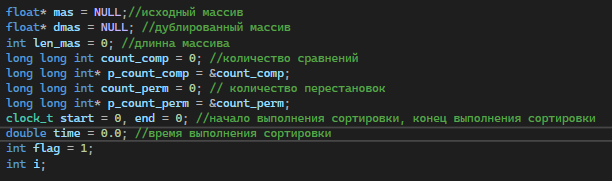
. 

Рис.6

Далее вызывается оператор switch, который в зависимости от возвращаемого значения функции menu, выполняет различный набор действий.



Рис. 7

* 1. Если значение menu == 1, программа просит пользователя ввести размер задаваемого массива (len\_mas) и при помощи функции malloc выделяет определенное количество памяти, в зависимости от значения len\_mas (Рис.7).

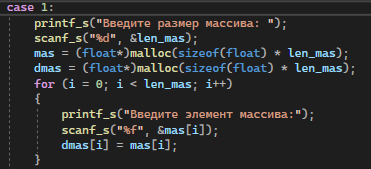


Рис.8

Далее представлен ввод элементов массива с клавиатуры (Рис.8)

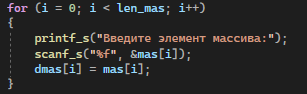


Рис.9

После успешного ввода элементов с клавиатуры, программа выводит сообщение об успешном заполнении массива и введенные значения (Рис. 9). Затем ветка действий при значении menu == 1 завершается оператором break.

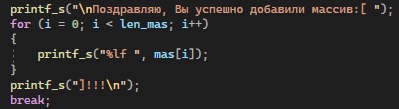
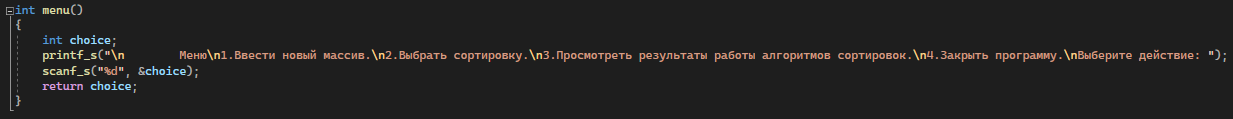


Рис. 10

* 1. Если значение menu == 2, программа выполняет проверку на наличие введенного массива, оценивая значение его длины – переменной len\_mas. Если len\_mas != 0, происходит вызов функции sort\_menu. Однако если len\_mas == 0, выводится сообщение о необходимости ввода массива и ветка действий при значении menu == 2 завершается оператором break, возвращая пользователя в главное меню. Также данная ветка действий выводит значение времени, за которое был отсортирован массив, количество сравнений и количество перестановок, потребовавшиеся во время выполнения алгоритма.
  2. Если значение menu == 3, переменная flag зануляется, бесконечный цикл while, необходимый для многократной работы с программой без ее перезапуска, завершается и функция main возвращает 0, завершая работу программы.

1. Функция int menu();

Функция menu представляет из себя основной интерфейс программы. Она выводит на экран варианты действий, доступные пользователю и считывает с клавиатуры номер команды, которую необходимо выполнить. Возвращаемое значение: номер команды.

Рис.11

1. void sort\_menu(float mas[], int len\_mas, long long int\* p\_count\_comp, long long int\* p\_count\_perm, clock\_t\* start, clock\_t\* end);

Функция sort\_menu является связующей между функциями сортировок и основным меню. Выбор одной из сортировок выполнен аналогично функции menu: программа выводит возможные для выбора варианты, считывает выбор пользователя с клавиатуры и в зависимости от введенной команды, обращается к определенной функции сортировки (Рис.12). На вход подается указатель на начало массива mas[] и длина массива len\_mas. Добавить картинку

1. Функция void swap(float mas[], int i, int j);

Данная функция меняет местами два элемента массива. На вход подается указатель на начало массива и индексы двух элементов, которые необходимо поменять местами.

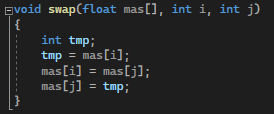


Рис.13

1. Функция void\* malloc(size\_t size);

Функция malloc является встроенной функцией в библиотеке stdlib.h и используется для выделения блока памяти не менее size байт. В данной программе используется для создания динамического массива.

1. int bubble(float mas[], int len\_mas, long long int\* p\_count\_comp, long long int\* p\_count\_perm);

Функция bubble осуществляет сортировку массива “пузырьком”. Ее алгоритм описан в главе “Методы решения”. На вход подается указатель на начало массива mas[] и размер массива len\_mas. Переменная flag отвечает за досрочное прекращение работы программы, при отсортированном массиве. По окончании работы, функция выводит сообщение об успешной сортировке массива.

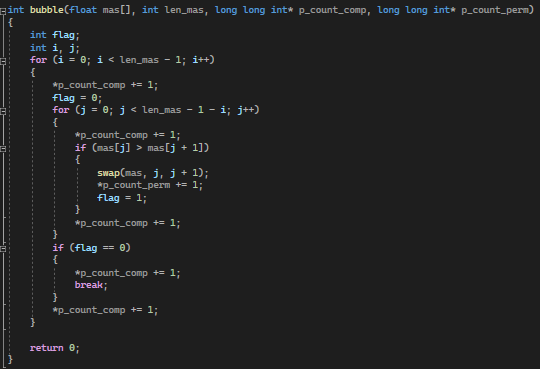


Рис.14

1. Функции void ckoarrr(float mas[], int l, int r, long long int\* p\_count\_comp, long long int\* p\_count\_perm); и int middle(float mas[], int l, int r, long long int\* p\_count\_comp, long long int\* p\_count\_perm);

Эти две функции выполняют сортировку Хоара. Ее алгоритм описан в разделе “Методы решения”. Эти функции получают на вход указатель на начало массива, левую правую границу сортируемого массива или сортируемой части.

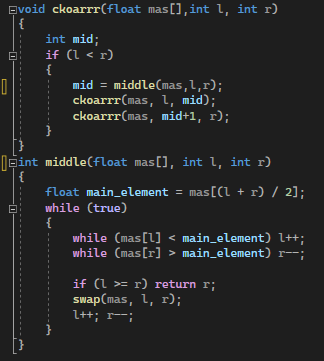


Рис. 15

1. Функция void insert\_sort(float mas[], int len\_mas, long long int\* p\_count\_comp, long long int\* p\_count\_perm);

Данная функция представляет из себя реализацию алгоритма сортировки вставкой. На вход подается указатель на начало массива mas[], его длинна len\_mas и указатели на переменные, отвечающие за подсчет количества сравнений и перестановок.

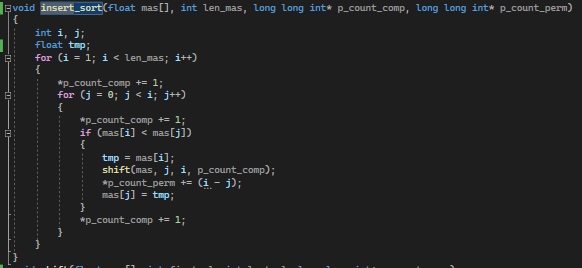


Рис.16

1. Функция void shift(float mas[], int first\_el, int last\_el, long long int\* p\_count\_comp);

Функция shift выполняет сдвиг элементов массива mas с first\_el позиции, до last\_el позиции

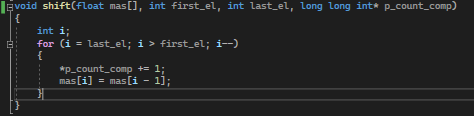
.

Рис.17

1. Функция void shell(float mas[], int len\_mas, long long int\* p\_count\_comp, long long int\* p\_count\_perm);

Функция shell выполняет алгоритм сортировки Шелла. На вход подаются: указатель на начало массива mas[], длинна массива len\_mas, два счетчика для подсчета количества сравнений, перестановок.

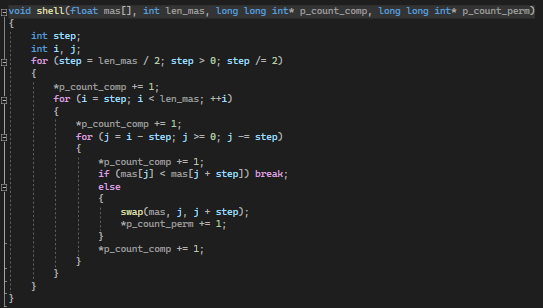


Рис. 18

1. Функция void test\_sort(float sortmas[], float unsortmas[], int len\_mas);

Данная функция проверяет корректность работы написанных алгоритмов, сравнивая два отсортированных разными способами массива.

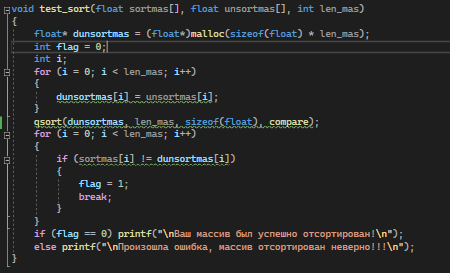


Рис. 19

# Подтверждение корректности

Для подтверждения корректности в программе была создана функция, сравнивающая массив, отсортированный готовой функцией qsort из библиотеки stdlib.h, и массив, полученный в результате работы программы. Если написанный алгоритм неверно выполнил работу, программа выводит сообщение о том, что массив был отсортирован неверно.

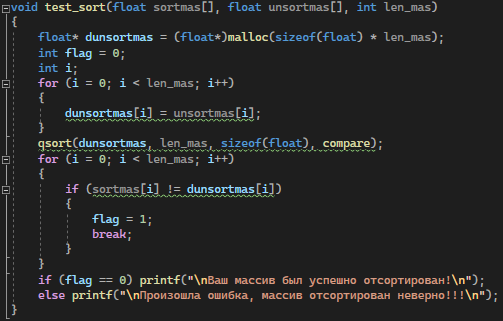


Рис. 26

# Результаты экспериментов

Для проведения эксперементов, программа была немного модифицирована, а именно изменен процесс ввода элементов массива. Теперь пользователю необходимо ввести лишь количество элементов в массиве, а сам массив заполняется при помощи встроенной функции rand().

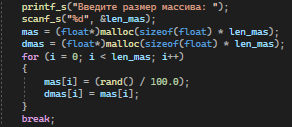


Рис. 21

Мной были проведены исследования на Рис. 22 можно видеть результаты исследования работы алгоритмов сортировок. Алгоритмы bubble, insert очень долго выполняют сортировки массивов по сравнению с ckoarrr и shell. Из Рис.2 можно видеть, что быстрее всех сортировку выполняет алгоритм Хоара.

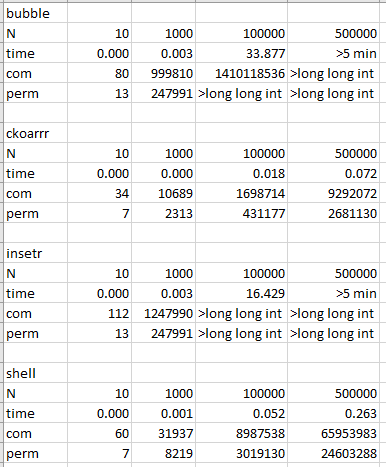


Рис.22

# Заключение

Итоговая версия программы дает возможность пользователю, при помощи примитивного интерфейса (меню), создавать массив определенной длины, сортировать его одной из четырех представленных сортировок: “пузырьком”, Хоара, вставками, Шелла (по выбору пользователя). Также проведена оценка зависимости количества операций, выполняемых каждым алгоритмом, от размера входных данных (сортируемого массива).

# Приложение

Часть кода, который вы считаете основным