

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ(ТУСУР)

Кафедра комплексной информационной безопасности электронно-  
вычислительных систем (КИБЭВС)

## Массивы

Отчет по лабораторной работе №3  
по дисциплине “Основы программирования”

Выполнил

Студент гр. 728-2

\_\_\_\_\_ Геворгян Д. Р.

28 марта 2019

Доцент кафедры БИС

\_\_\_\_\_ Харченко С. С.

“ \_\_ ” \_\_\_\_\_ 2019 г.

## 1. Введение

Цель работы: создание блок-схем алгоритма и программ для обработки одно- и двумерных массивов.

### Вариант 7

Задача 1: Дан массив из  $N$  элементов (вещественные числа). Вычислить: 1) номер максимального элемента массива; 2) произведение элементов массива, расположенных между первым и вторым нулевыми элементами. Преобразовать массив так, чтобы сначала располагались все элементы, стоящие в четных позициях, а потом – элементы, стоящие в нечетных позициях.

Задача 2: Дана матрица  $5 \times 5$ . Вывести ее в верхнем треугольном виде (т.е. напечатать только элементы верхнего треугольника и именно в виде треугольника).

## 2. Теоретическая часть

В программах на C++ можно использовать массивы. Напомним, что массив – это упорядоченный набор величин, обозначаемых одним именем. Данные, являющиеся элементами массива, располагаются в памяти компьютера в определенном порядке, который задается индексами (порядковыми номерами элементов массива). В C++ массив, как и любая переменная, должен быть объявлен. Делается это с помощью служебного слова, указывающего тип, затем указывается имя массива и в квадратных скобках его длина. Заметим, что индексы массива ведут счет с нуля, поэтому запись вида: `double b[14]` означает, что резервируется память для 14 чисел типа `double` с именем `b` и порядковыми номерами от 0 до 13. Отдельный элемент массива записывается с указанием имени и индекса в квадратных скобках.

При работе с массивами необходимо внимательно следить за тем, чтобы не выходить за их объявленные границы. Компилятор C++ (в отличие, например, от Паскаля) не предупреждает об этой ошибке! Попытка ввести больше элементов, чем описано, приведет к неверным результатам, а попытка вывести – выведет случайный результат, находящийся в памяти.

Очевидно, что элементы массива, как правило, располагаются в произвольном порядке. Но во многих случаях может понадобиться расположить их, например, в порядке возрастания. Такая процедура упорядочения называется сортировкой. Рассмотрим простейший способ сортировки – метод пузырька. Суть его в том, что последовательно сравниваются элементы массива и, если очередной элемент меньше предыдущего, то их меняют местами.

Массивы могут быть и многомерными, например, двумерными. В этом случае размерности записываются в двух квадратных скобках: `int d[5][4]` – описан целочисленный массив из 5 строк и 4 столбцов. Напомним, что в математике двумерный массив принято называть матрицей, при этом матрица, у которой число строк равно числу столбцов называется квадратной. Если у квадратной матрицы элементы симметрично относительно главной диагонали равны ( $a_{21} = a_{12}$  и т.д.), то ее называют симметричной. Напомним, что главная диагональ матрицы содержит элементы с одинаковыми индексами:  $a_{11}$ ,  $a_{22}$  и т.д. Побочная диагональ – вторая диагональ матрицы; верхний треугольник – элементы над главной диагональю (включая и элементы на диагонали); нижний треугольник – элементы под главной диагональю (включая и элементы на диагонали). В случае, когда элементы верхнего и нижнего треугольника совпадают, то говорят, что матрица симметрична.

Также можно использовать датчик случайных чисел. На C++ он реализуется функцией `rand()%RAND_MAX` – дает случайное целое число из интервала от 0 до положительного целого `RAND_MAX-1` (в качестве `RAND_MAX` можно указать сразу конкретное число). Кроме того, для запуска этого датчика случайных чисел (чтобы получать новый набор случайных чисел при каждом новом запуске программы), используют функцию `srand(time(0))`, т.е. для старта генератора случайных чисел будет использоваться системное время компьютера. Причем это время следует преобразовать в беззнаковое целое (`unsigned int`). Кроме того, требуется подключить заголовочный файл, содержащий эти функции. Для получения системного времени нужно подключить заголовочный файл.

### 3. Ход работы

#### 3.1. Решение задачи 1

Для начала составим алгоритм:

- A1.  $n \leftarrow 7, q \leftarrow 1, i \leftarrow 0$ ;
- A2. Если  $i < n$  выполняем A3, иначе A5;
- A3. Ввод  $a[i]$ ;
- A4.  $i \leftarrow i + 1$ , переход на A2;
- A5.  $\max \leftarrow a[0], i \leftarrow 0$ ;
- A6. Если  $i < n$  выполняем A7, иначе A10;
- A7. Если  $a[i] > \max$  выполняем A8, иначе A9;
- A8.  $\max \leftarrow a[i]$ ;
- A9.  $i \leftarrow i + 1$ , переход на A6;
- A10. Вывод “Максимальный элемент массива = ”,  $n$ ;
- A11.  $i \leftarrow 0$ ;
- A12. Если  $i < n$  выполняем A13, иначе A17
- A13. Если  $a[i] = 0$  выполняем A14, иначе A16
- A14.  $i1 \leftarrow i$ ;
- A15.  $i \leftarrow n$ , переход на A12;
- A16.  $i \leftarrow i + 1$ , переход на A12;
- A17. Если  $i1 = -1$  выполняем A18, иначе A20;
- A18. Вывод “В массиве отсутствуют нулевые элементы,  
перезапустите программу и введите другие данные”;
- A19. Конец программы;
- A20.  $i \leftarrow i1$ ;
- A21. Если  $i < n$  выполняем A22, иначе A26;
- A22. Если  $a[i] = 0$  выполняем A23, иначе A25;
- A23.  $i2 \leftarrow i$ ;

A24.  $i \leftarrow n$ , переход на A21;  
 A25.  $i \leftarrow i + 1$ , переход на A21;  
 A26. Если  $i_2 = -1$  выполняем A27, иначе A29;  
 A27. Вывод “В массиве присутствует всего один нулевой элемент, перезапустите программу и введите другие данные”;  
 A28. Конец программы;  
 A29.  $i \leftarrow i_1 + 1$ ;  
 A30. Если  $i < i_2$  выполняем A31, иначе A33;  
 A31.  $q \leftarrow q * a[i]$ ;  
 A32.  $i \leftarrow i + 1$ , переход на A31;  
 A33. Если  $i_2 - i_1 = 0$  выполняем A34, иначе A35;  
 A34.  $q = 0$ ;  
 A35. Вывод “Произведение элементов массива, расположенных между первым и вторым ненулевым элементами, равно ”  $q$ ;  
 A36.  $j \leftarrow \frac{n}{2}$ ,  $k \leftarrow 0$ ,  $i \leftarrow 0$ ;  
 A37. Если  $i < n$  выполняем A38, иначе A40;  
 A38.  $b[i] \leftarrow 0$ ;  
 A39.  $i \leftarrow i + 1$ , переход на A37;  
 A40.  $i \leftarrow 0$ ;  
 A41. Если  $i < n$  выполняем A42, иначе A47;  
 A42. Если остаток от деления  $\frac{i}{2} = 0$  выполняем A43, иначе A45;  
 A43.  $b[j] = a[i]$ ;  
 A44.  $j \leftarrow j + 1$ ,  $i \leftarrow i + 1$ , переход на A41;  
 A45.  $b[k] \leftarrow a[i]$ ;  
 A46.  $k \leftarrow k+1$ ,  $i \leftarrow i + 1$ , переход на A41;  
 A47.  $i \leftarrow 0$ ;  
 A48. Если  $i < n$  выполняем A49, иначе A51;  
 A49. Вывод  $b[i]$ , “ ”;

A50.  $i \leftarrow i + 1$ , переход на A48;

A51. Конец программы.

Теперь представим решение задачи в виде блок-схемы:

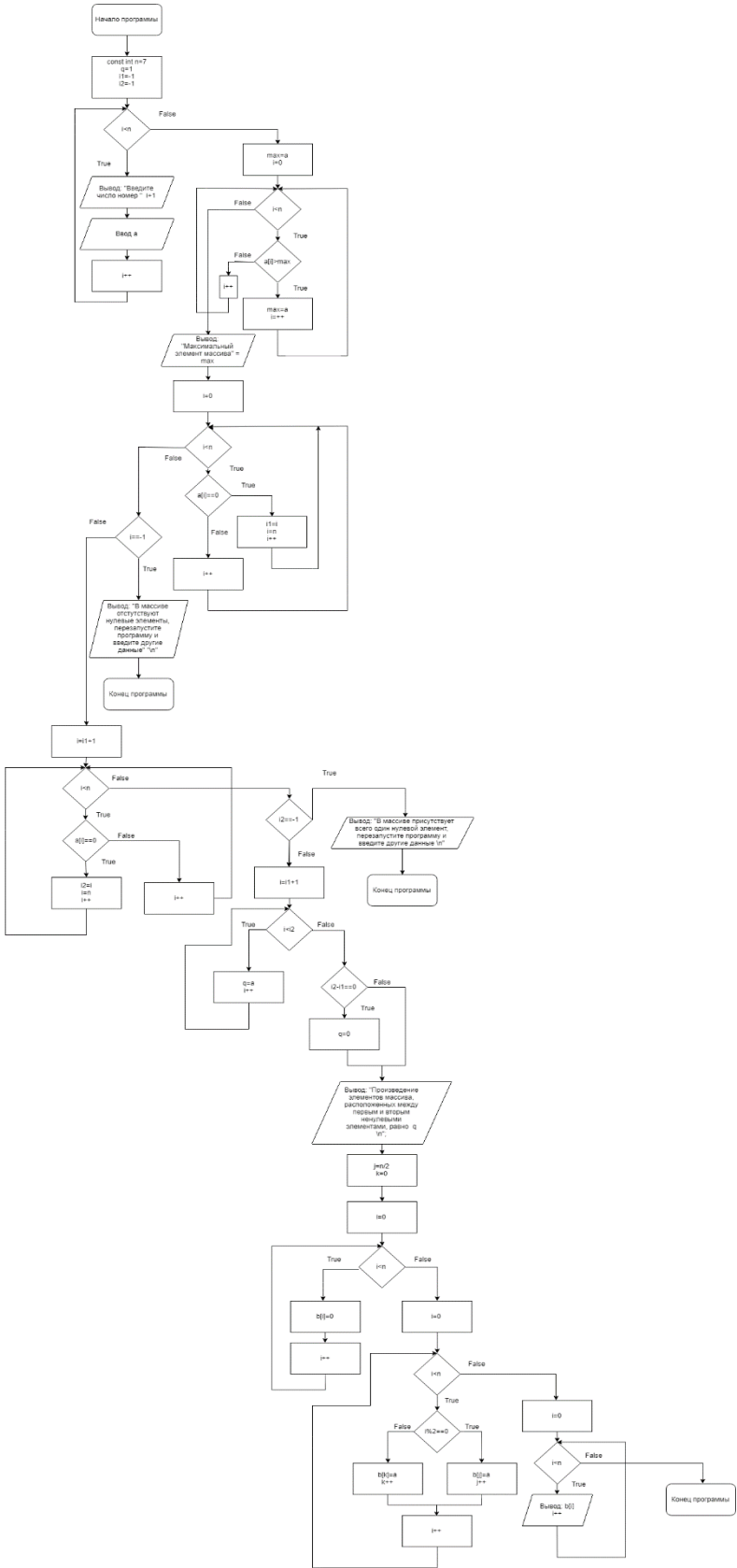


Рисунок 1 – Блок-схема задачи 1



Далее напишем код программы на языке C++:

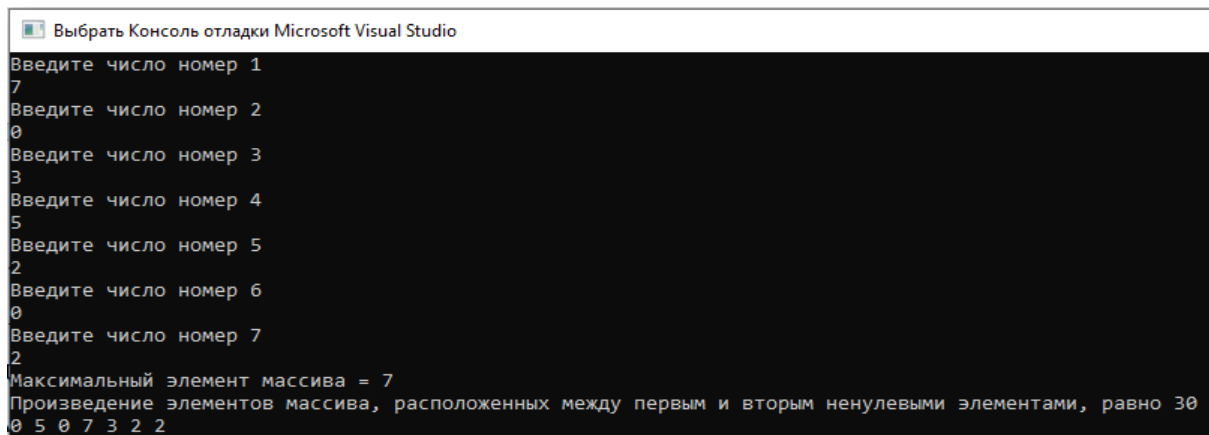
```
#include "pch.h"
#include <iostream>
#include <ctime>

using namespace std;
int main()
{
    setlocale(LC_ALL, "Russian");
    const int n = 7;
    int q = 1, max, i, i1 = -1, i2 = -1, a[n];
    for (i = 0; i < n; i++)
    {
        cout << "Введите число номер " << i + 1 << "\n";
        cin >> a[i];
    }
    max = a[0];
    for (i = 0; i < n; i++)
    {
        if (a[i] > max)
        {
            max = a[i];
        }
    }
    cout << "Максимальный элемент массива = " << max << "\n";
    for (i = 0; i < n; i++)
    {
        if (a[i] == 0)
        {
            i1 = i;
            i = n;
        }
    }
    if (i1 == -1)
    {
        cout << "В массиве отсутствуют нулевые элементы,
перезапустите программу и введите другие данные" << "\n";
        return 0;
    }
    for (i = i1 + 1; i < n; i++)
    {
        if (a[i] == 0)
```

```

        {
            i2 = i;
            i = n;
        }
    }
    if (i2 == -1)
    {
        cout << "В массиве присутствует всего один нулевой элемент,
перезапустите программу и введите другие данные" << "\n";
        return 0;
    }
    for (i = i1 + 1; i < i2; i++)
    {
        q *= a[i];
    }
    if (i2 - i1 == 0)
        q = 0;
    cout << "Произведение элементов массива, расположенных между
первым и вторым ненулевыми элементами, равно " << q << "\n";
    int b[n], j = n / 2, k = 0;
    for (i = 0; i < n; i++)
    {
        b[i] = 0;
    }
    for (i = 0; i < n; i++)
    {
        if (i % 2 == 0)
        {
            b[j] = a[i];
            j++;
        }
        else
        {
            b[k] = a[i];
            k++;
        }
    }
    for (i = 0; i < n; i++)
    {
        cout << b[i] << " ";
    }
    return 0;
}

```

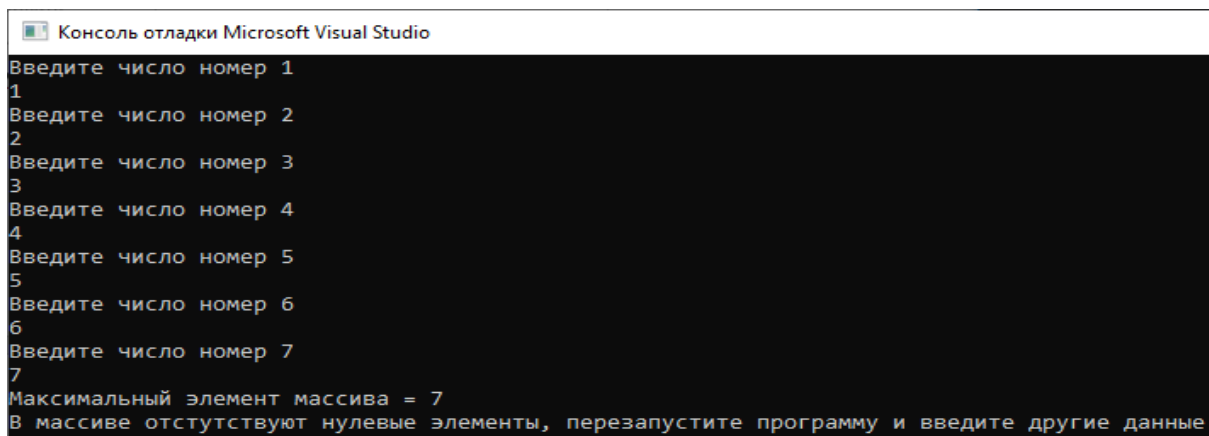


```

Выбрать Консоль отладки Microsoft Visual Studio
Введите число номер 1
7
Введите число номер 2
0
Введите число номер 3
3
Введите число номер 4
5
Введите число номер 5
2
Введите число номер 6
0
Введите число номер 7
2
Максимальный элемент массива = 7
Произведение элементов массива, расположенных между первым и вторым ненулевыми элементами, равно 30
0 5 0 7 3 2 2

```

Рисунок 2 – Результат работы программы 1 при корректных значениях

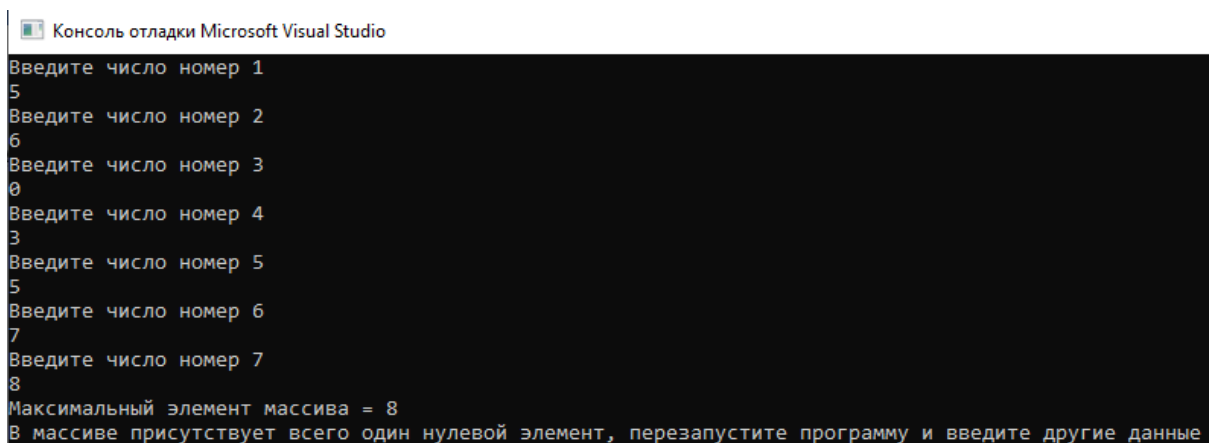


```

Консоль отладки Microsoft Visual Studio
Введите число номер 1
1
Введите число номер 2
2
Введите число номер 3
3
Введите число номер 4
4
Введите число номер 5
5
Введите число номер 6
6
Введите число номер 7
7
Максимальный элемент массива = 7
В массиве отсутствуют нулевые элементы, перезапустите программу и введите другие данные

```

Рисунок 3 – Результат работы программы 1 при отсутствии нулевых элементов



```

Консоль отладки Microsoft Visual Studio
Введите число номер 1
5
Введите число номер 2
6
Введите число номер 3
0
Введите число номер 4
3
Введите число номер 5
5
Введите число номер 6
7
Введите число номер 7
8
Максимальный элемент массива = 8
В массиве присутствует всего один нулевой элемент, перезапустите программу и введите другие данные

```

Рисунок 4 – Результат работы программы 1 при наличии одного нулевого элемента

### 3.1. Решение задачи 2

Для начала составим алгоритм:

- A1.  $n \leftarrow 5$ ;
- A2. Инициализация генератора псевдослучайных чисел;
- A3.  $i \leftarrow 0$ ;  $j \leftarrow 0$ ;
- A4. Если  $i < n$  выполняем A5, иначе A9;
- A5. Если  $j < n$  выполняем A6, иначе A8;
- A6.  $a[i][j] \leftarrow$  случайное число  $[1..10]$ ;
- A7.  $j \leftarrow j$ , переход на A5;
- A8.  $i \leftarrow i$ , переход на A4;
- A9.  $i \leftarrow 0$ ,  $j \leftarrow 0$ ;
- A10. Если  $i < n$  выполняем A11, иначе A19;
- A11. Если  $j < n$  выполняем A12, иначе A17;
- A12. Если  $i > j$  выполняем A13, иначе A14;
- A13. Вывод “ ”, переход на A16;
- A14. Вывод  $a[i][j]$ ;
- A15. Вывод “ ”;
- A16.  $j \leftarrow j + 1$ , переход на A11;
- A17. Вывод символа новой строки;
- A18.  $i \leftarrow i + 1$ , переход на A10;
- A19. Конец программы.

Теперь представим блок-схему к данному алгоритму:

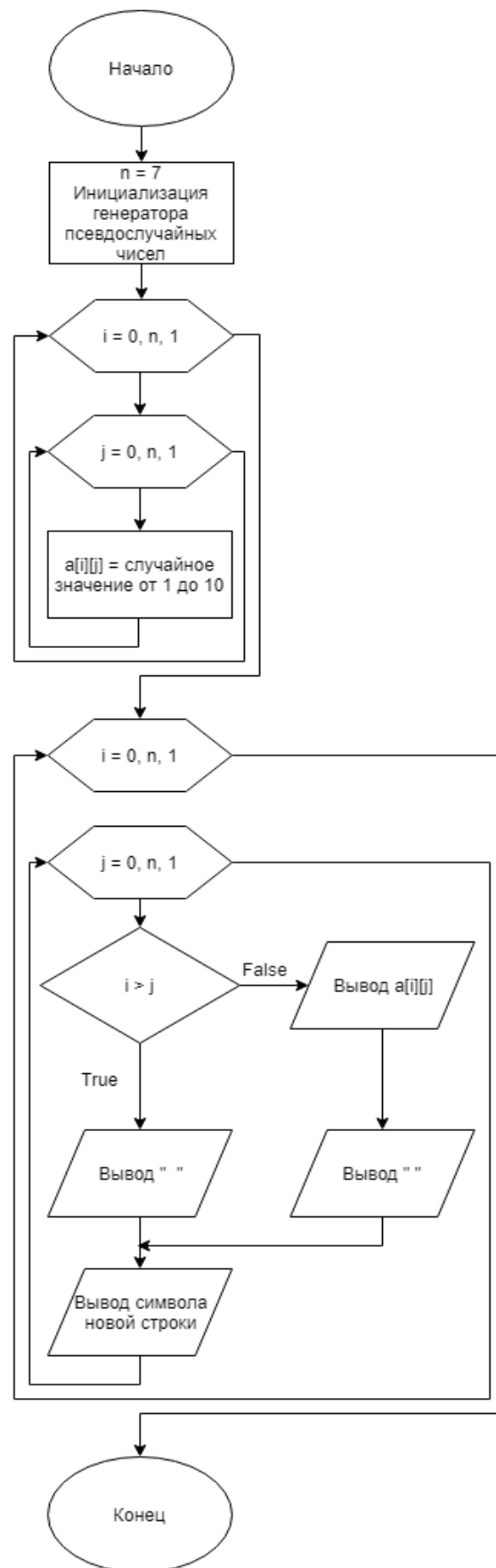
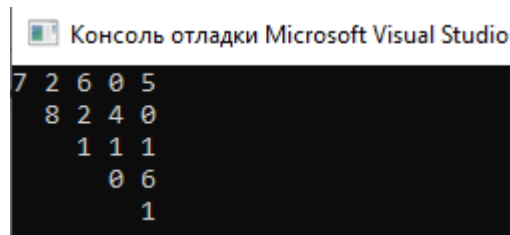


Рисунок 5 – Блок-схема задачи 2

Код программы на языке C++:

```
#include "pch.h"
#include <iostream>
#include <stdlib.h>
#include <iomanip>
#include <time.h>

using namespace std;
int main()
{
    setlocale(LC_ALL, "Russian");
    const int n = 5;
    int i, j, a[n][n];
    srand((unsigned int)time(0));
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
        {
            a[i][j] = rand() % 10;
        }
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < n; j++)
        {
            if (i > j)
                cout << " ";
            else
            {
                cout << a[i][j];
                cout << " ";
            }
        }
        cout << "\n";
    }
    return 0;
}
```



```
Консоль отладки Microsoft Visual Studio
7 2 6 0 5
8 2 4 0
1 1 1
0 6
1
```

Рисунок 6 – Результат работы программы 2

#### 4. Заключение

В процессе выполнения лабораторной работы были получены навыки создания блок-схем алгоритма и программ для обработки одно- и двумерных массивов.

Отчёт был составлен согласно ОС ТУСУР 2013.