**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ВЫСШАЯ ШКОЛА ПЕЧАТИ И МЕДИАИНДУСТРИИ**

***Институт Принтмедиа и информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № 10-11**

**Дисциплина:** Основы алгоритмизации и программирования.

**Тема:** Алгоритм сортировки «Шелла»

**Выполнила:**

**студентка группы 201-723**

Круглова А.М.

20.11.20  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил:** преп. Хуснулина Д.Р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2020**

**Оглавление**

[Цель 3](#_Toc57112146)

[Постановка задачи 3](#_Toc57112147)

[Идея алгоритма 3](#_Toc57112148)

[Словесное представление алгоритма 4](#_Toc57112149)

[Блок-схема с использованием элемента модификации 5](#_Toc57112150)

[Листинг программы с использованием параметрического цикла 6](#_Toc57112151)

[Результат работы программы 7](#_Toc57112152)

[Блок-схема без использования элемента модификации 8](#_Toc57112153)

[Листинг программы с использованием цикла с предусловием 9](#_Toc57112154)

[Результат работы программы 10](#_Toc57112155)

# **Цель**

Получить практические навыки разработке алгоритмов и их программной

реализации.

# **Постановка задачи**

Необходимо выполнить и оформить описание следующих пунктов:

1. Сформулировать идею алгоритма
2. Выполнить словесное представление алгоритма
3. Выполнить полнить представление алгоритма с помощью блок схем с использованием элемента модификации и без него.
4. Выполнить программную реализацию алгоритмов на языке С с использованием параметрического цикла и цикла с предусловием.

# **Идея алгоритма**

Сортировка Шелла (англ. Shell sort) — алгоритм сортировки, являющийся усовершенствованным вариантом сортировки вставками. Идея метода Шелла состоит в

сравнении элементов, стоящих не только рядом, но и на определённом расстоянии друг от

друга. Иными словами — это сортировка вставками с предварительными «грубыми»

проходами. Аналогичный метод усовершенствования пузырьковой сортировки называется сортировка расчёской.

Алгоритм сортирует элементы, отстоящие друг от друга на некотором расстоянии. Затем

сортировка повторяется при меньших значениях шага, и в конце процесс сортировки

Шелла завершается при шаге, равном 1 (а именно обычной сортировкой вставками). Шелл

предложил такую последовательность размера шага: N/2, N/4, N/8 …, где N – количество

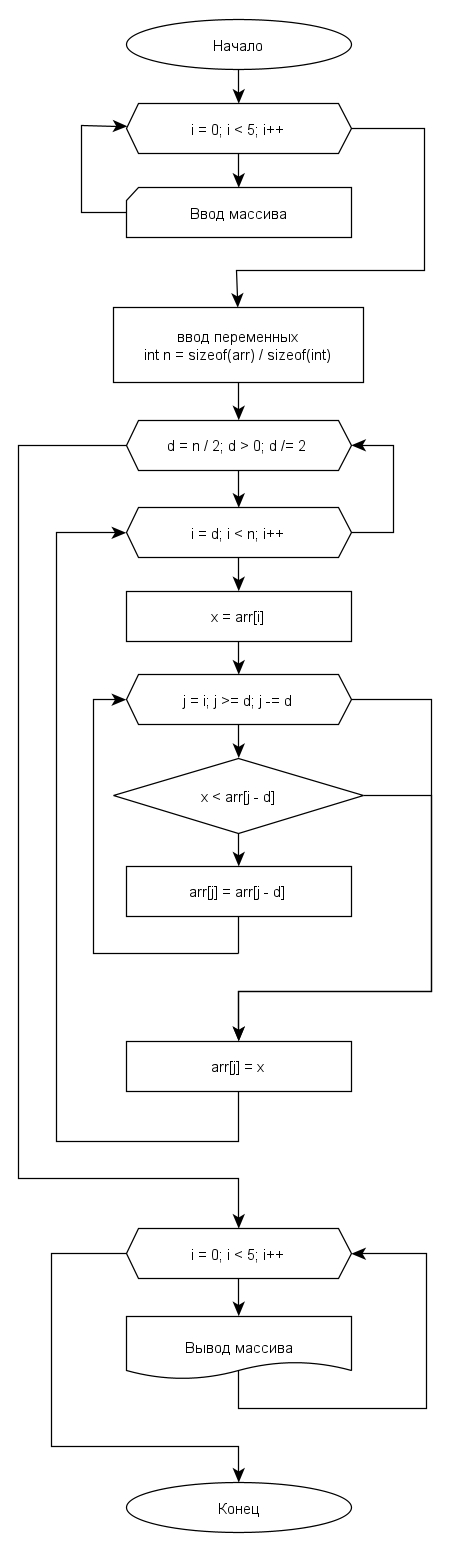
элементов в сортируемом массиве.

Словесное представление алгоритма

n – длина массива, d – шаг

1. Рассчитываем начальное значение шага: d = n / 2
2. Если d > 0, то п.3, иначе п.14
3. Номер анализ. эл-та = d
4. Если номер анализ. элемента < n, то п.5, иначе п.13
5. Запоминаем значение анализ. элемента
6. Номер текущего элемента = номеру анализ. элемента
7. Если номер текущего элемента ≥ d, то п.8, иначе п.11
8. Если значение текущего элемента < значение элемента с номером (текущего элемента – d), то п.9, иначе п.11
9. Значение т.э. = значение эл-та с номером (т.э – d)
10. Номер т.э = номер (т.э – d), п.7
11. Значение т.э = значение анализ. элемента
12. i++, п.4
13. d /=2, п.2
14. Конец алгоритм

Блок-схема с использованием элемента модификации



Да

Нет

Рисунок 1 — Блок-схема 1 программы

Листинг программы с использованием параметрического цикла

Листинг 1 — Исходный код 1 программы

#include <iostream>

#include <stdlib.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "Russian"); //установка русского языка

int i, j, x, d; //ввод переменных

int arr[5]; // Объявляем массив из 5 элементов

printf("Введите 5 цифр в столбик:\n");

for (i = 0; i < 5; i++) // Вводим значения элементов массива через клавиатуру

{

scanf\_s("%d", &arr[i]);

}

int n = sizeof(arr) / sizeof(int); //размер массива

for (d = n / 2; d > 0; d /= 2) //шагу присваивается начальное значение

{

for (i = d; i < n; i++) //начало цикла

{

x = arr[i]; //запоминаем значение анализируемого элемента

for (j = i; j >= d; j -= d) //номеру текущего значения присваивается значение анализируемого элемент

{

if (x < arr[j - d]) //сравнение значений

arr[j] = arr[j - d]; //перестановка

else break;

}

arr[j] = x; //значение текущего элемента становится равно анализир.элементу

}

}

printf("Отсортированный массив:\n");

for (i = 0; i < 5; i++) //вывод массива на экран

printf("%d ", arr[i]);

printf("\n");

return 0;

}

Результат работы программы

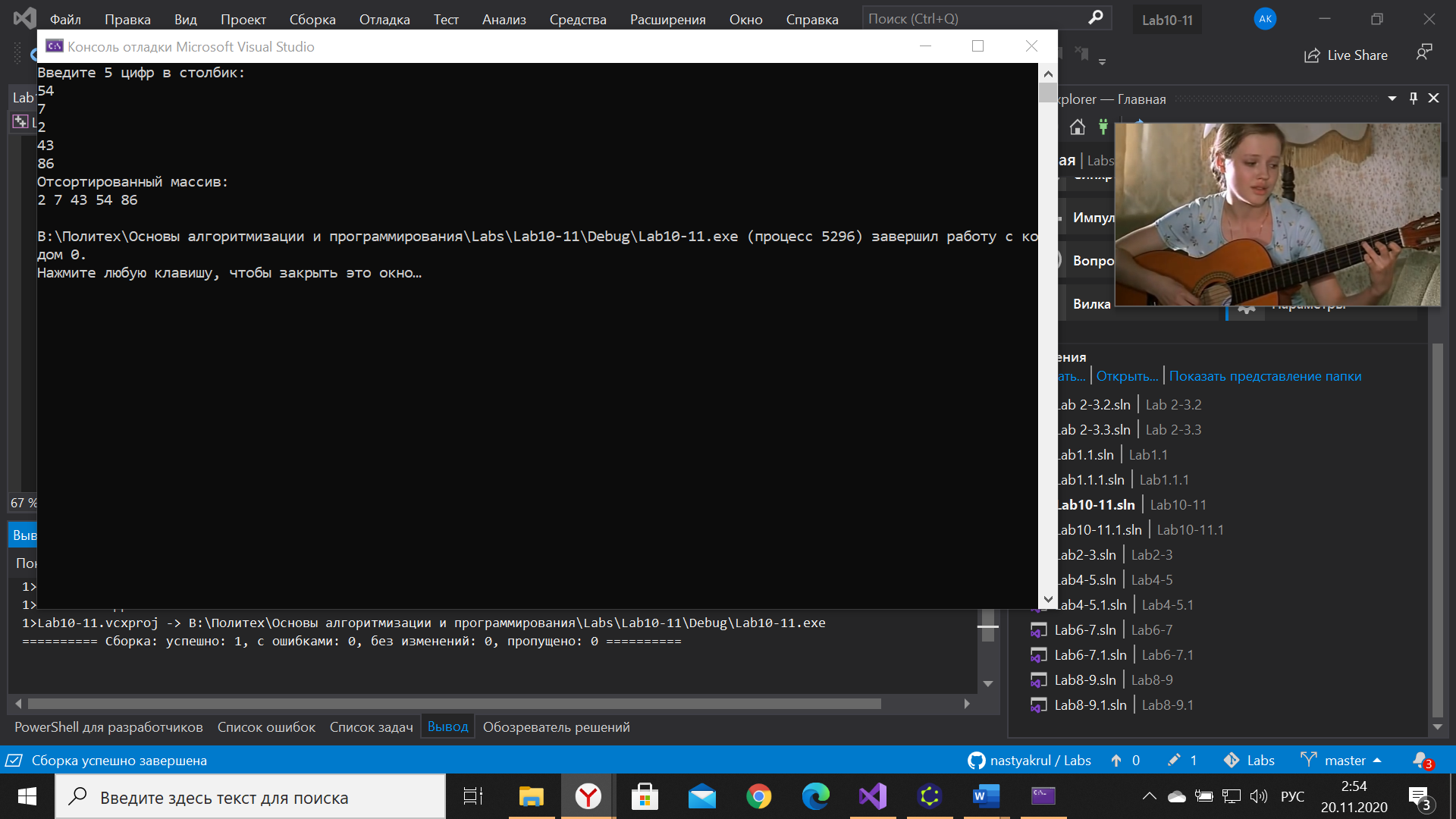
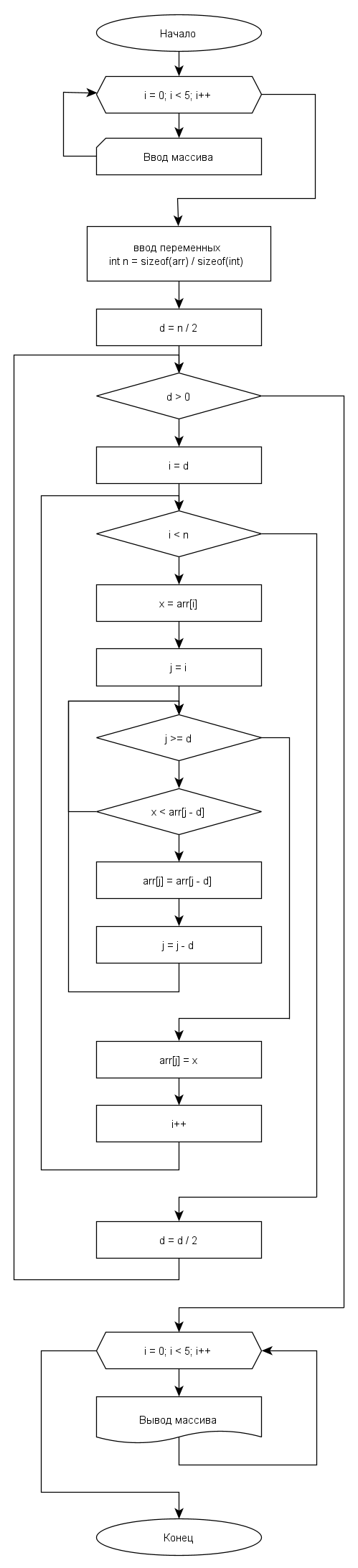


Рисунок 2 — Результат работы 1 программы

Блок-схема без использования элемента модификации



Нет

Да

Да

Нет

Да

Да

Нет

Нет

Рисунок 3 — Блок-схема 2 программы

Листинг программы с использованием цикла с предусловием

Листинг 2 — Исходный код 2 программы

#include <iostream>

#include <stdlib.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "Russian"); //установка русского языка

int i, j, x, d; //ввод переменных

int arr[5]; // Объявляем массив из 5 элементов

printf("Введите 5 цифр в столбик:\n");

for (i = 0; i < 5; i++) // Вводим значения элементов массива через клавиатуру

{

scanf\_s("%d", &arr[i]);

}

int n = sizeof(arr) / sizeof(int); //размер массива

d = n / 2; //начальное значение шагу

while (d > 0)

{

i = d; //номер анализир.элемента приравнивается значению шага

while (i < n)

{

x = arr[i]; //запоминаем значение анализ.элемента

j = i; //ноеру текущего значения присваивается значение анализ. элемента

while (j >= d)

{

if (x < arr[j - d]) //сравнение

{

arr[j] = arr[j - d]; //перестановка

j = j - d; //присваивается новое значение

}

else break;

}

arr[j] = x; //значение текущего элемента становится равным значению анализ.элемента

i++; //увелечение номера анализ.элемента

}

d = d / 2; //новое значение шага

}

printf("Отсортированный массив:\n");

for (i = 0; i < 5; i++) //вывод массива на экран

printf("%d ", arr[i]);

printf("\n");

return 0;

}

Результат работы программы

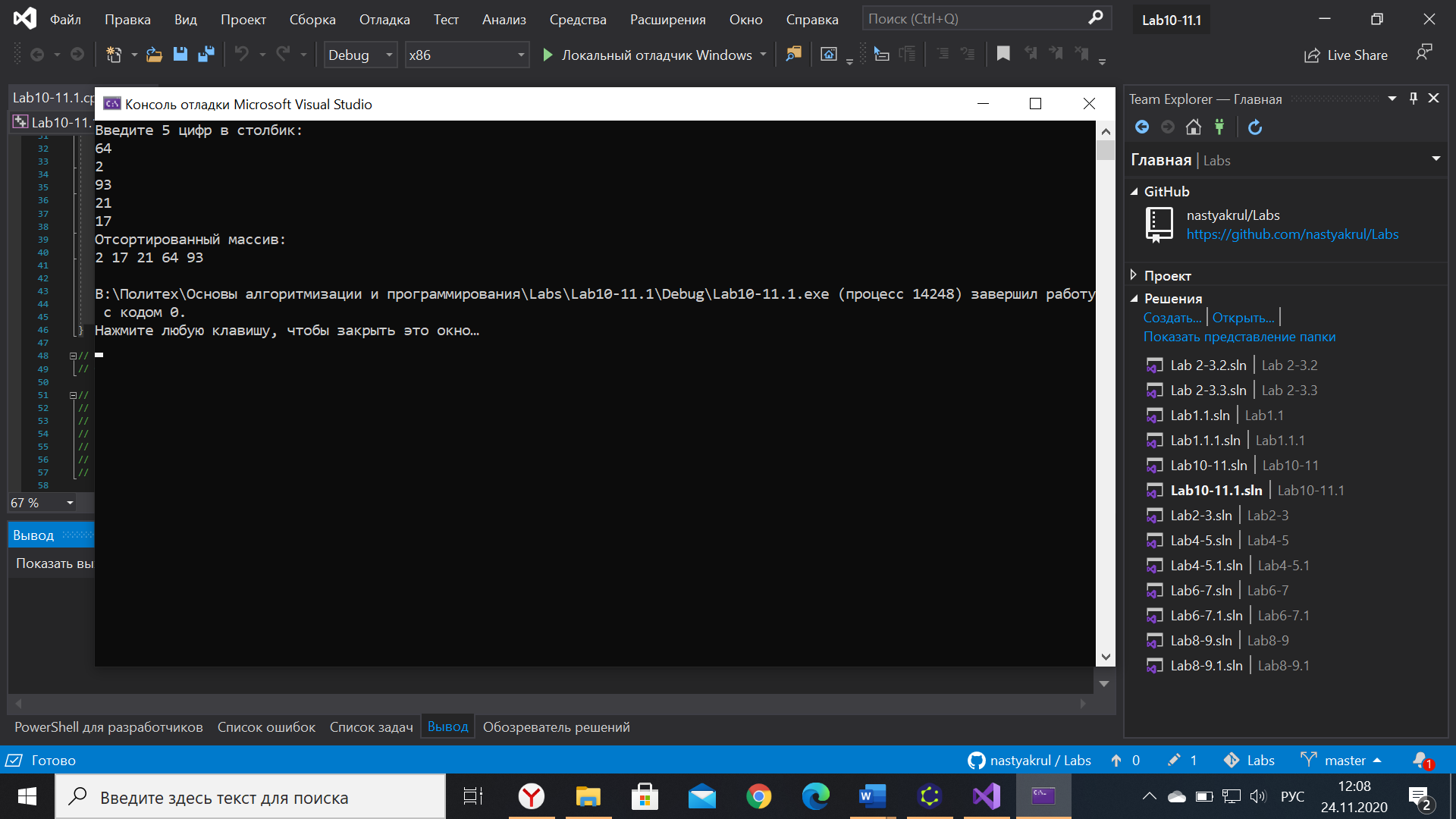


Рисунок 4 — Результат работы 2 программы