Федеральное агентство по образованию Российской Федерации

Государственное образовательное учреждение

высшего профессионального образования

Нижегородский государственный университет им. Н.И. Лобачевского

Институт Информационных Технологий Математики и Механики

Отчёт по лабораторной работе

Сортировки. Сравнение времени работы

Выполнил:

студент ф-та ИТММ гр. 3821Б1ПМ3

Жеребцов В.А.

Проверил:

заведующий лабораторией суперкомпьютерных технологий и высокопроизводительных вычислений

Лебедев И.Г.

Нижний Новгород

2021 г.

Содержание

[Введение](#_Введение)……………………………………………………………………………………3

[Постановка задачи](#_Постановка_задачи)…………………………………………………………………………4

[Руководство пользователя](#_Руководство_пользователя)...………………………………………………………………5

[Руководство программиста](#_Руководство_программиста)..………………………………………………………………7

[Описание структуры программы](#_Описание_структуры_программы)…………………………………………………………7

[Описание структур данных](#_Описание_структур_данных)….……….…………………………………………………....9

[Описание алгоритмов](#_Описание_алгоритмов)...…………………………………………………………………..10

[Эксперименты](#_Эксперименты_1)…………………………………………………………………………….16

[Заключение](#_Заключение)………………………………………………………………………………..19

[Литература](#_Литература)………………………………………………………………………………...19

[Приложения](#_Приложения)……………………………………………………………………………….20

[Приложение 1](#_Приложение_1)……………………………………………………………………..21

[Приложение 2](#_}Приложение_2)……………………………………………………………………..21

# Введение

Сортировка данных – это очень нужная вещь в работе с какими-либо данными в различных проектах. В своей работе я применил 3 метода сортировок к массиву из различных случайных чисел, записанных в файле и сравнил время их работы.

# Постановка задачи

Реализовать сортировки массивов данных (тип данных определяется преподавателем) задаваемых: обязательно случайно, дополнительно с клавиатуры или из файла.

Реализовать сортировки: пузырьком, вставкой, быстрая. Сравнить время работы, сделать выводы.

Первая программа создает текстовый файл с записанными в него числами. Программа принимает количество чисел n, максимальное и минимальное значение.

Вторая программа читает текстовый файл с набором чисел, выводит консольный интерфейс (печать, сортировка, сброс, выход), выполняет выбранные действия.

# Руководство пользователя

***Первая программа:***

Начинать работу нужно с первой программы, так как в ней мы будем генерировать числа в файл. При её запуске пользователь увидит окно, в котором ему необходимо ввести данные, нужные для дальнейшей работы (рис.1):



Рис. 1. Первый запуск первой программы.

После ввода всех данных пользователь увидит сообщение, что числа успешно сгенерированы (рис.2). Это значит, что нужно переходить ко второй программе:

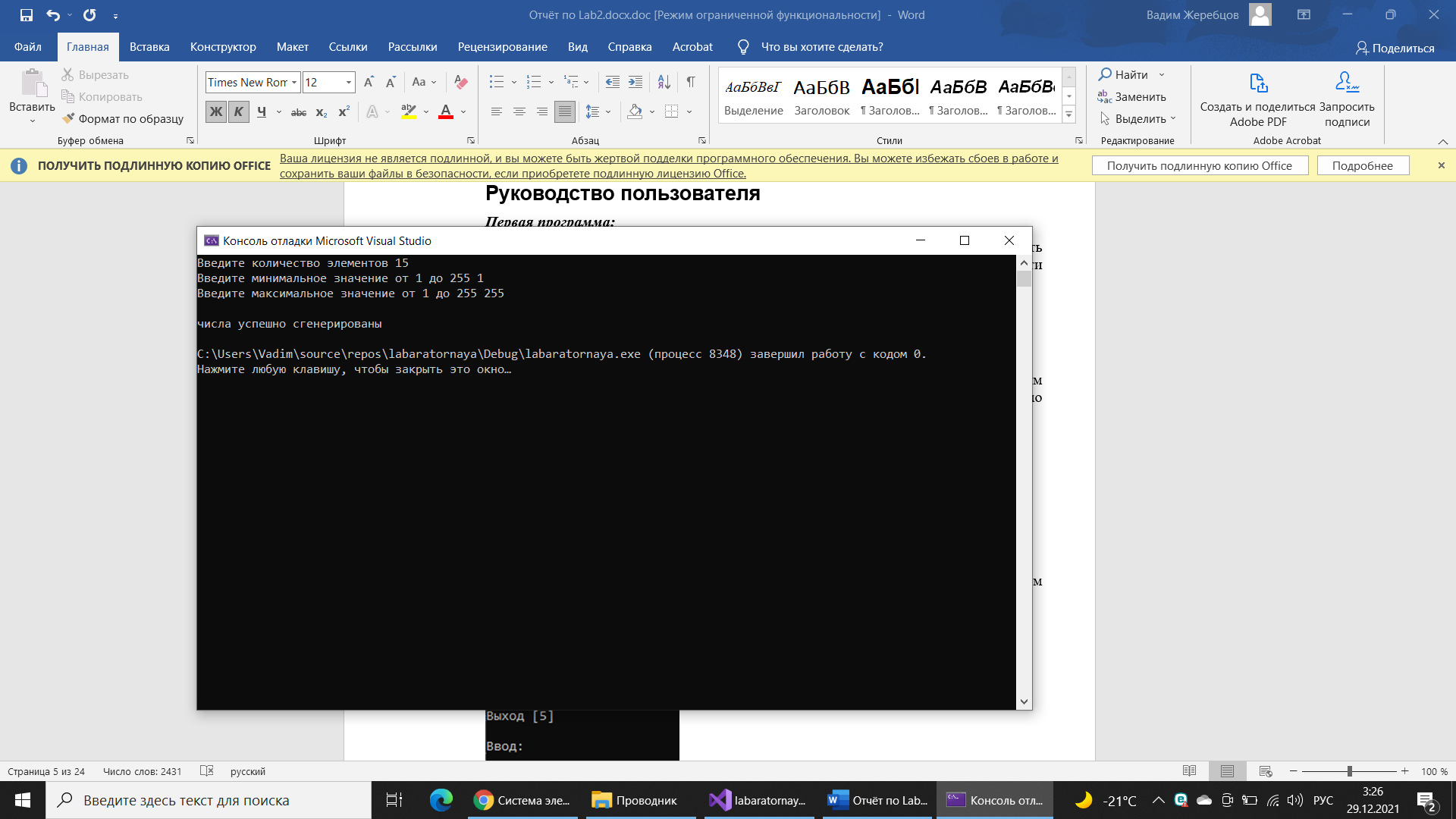


Рис. 2. Успешная работа первой программы.

***Вторая программа:***

При запуске второй программы мы увидим её главное меню, в котором сможем выбрать её дальнейшие действия:

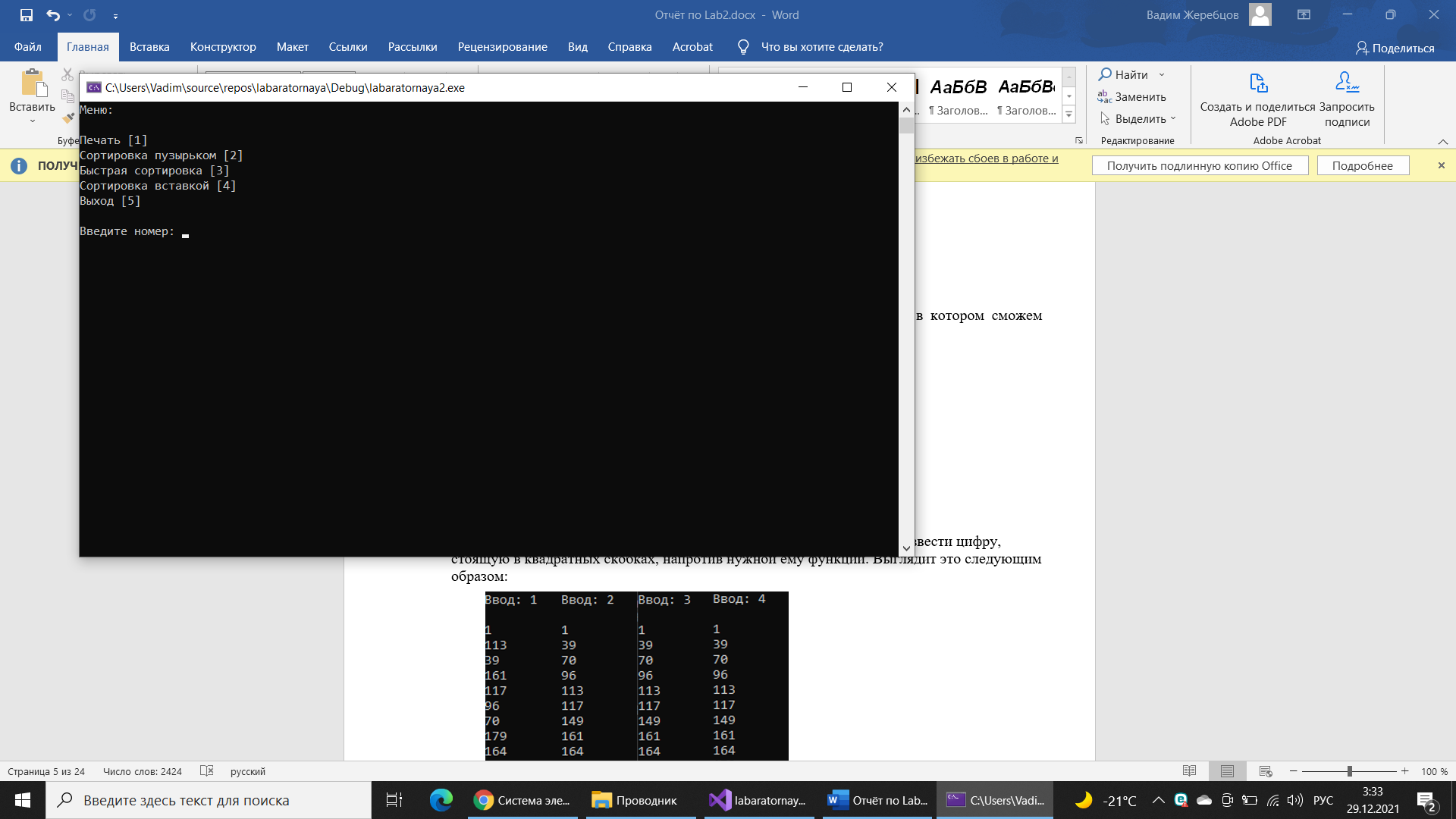


Рис. 3. Первый запуск второй программы.

Для выбора любого из пунктов меню, пользователю необходимо ввести цифру, стоящую в квадратных скобках, напротив нужной ему функции. Выглядит это следующим образом:

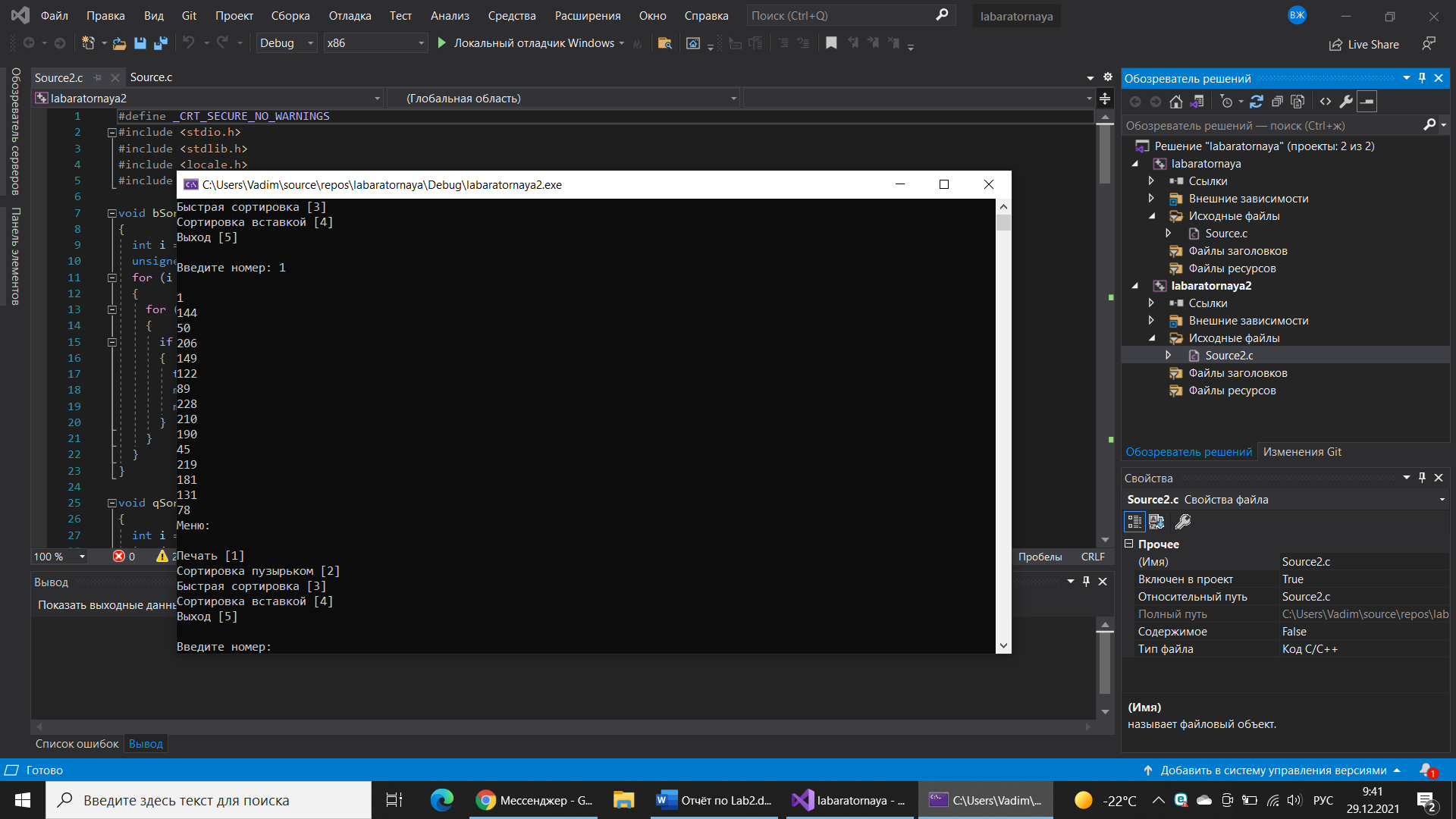
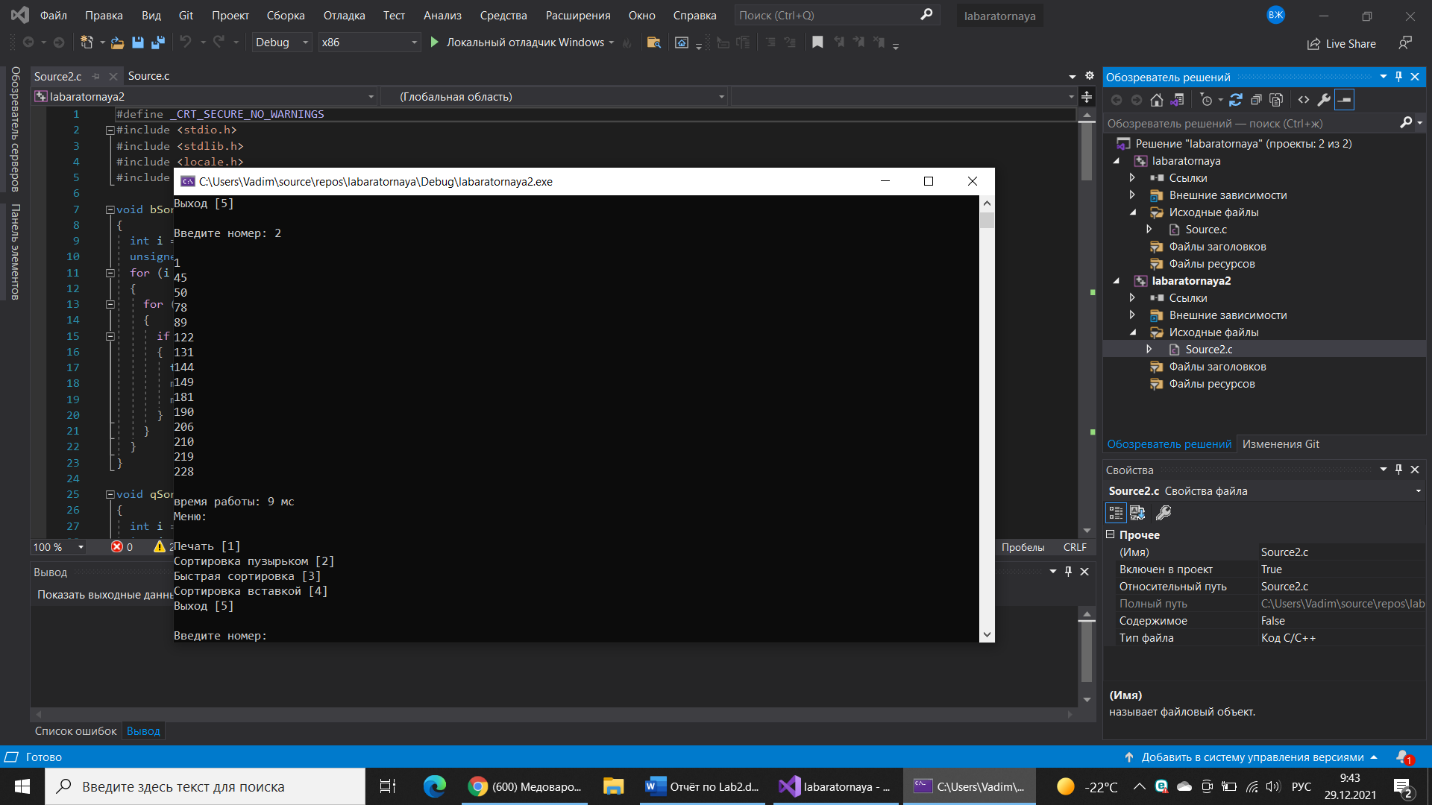
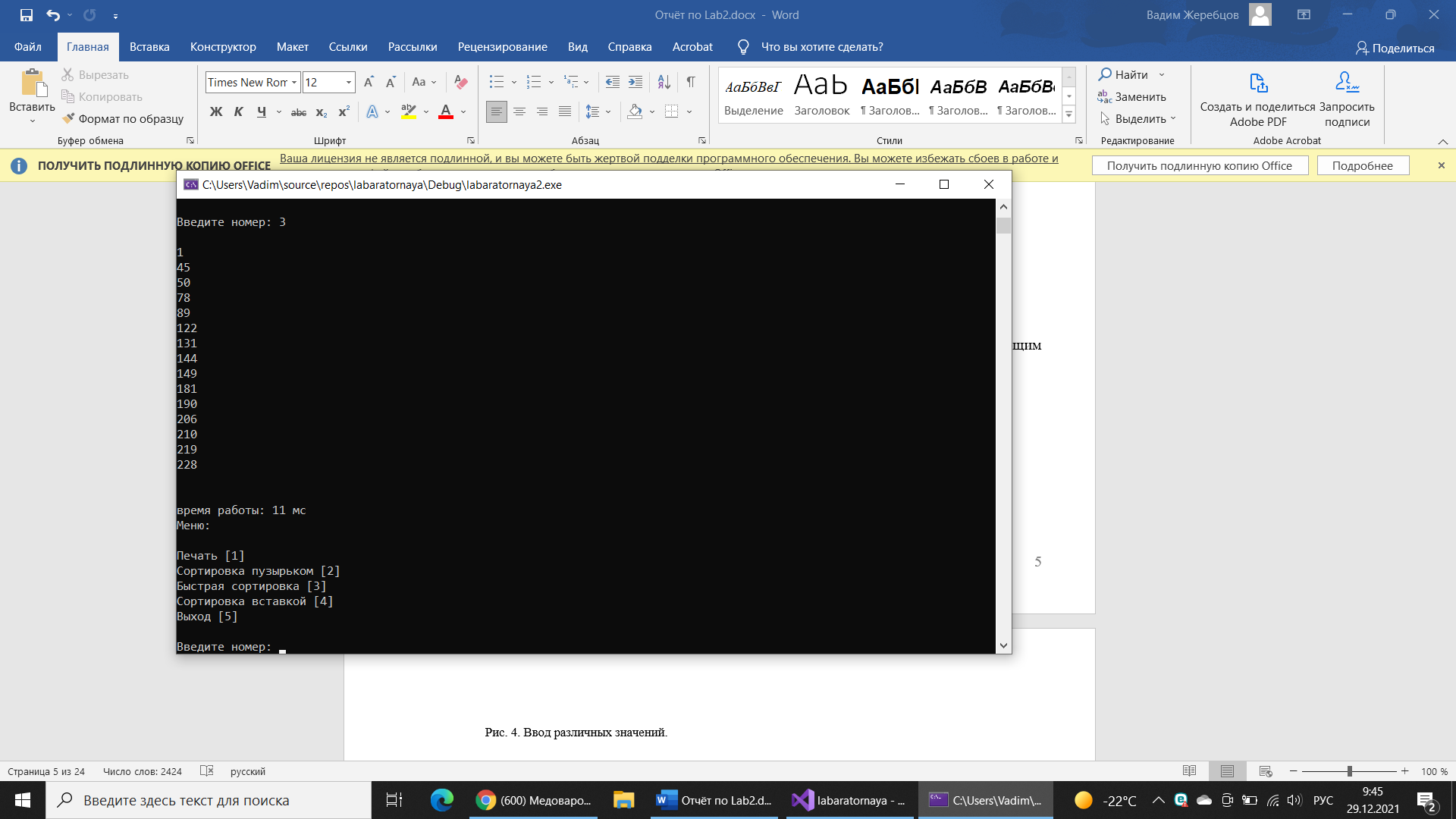
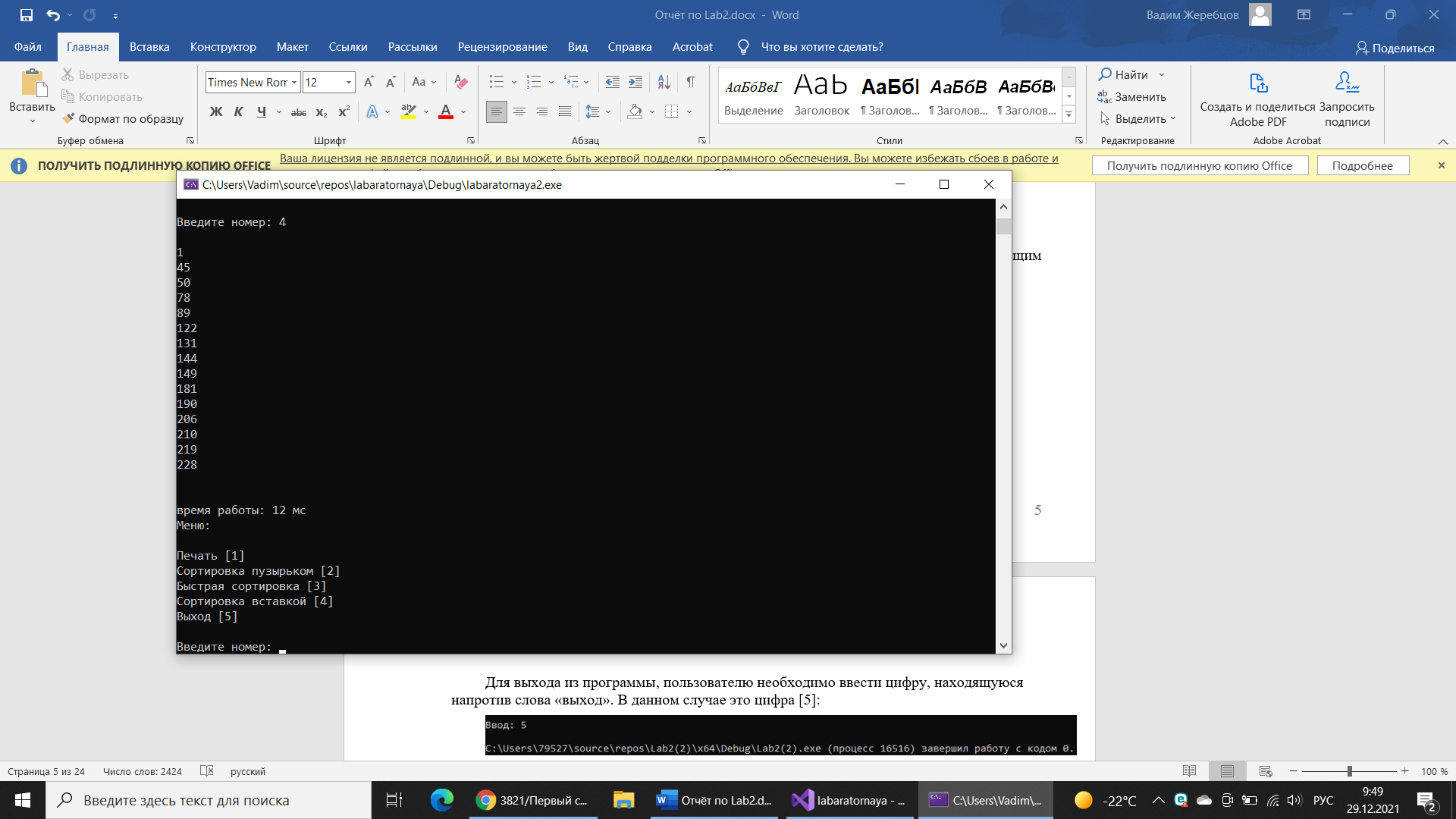


Рис. 4. Ввод различных значений.

Для выхода из программы, пользователю необходимо ввести цифру, находящуюся напротив слова «выход» (рис. 5). В данном случае это цифра «5»:

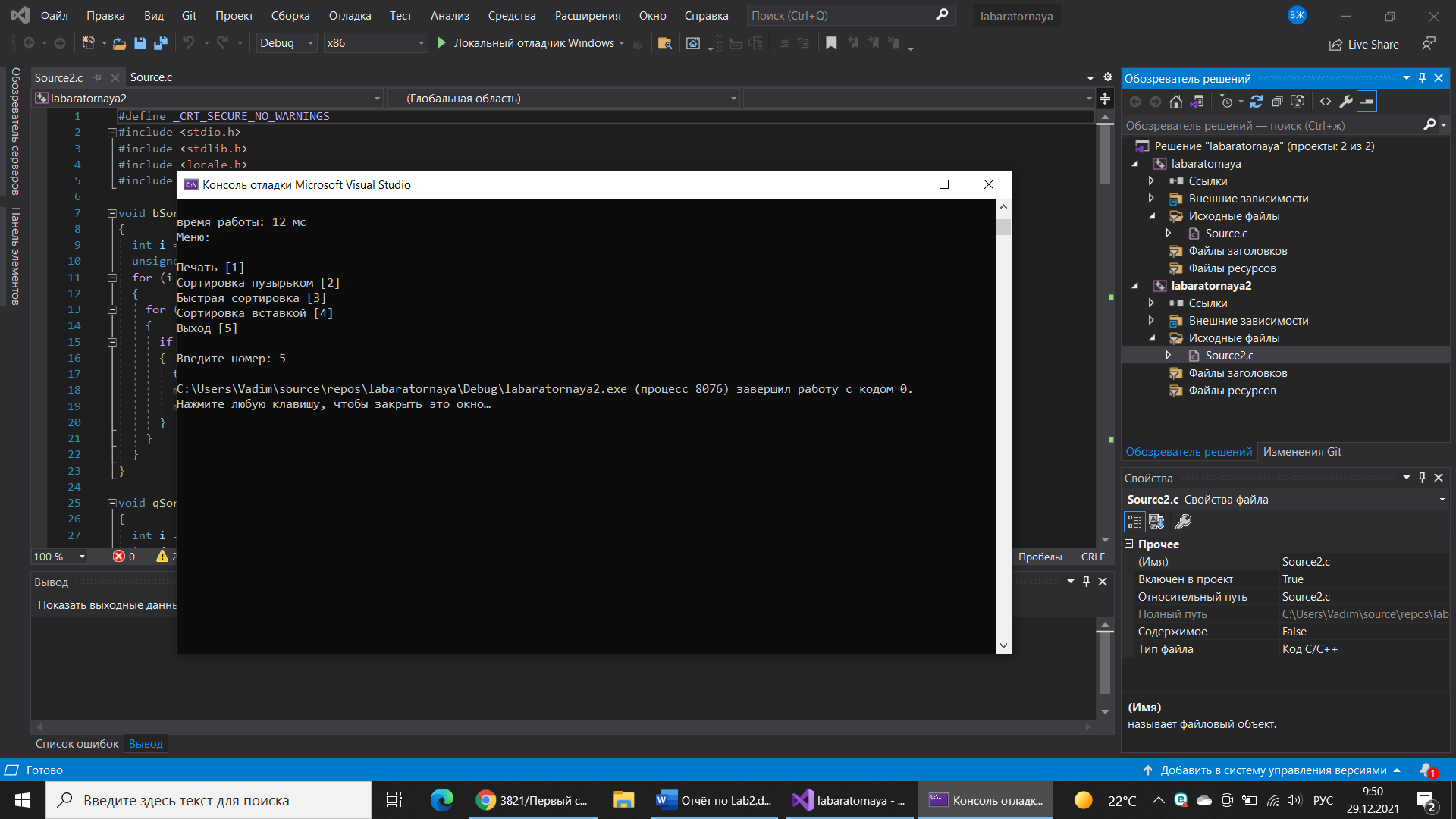


Рис. 5. Выход.

Так же во время выполнений каждой из сортировок программа замеряет время работы и показывает его пользователю (рис.6):



Рис 6. Время работы.

# 

# Руководство программиста

## Описание структуры программы

***Первая программа состоит из следующих модулей:***

1. Подключение библиотек и констант.

Фрагмент 1:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

1. Русификация и объявление переменных.

Фрагмент кода 2:

setlocale(LC\_ALL, "Russian");

int i = 0, n = 0, min = 0, max = 0;

unsigned char\* mas;

1. Взаимодействие с пользователем.

Фрагмент кода 3:

printf("Введите количество чисел ");

scanf("%d", &n);

printf("Введите минимальное число диапазона от 1 до 255 ");

scanf("%d", &min);

printf("Введите максимальное число диапазона от 1 до 255 ");

scanf("%d", &max);

printf("\n");

if ((max <= min) || (max > 255) || (min > 255) || (min <= 0) || (n <= 0) || (max <= 0))

{

printf("Введено неверное значение\n");

return 0;

}

1. Создание массива и помещение в него случайных чисел.

Фрагмент кода 4:

mas = (unsigned char\*)malloc(n \* sizeof(unsigned char));

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

{

mas[i] = (((((double)rand()) / RAND\_MAX) \* (max - min)) + min);

}

1. Обращение к файлу и запись в него чисел из массива.

Фрагмент кода 5:

FILE\* file;

file = fopen("numbers.txt", "w+");

fprintf(file, "%d\n", n);

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

{

fprintf(file, "%d\n", mas[i]);

}

1. Очистка памяти и завершение работы программы.

Фрагмент кода 6:

free(mas);

fclose(file);

***Вторая программа состоит из следующих модулей:***

1. Подключение библиотек и констант.

Фрагмент кода 7:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <time.h>

1. Три функции, выполняющие сортировки

Фрагмент кода 8:

void bSort(unsigned char\* mas, int n)

void qSort(unsigned char\* mas, int n)

void iSort(unsigned char\* mas, int n)

1. Локализация и объявление переменных.

Фрагмент кода 9:

setlocale(LC\_ALL, "Russian");

int n = 0;

int i = 0;

unsigned char\* mas;

int num = 0;

1. Обращение к файлу, создание массива и считывание в него чисел из файла.

Фрагмент кода 10:

FILE\* file;

file = fopen("C:/Users/Vadim/source/repos/labaratornaya/labaratornaya/numbers.txt", "r+");

fscanf\_s(file, "%d", &n);

mas = (unsigned char\*)malloc(n \* sizeof(unsigned char));

for (i = 1; i < n+1; i++)

{

fscanf\_s(file, "%u\n", &mas[i-1]);

}

1. Главное меню и реализация его действий.

Фрагмент кода 11:

while (num != 5)

{

switch (num)

{

case 1:

printf("\n");

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

{

printf("%u\n", mas[i]);

}

break;

case 2:

printf("\n");

unsigned int start1 = clock();

bSort(mas, n);

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

{

printf("%u\n", mas[i]);

}

unsigned int end1 = clock();

unsigned int time1 = end1 - start1;

printf("\nвремя работы: %u мс\n", time1);

break;

case 3:

printf("\n");

unsigned int start2 = clock();

qSort(mas, n);

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

{

printf("%u\n", mas[i]);

}

unsigned int end2 = clock();

unsigned int time2 = end2 - start2;

printf("\n\nвремя работы: %u мс\n", time2);

break;

case 4:

printf("\n");

unsigned int start3 = clock();

iSort(mas, n);

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

{

printf("%u\n", mas[i]);

}

unsigned int end3 = clock();

unsigned int time3 = end3 - start3;

printf("\n\nвремя работы: %u мс\n", time3);

break;

}

fclose(file);

file = fopen("C:/Users/Vadim/source/repos/labaratornaya/labaratornaya/numbers.txt", "r+");

fscanf\_s(file, "%d", &n);

for (i = 1; i < n + 1; i++)

{

fscanf\_s(file, "%u\n", &mas[i - 1]);

}

printf("Меню:\n");

printf("\n");

printf("Печать [1]\n");

printf("Сортировка пузырьком [2]\n");

printf("Быстрая сортировка [3]\n");

printf("Сортировка вставкой [4]\n");

printf("Выход [5]\n\n");

printf("Введите номер: ");

scanf\_s("%d", &num);

}

1. Очистка памяти и завершение работы программы.

Фрагмент кода 12:

free(mas);

fclose(file);

return 0;

## Описание структур данных

## Первая программа:

В программе используются следующие типы данных:

1. Счётчик цикла «i» имеет тип int, так как является счётчиком и должен принимать значения целых чисел.
2. Переменная n – количество случайных чисел, тип int.
3. Переменные «min» и «max» имеют тип int.
4. Указатель «\*mas» имеет тип unsigned char и хранит в себя случайно сгенерированные числа.
5. «file» - это тот файл, в который мы записываем наши случайные числа.

## Вторая программа:

В программе используются следующие типы данных:

1. Счётчики циклов «j» и «i» имеют тип int, так как являются счётчиками и должны принимать значения целых чисел.
2. Unsigned char temp – переменная, временно сохраняющая в себя число из массива.
3. unsigned int end = clock() и unsigned int start = clock() используются для измерения времени работы сортировки.
4. Переменная n – количество случайных чисел. Тип int.
5. «file» - это тот файл, в который мы записываем наши случайные числа.
6. Указатель «\*mas» имеет тип unsigned char и сохраняет в себя случайно сгенерированные числа.

## Описание алгоритмов

# Первым алгоритмом в коде является сортировка пузырьком. Суть сортировки заключается в том, чтобы проверить является ли проверяемый элемент больше последующего, если да, то мы меняем их местами, если нет, то оставляем всё как есть и проверяем следующий элемент.

Фрагмент кода 14:

void bSort(unsigned char\* mas, int n)

{

int i = 0, j = 0;

unsigned char temp = 0;

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

{

for (j = 0; j < n - i - 1; j++)

{

if (mas[j] > mas[j + 1])

{

temp = mas[j];

mas[j] = mas[j + 1];

mas[j + 1] = temp;

}

}

}

}

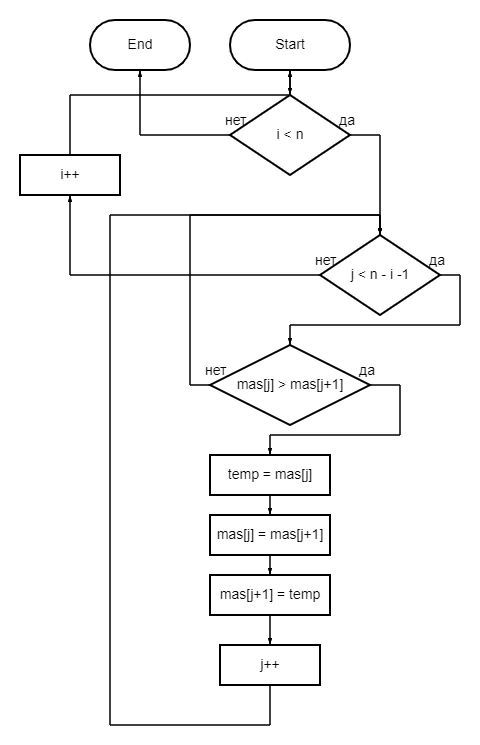


Рис. 7. Блок-схема к алгоритму сортировки пузырьком.

# Второй алгоритм – это быстрая сортировка. Мы выбираем центральный элемент массива. Сравниваем с центральным элементы справа и слева от него. Элементы слева, большие или равные ему, перекидываем в правую часть, и, аналогично, элементы справа, меньшие или равные, перекидываем в левую часть. В результате слева от центрального останутся меньшие или равные ему элементы, а справа, большие или равные. Рекурсивно повторяем действие для левой и правой части, относительно опорного элемента массива.

Фрагмент кода 15:

void qSort(unsigned char\* mas, int n)

{

int i = 0;

int j = n - 1;

int tmp = 0;

int c = mas[n / 2];

do

{

while (mas[i] < c) i++;

while (mas[j] > c) j--;

if (i <= j)

{

tmp = mas[i];

mas[i] = mas[j];

mas[j] = tmp;

i++;

j--;

}

} while (i <= j);

if (j > 0) qSort(mas, j + 1);

if (i < n) qSort(&mas[i], n - i);

}

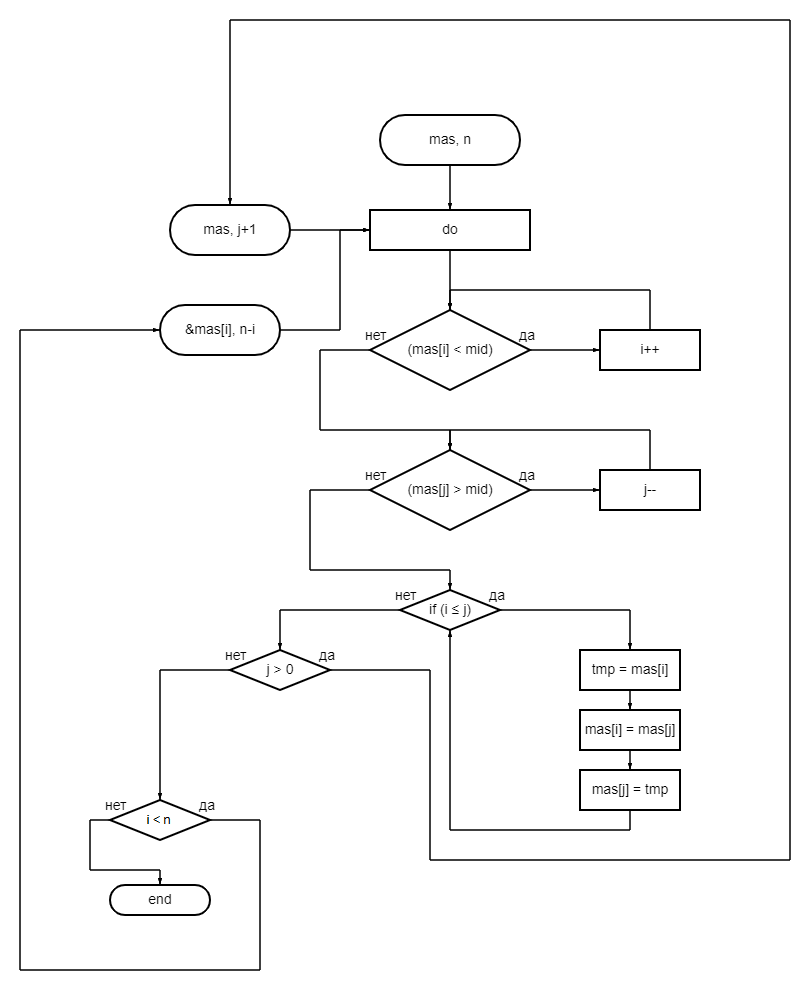


Рис. 8. Блок-схема к алгоритму быстрой сортировки.

1. Сортировка вставкой. В данном алгоритме мы рассматриваем все элементы массива циклом «for» и сразу подставляем их на нужное место в ряду. Мы присваиваем элемент массива временной переменной и проверяем, не оказалась ли она меньше предыдущего элемента массива с помощью цикла «while». В случае, если она оказалась меньше, мы меняем местами исходный элемент, и тот, который оказался больше него, и продолжаем проверять. Когда временная переменная окажется больше или равна предыдущему элементу, мы заканчиваем наш цикл «while» и рассматриваем следующий элемент в цикле «for».

Фрагмент кода 16:

void iSort(unsigned char\* mas, int n)

{

unsigned char temp = 0;

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

{

temp = mas[i];

int j = i;

while ((j > 0) && (mas[j - 1] > temp))

{

mas[j] = mas[j - 1];

j--;

}

mas[j] = temp;

}

}

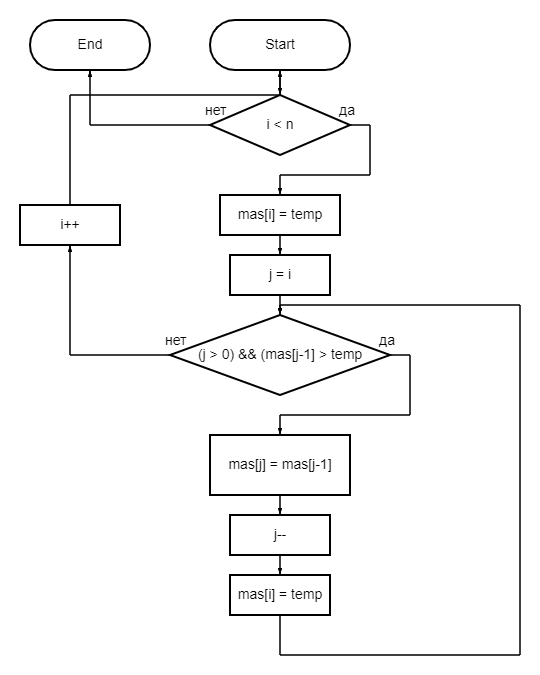


Рис. 9. Блок-схема к алгоритму сортировки вставкой.

# Эксперименты

1. В качестве первого эксперимента, я введу в первую программу 50 чисел в диапазоне от 1 до 200:

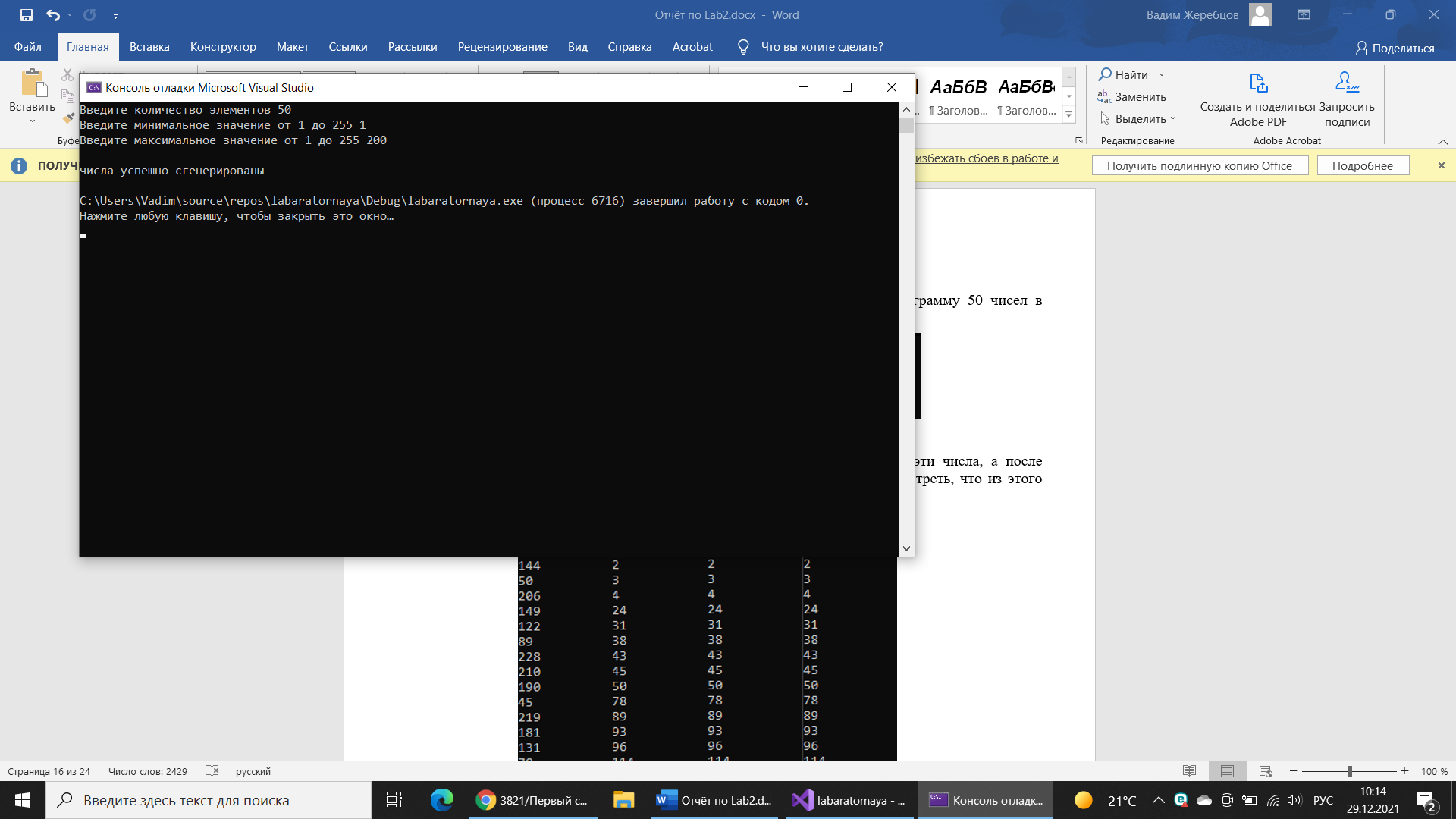


Рис. 10. Вводим данные.

Далее я запущу вторую программу и попробуем отсортировать их тремя разными способами и замерять время:

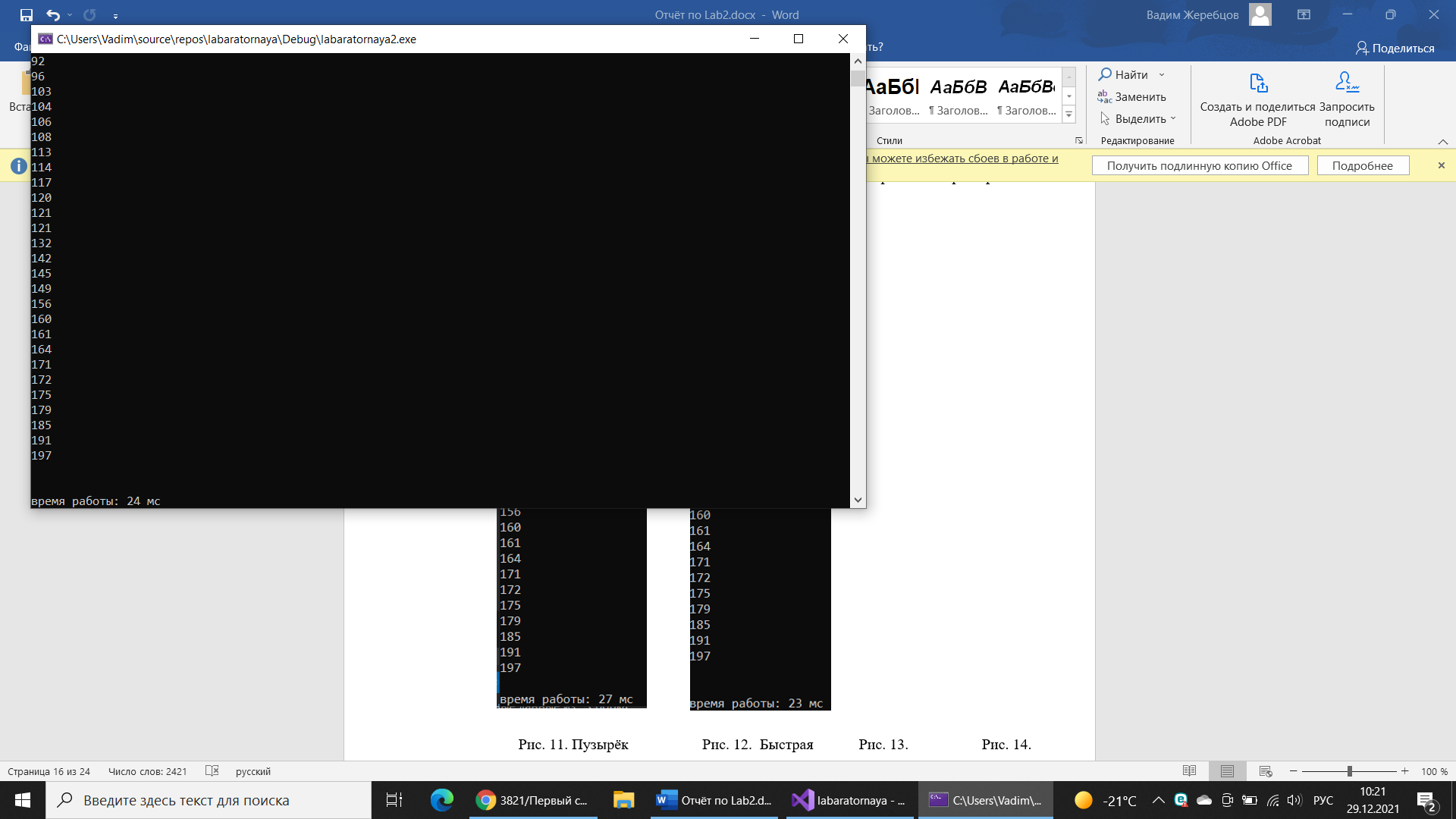
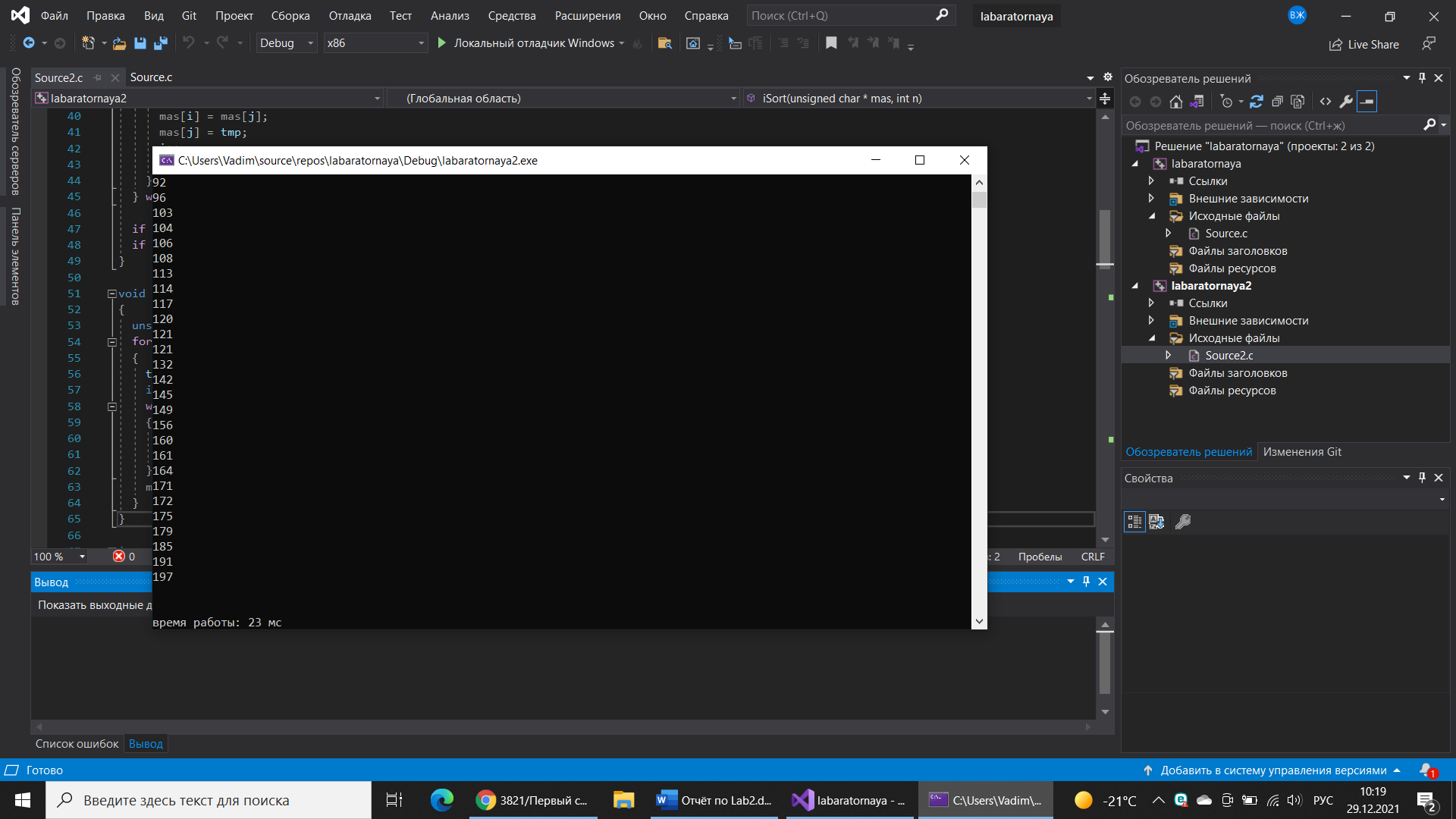
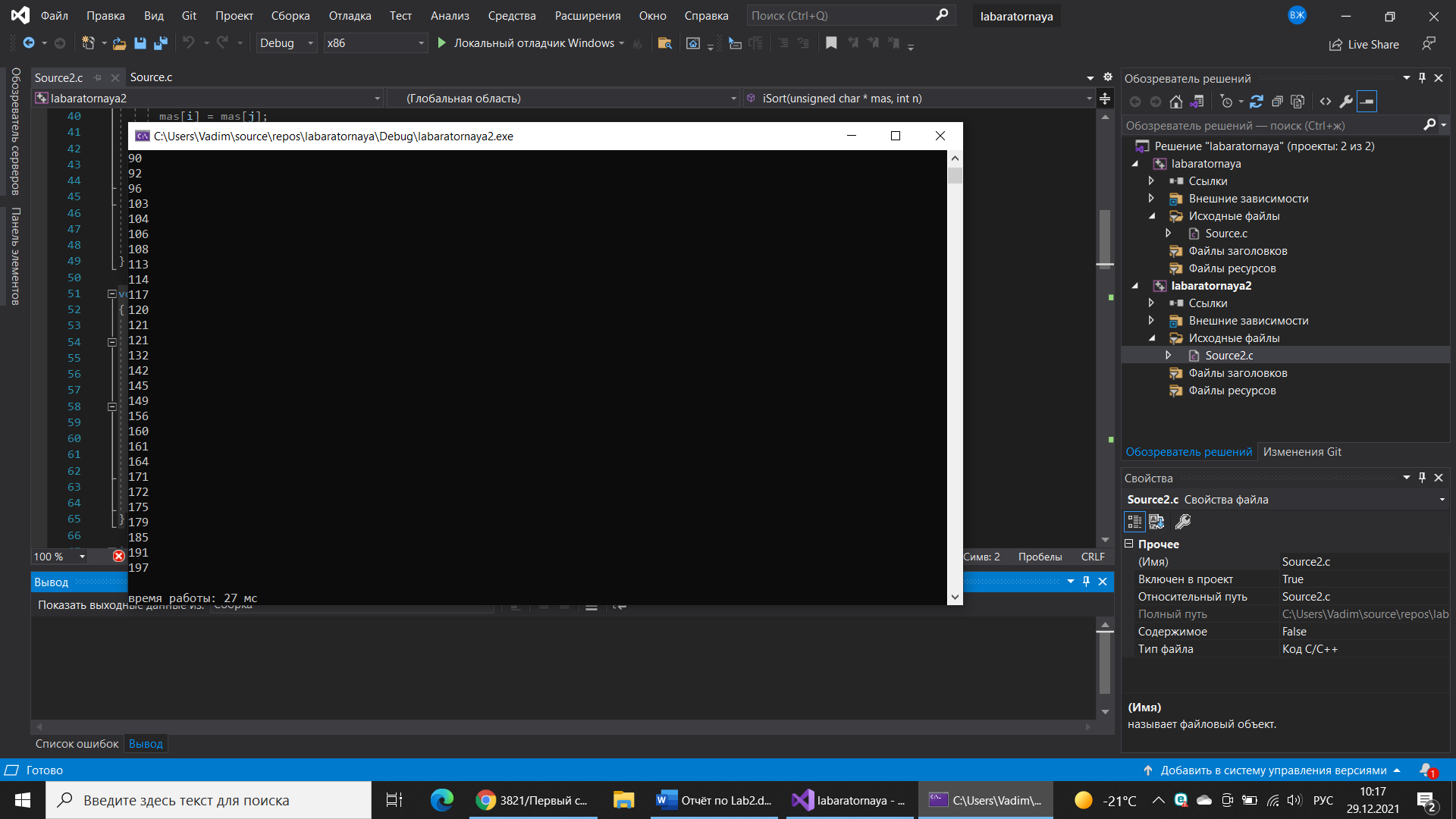


Рис. 11. Пузырёк Рис. 12. Быстрая Рис. 13. Вставки

Как мы можем увидеть, каждая из сортировок отлично справилась со своей задачей.

1. Эксперимент с проверка первой программы на корректно введенные данные:

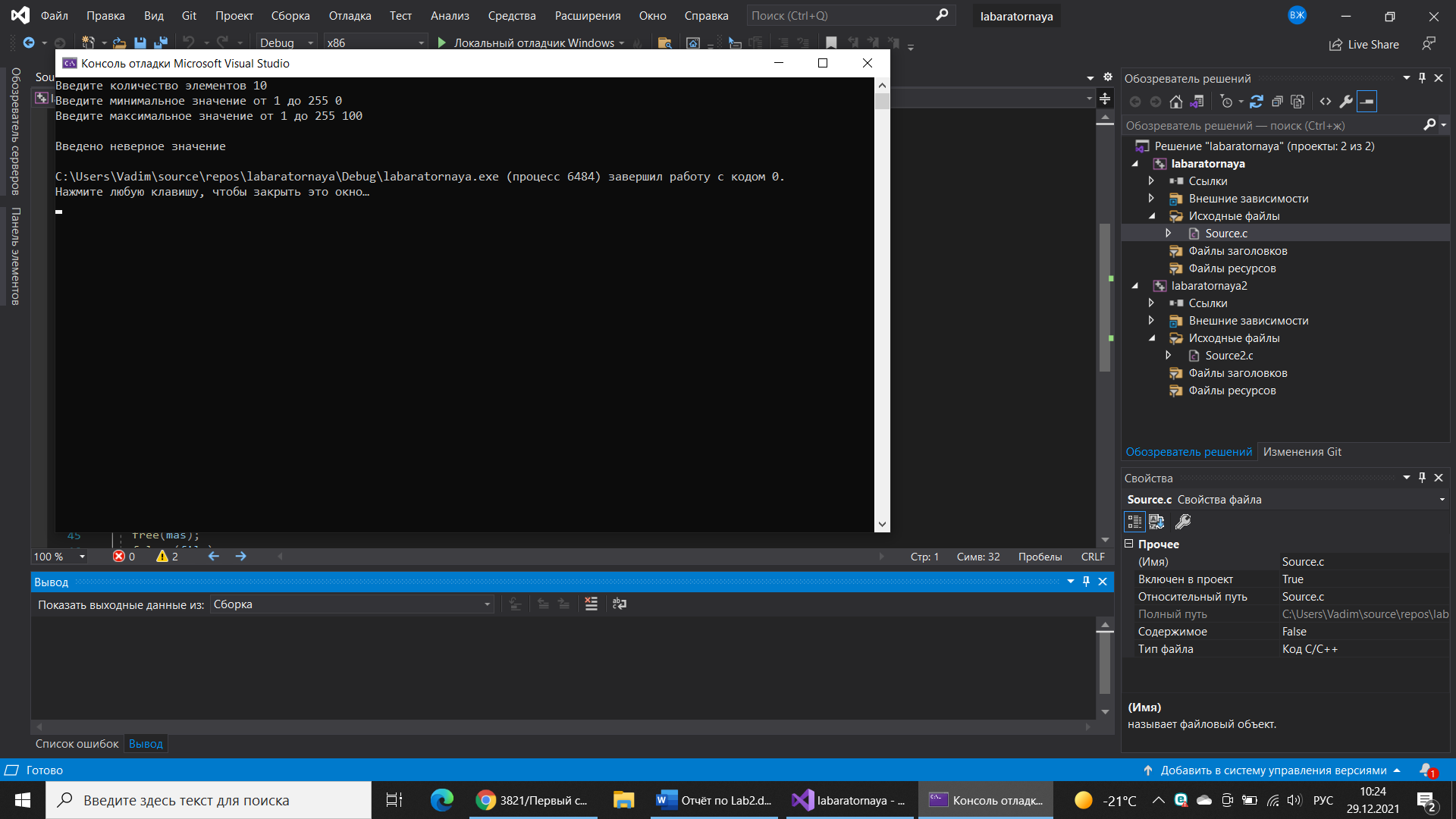


Рис. 15. Вводим данные.

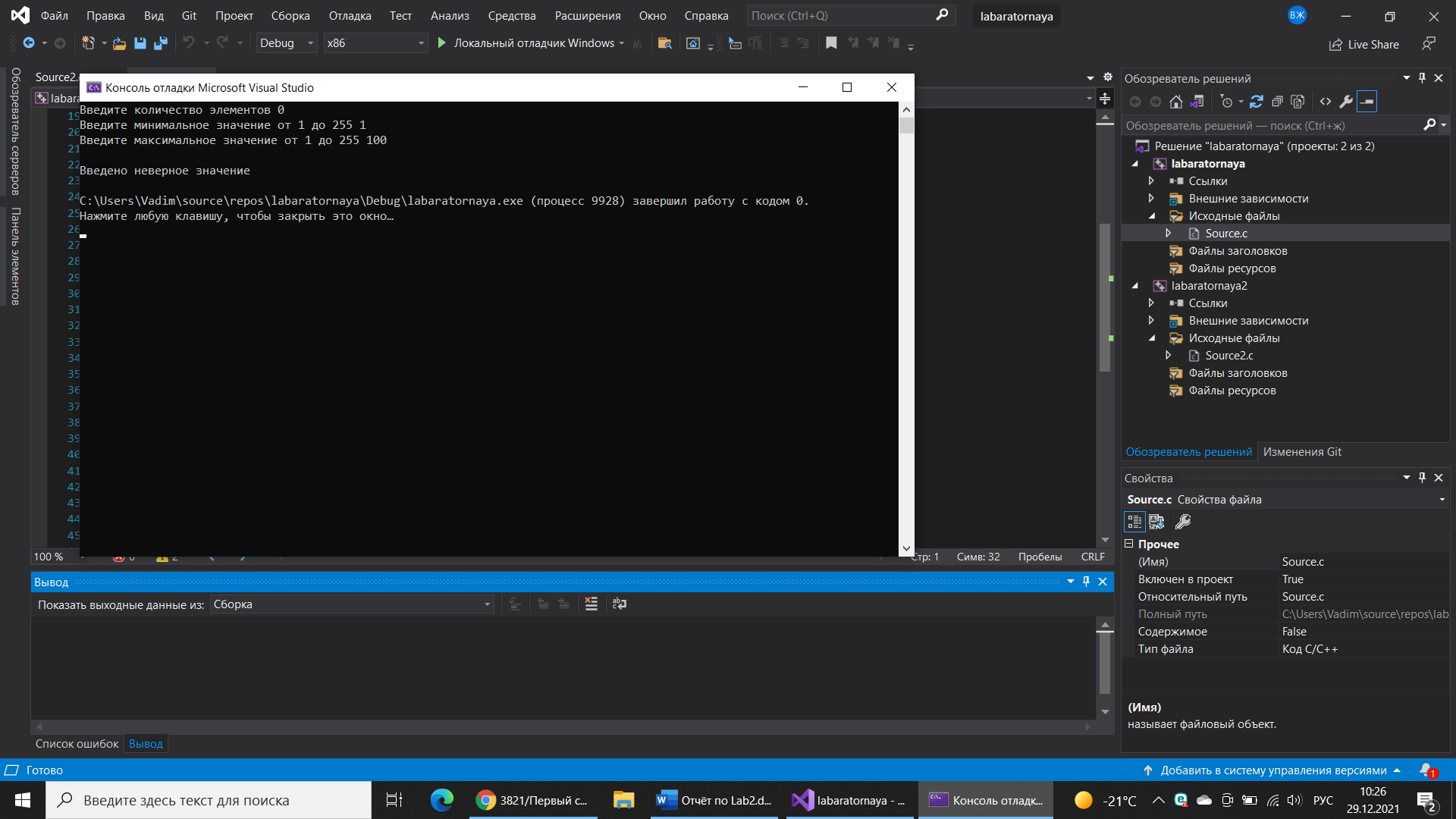


Рис. 16. Вводим данные.

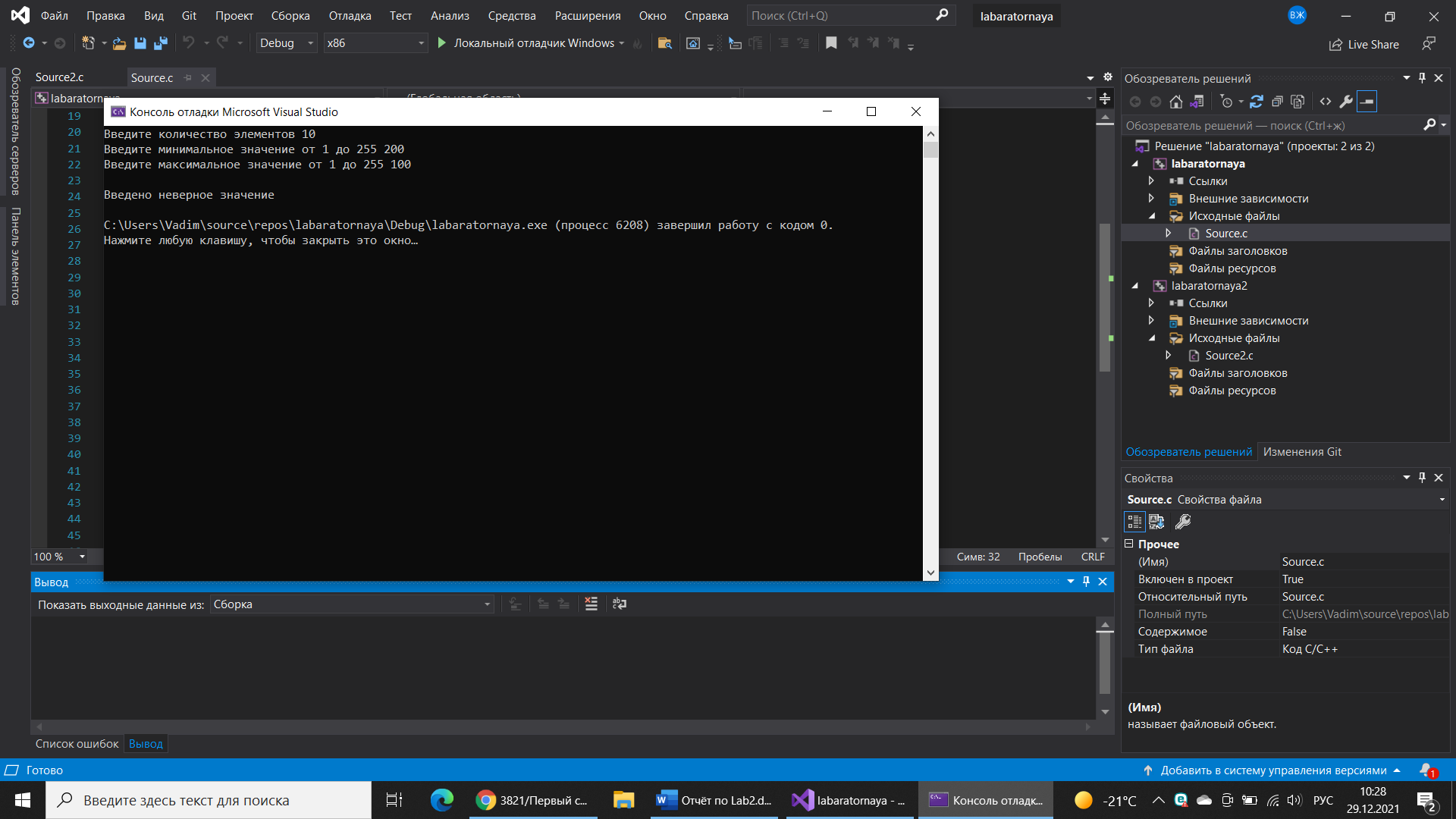


Рис. 17. Вводим данные.

Как мы можем заметить, с этим программа так же отлично справляется и сообщает нам о том, что данные некорректны.

1. Следующий эксперимент будет связан с временем работы. Проведем несколько исследований. Возьмем диапазон чисел от 1 до 255

|  |  |  |  |
| --- | --- | --- | --- |
| количество элементов, n | Пузырёк  время работы (мс) | Быстрая  время работы (мс) | Вставки  время работы (мс) |
| 10 | 2 | 2 | 2 |
| 100 | 51 | 49 | 49 |
| 1000 | 572 | 556 | 560 |
| 10000 | 5632 | 1239 | 1334 |
| 100000 | 49283 | 12143 | 20743 |
| 1000000 |  | 135371 |  |

табл.1. Замеры времени

# Заключение

По итогу этой лабораторной работы, я научился работать с файлами, узнал два новых для себя вида сортировки, смог написать код для каждой из них. Также я замерил время работы каждой из сортировок, выяснил, что самой оперативной является быстрая сортировка, ведь она является усовершенствованным видом сортировки пузырьком. Сортировка вставкой нуждается в более сложных математических расчётах, поэтому так же не обладает высокой скоростью.

# Литература

1. Язык «С» / Б.В. Кёрниган, Д.М. Ричи – Англия, 1978 год - 299 страниц.
2. Язык «С» в 21-м веке / Бен Клименс - Австралия, 2017 год - 376 страниц.

# Приложения

## Приложение 1

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "Russian");

int i = 0, n = 0, min = 0, max = 0;

unsigned char\* mas;

printf("Введите количество элементов ");

scanf\_s("%d", &n);

printf("Введите минимальное значение от 1 до 255 ");

scanf\_s("%d", &min);

printf("Введите максимальное значение от 1 до 255 ");

scanf\_s("%d", &max);

printf("\n");

if ((max <= min) || (max > 255) || (min > 255) || (min <= 0) || (n <= 0) || (max <= 0))

{

printf("Введено неверное значение\n");

return 0;

}

mas = (unsigned char\*)malloc(n \* sizeof(unsigned char));

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

{

mas[i] = (((((double)rand()) / RAND\_MAX) \* (max - min)) + min);

}

FILE\* file;

file = fopen("numbers.txt", "w+");

fprintf(file, "%d\n", n);

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

{

fprintf(file, "%d\n", mas[i]);

}

printf("числа успешно сгенерированы\n ");

free(mas);

fclose(file);

return 0;

}

## Приложение 2

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <time.h>

void bSort(unsigned char\* mas, int n)

{

int i = 0, j = 0;

unsigned char temp = 0;

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

{

for (j = 0; j < n - i - 1; j++)

{

if (mas[j] > mas[j + 1])

{

temp = mas[j];

mas[j] = mas[j + 1];

mas[j + 1] = temp;

}

}

}

}

void qSort(unsigned char\* mas, int n)

{

int i = 0;

int j = n - 1;

int tmp = 0;

int c = mas[n / 2];

do

{

while (mas[i] < c) i++;

while (mas[j] > c) j--;

if (i <= j)

{

tmp = mas[i];

mas[i] = mas[j];

mas[j] = tmp;

i++;

j--;

}

} while (i <= j);

if (j > 0) qSort(mas, j + 1);

if (i < n) qSort(&mas[i], n - i);

}

void iSort(unsigned char\* mas, int n)

{

unsigned char temp = 0;

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

{

temp = mas[i];

int j = i;

while ((j > 0) && (mas[j - 1] > temp))

{

mas[j] = mas[j - 1];

j--;

}

mas[j] = temp;

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

int n = 0;

int i = 0;

unsigned char\* mas;

int num = 0;

FILE\* file;

file = fopen("C:/Users/Vadim/source/repos/labaratornaya/labaratornaya/numbers.txt", "r+");

fscanf\_s(file, "%d", &n);

mas = (unsigned char\*)malloc(n \* sizeof(unsigned int));

for (i = 1; i < n + 1; i++)

{

fscanf\_s(file, "%u\n", &mas[i - 1]);

}

printf("Меню:\n");

printf("\n");

printf("Печать [1]\n");

printf("Сортировка пузырьком [2]\n");

printf("Быстрая сортировка [3]\n");

printf("Сортировка вставкой [4]\n");

printf("Выход [5]\n\n");

printf("Введите номер: ");

scanf\_s("%d", &num);

while (num != 5)

{

switch (num)

{

case 1:

printf("\n");

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

{

printf("%u\n", mas[i]);

}

break;

case 2:

printf("\n");

unsigned int start1 = clock();

bSort(mas, n);

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

{

printf("%u\n", mas[i]);

}

unsigned int end1 = clock();

unsigned int time1 = end1 - start1;

printf("\nвремя работы: %u мс\n", time1);

break;

case 3:

printf("\n");

unsigned int start2 = clock();

qSort(mas, n);

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

{

printf("%u\n", mas[i]);

}

unsigned int end2 = clock();

unsigned int time2 = end2 - start2;

printf("\n\nвремя работы: %u мс\n", time2);

break;

case 4:

printf("\n");

unsigned int start3 = clock();

iSort(mas, n);

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

{

printf("%u\n", mas[i]);

}

unsigned int end3 = clock();

unsigned int time3 = end3 - start3;

printf("\n\nвремя работы: %u мс\n", time3);

break;

}

fclose(file);

file = fopen("C:/Users/Vadim/source/repos/labaratornaya/labaratornaya/numbers.txt", "r+");

fscanf\_s(file, "%d", &n);

for (i = 1; i < n + 1; i++)

{

fscanf\_s(file, "%u\n", &mas[i - 1]);

}

printf("Меню:\n");

printf("\n");

printf("Печать [1]\n");

printf("Сортировка пузырьком [2]\n");

printf("Быстрая сортировка [3]\n");

printf("Сортировка вставкой [4]\n");

printf("Выход [5]\n\n");

printf("Введите номер: ");

scanf\_s("%d", &num);

}

free(mas);

fclose(file);

return 0;

}