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**

**«Сортировка Шелла»**

**Выполнил**:

студент группы 382003-1

Филатов Андрей Александрович

**Проверил**:

ассистент каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2020

**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

Написать на языке С сортировку Шелла (*Shell sort*).

# Метод решения

Сортировка Шелла – это модификация алгоритма сортировки вставками, сравнивающая элементы на расстоянии . Шелл предлагал использовать .

Используемый алгоритм для решения:

1. Начало.
2. **Шаг 0.**
3. **Шаг 1.** Разобьем массив на списки элементов, отстающих друг от друга на . Таких списков будет .
4. **Шаг 2.** Отсортируем элементы каждого списка.
5. **Шаг 3.** Объединим списки обратно в массив. Уменьшим *i*. Если *i*неотрицательно — вернемся к **Шаг 1.**
6. Конец.

Сложность алгоритма будет зависеть от выбора шага. Посчитаем сложность алгоритма для указанного выше шага. Пусть список содержит n элементов, тогда общее число операций равно

Таким образом, сортировка Шелла с выбранными шагами - алгоритм имеющий время в худшем случае .

Сортировка Шелла не требует дополнительной памяти для работы – дополнительная память .

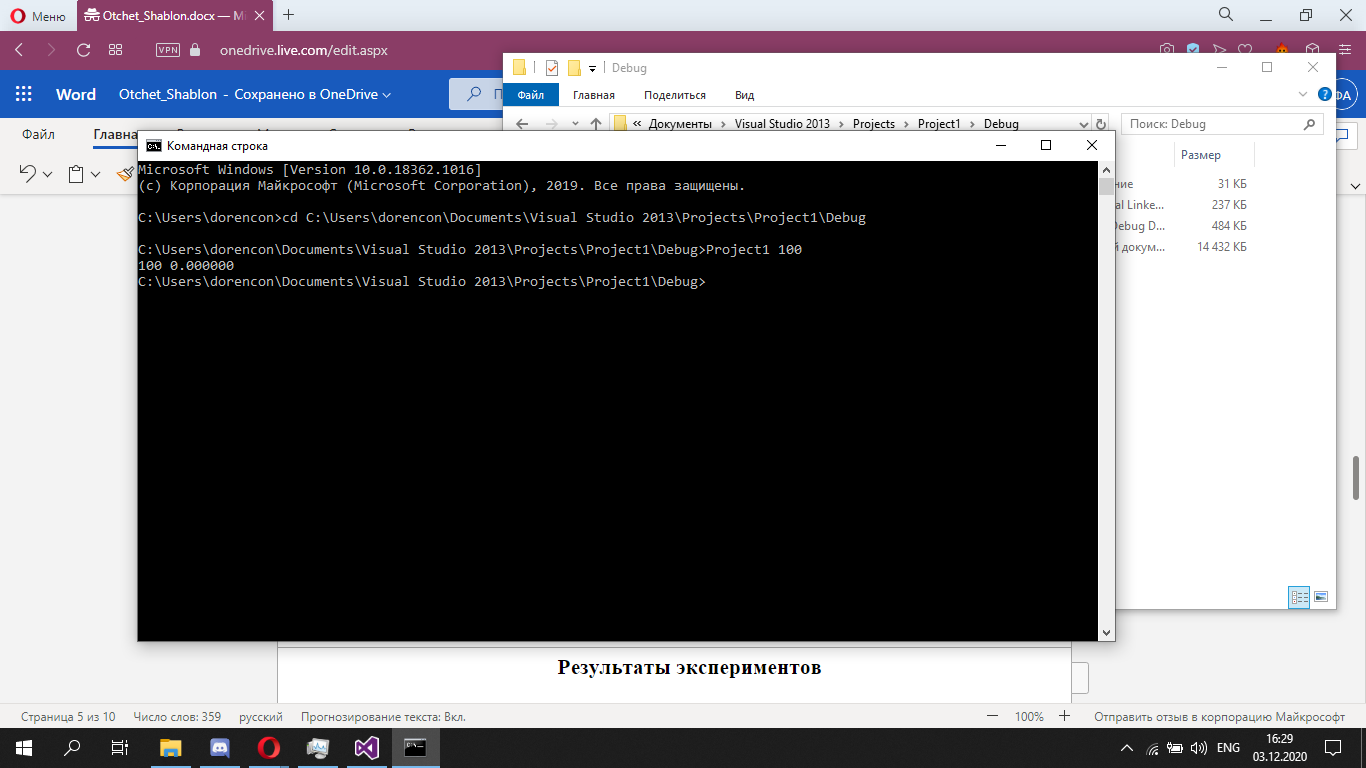
Среднее количество обменов, необходимое для выполнения сортировки массива .

# Руководство пользователя

Для использования программы пользователь должен иметь файл с именем test.txt, содержащий неотсортированный массив.

Для удобства тестирования, программа запускается из командной строки с одним аргументом. Аргумент – сколько последовательных чисел прочитать из файла.

Пример командной строки



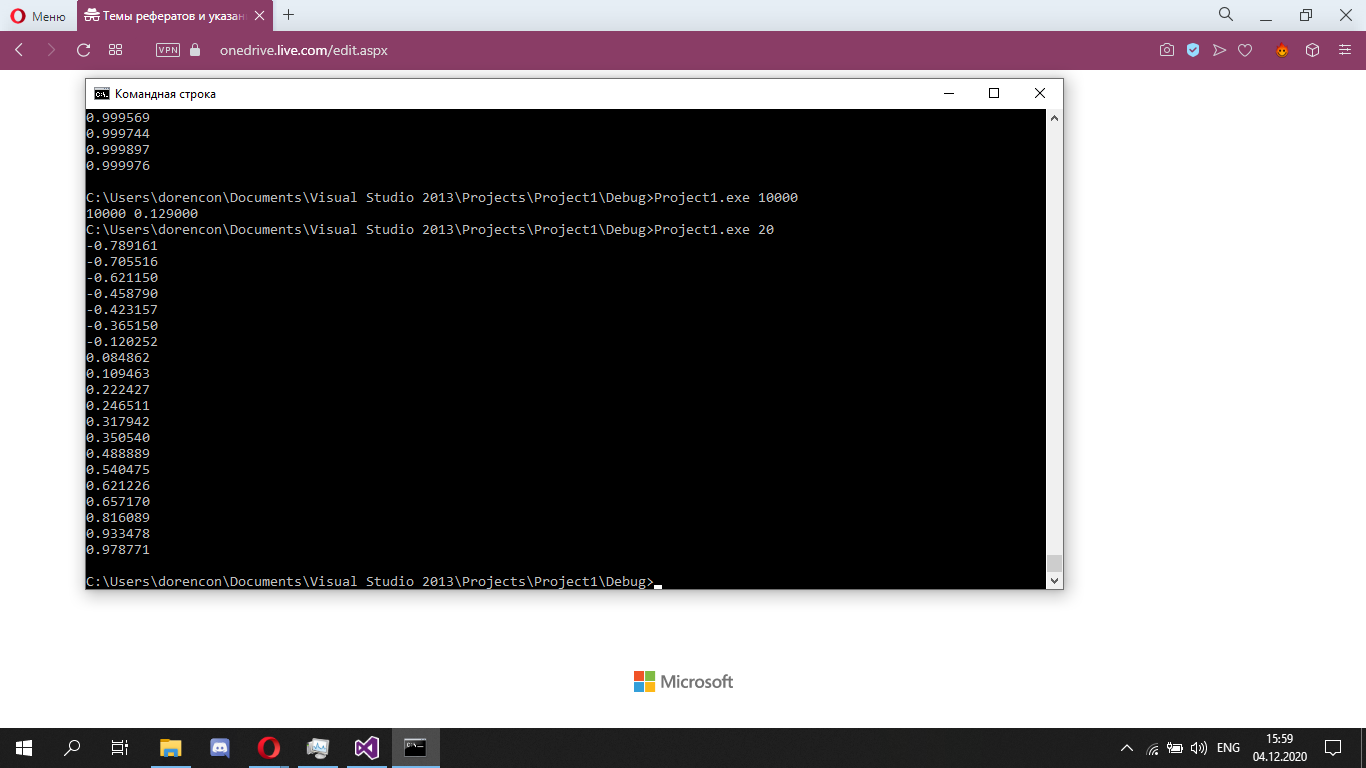
Результат работы программы – отсортированный массив, будет выведен на экран.

# Описание программной реализации

Программа состоит из одного текстового (тестового) файла и исполняемого файла.

# Подтверждение корректности

Для подтверждения корректности в программе были произведены тесты с различным количеством элементов. Пример работы программы на 20 элементах.



# Результаты экспериментов

По данным экспериментов видно, что сортировка выбором — квадратичный алгоритм, время его работы пропорционально квадрату от размера сортируемого списка.

Тесты проводились на ноутбуке Aser Intel Core i5-7200U CPU 2.50 GHz

# Заключение

Сортировка Шелла – это модифицированная сортировка вставками, зависящая от выбора шага.

Классический выбор шага, предложенный Шеллом, на некоторых массивах имеет время . Это ограничивает его применимость в реальных задачах.

Однако, современные модификации данного алгоритма улучшают его скоростные характеристики. Например, А.А. Папернов, Г.В. Стасевич предложили выбор шага как , что приводит к времени сортировки .

Дальнейшее улучшение было получено В. Праттом. Если все шаги при сортировке выбираются из множества чисел вида , то время выполнения алгоритма будет .

# Приложение

#if defined(\_WIN32) || defined(\_\_WIN32\_\_) || defined(WIN32)

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma warning(disable:4996)

#endif

#include <stdio.h>

#include <time.h>

int main(int argc, char \*\*argv)

{

int n = atoi(argv[1]);

FILE \*f;

f = fopen("test.txt", "r");

double\* a;

a = malloc(n \* sizeof(double));

for (int i = 0; i < n; i++)

{

fscanf(f, "%lf", &a[i]);

}

double temp;

int t0 = clock();

for (int s = n / 2; s > 0; s /= 2)

{

for (int i = s; i < n; i++)

{

for (int j = i - s; (j >= 0) && (a[j] > a[j + s]); j -= s)

{

temp = a[j];

a[j] = a[j + s];

a[j + s] = temp;

}

}

}

int t1 = clock();

/\*for (int i = 0; i < n; i++)

{

printf("%lf\n", a[i]);

}\*/

printf("%d %lf", n, (double)(t1 - t0) / CLOCKS\_PER\_SEC);

free(a);

return 0;

}