Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ПНИПУ)

ОТЧЁТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №5

по теме:

ФУНКЦИИ И МАССИВЫ

Выполнила: студентка группы РИС-22-1б

Черкасова А.А\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил: доцент кафедры ИТАС

Полякова О.А\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_/\_\_\_\_\_\_\_/\_\_\_\_\_г.

Пермь 2023

**Содержание**

[**Введение** 3](#_Toc127379672)

[**Анализ задачи** 3](#_Toc127379673)

[**Блок – схема** 5](#_Toc127379674)

[**Приложение А** 11](#_Toc127379675)

[**Приложение Б** 14](#_Toc127379676)

# **Введение**

**Постановка задачи:** Задан двумерный массив N x N.

1. Последовательно рассматриваются квадратные подмассивы, правый верхний элемент которых лежит на побочной диагонали.
2. В каждом таком подмассиве находится максимальный элемент.
3. Путем перестановок строк и столбцов (целиком) элемент надо переместить в правый верхний угол подмассива.
4. Проверить получилась ли на побочной диагонали убывающая последовательность элементов.

**Цель:** Организовать обработку массивов с использованием функций, научиться передавать массивы как параметры функций.

Для достижения поставленной цели, необходимо решить следующие **задачи**:

* Провести анализ задачи
* Реализовать задачу на языке С++
* Составить блок-схему

# **Анализ задачи**

1. Перебором двумерного массива находится максимальный элемент в подматрице с помощью функции *MaxElem*, а также его индексы *i* и *j*. Стоит учитывать, что сама матрица также является подматрицей.

2. После того, как нашли максимальный элемент, проверяем, в какой строке он стоит, если он стоит не в первой строке *(i != 0),* то меняем местами текущую строку с первой.

3. Далее, когда строка, в которой стоит максимальный элемент, стала первой, необходимо поменять столбец, в котором стоит максимальный элемент, со столбцом с индексом *j = N – 1 (N – номер).* Пункты 2 – 3 реализовываются в функции *Swap.*

4. Пункты 1-3 выполняются рекурсивно, каждый раз рассматривая подмассивы меньшей на единицу размерности. Рекурсия происходит в функции *Podmas.* Здесь выделяется подмассив, по нему создаётся новый двумерный массив, в нём ищется максимальный элемент и встаёт в правый верхний угол. Далее этот «отсортированный» подмассив возвращается в матрицу.

5. В функции *main* создаётся матрица, заполненная случайными числами и вызывается функция *Podmas.*

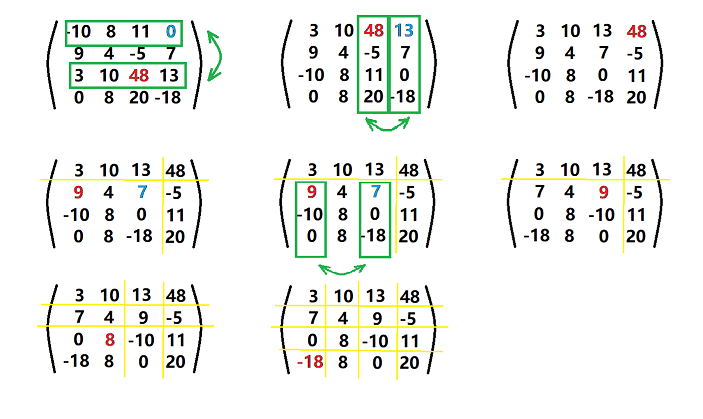
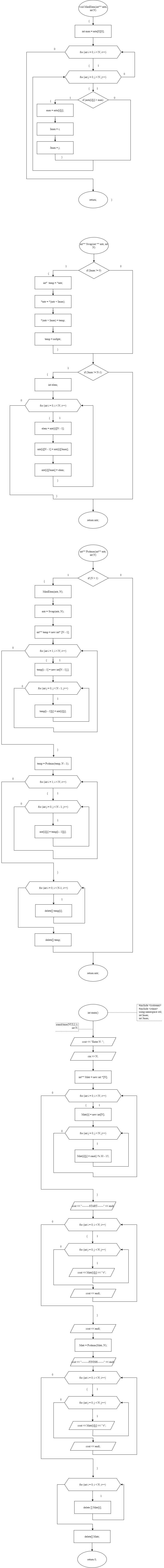


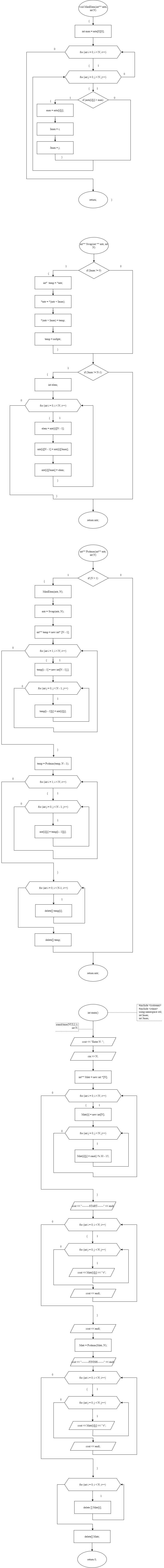
Рисунок 1 - Пример алгоритма

# **Блок – схема**

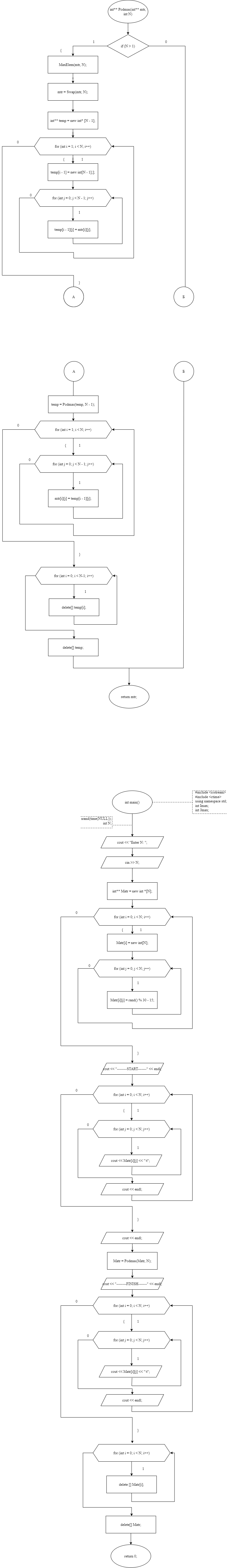
A) Функция *MaxElem:*

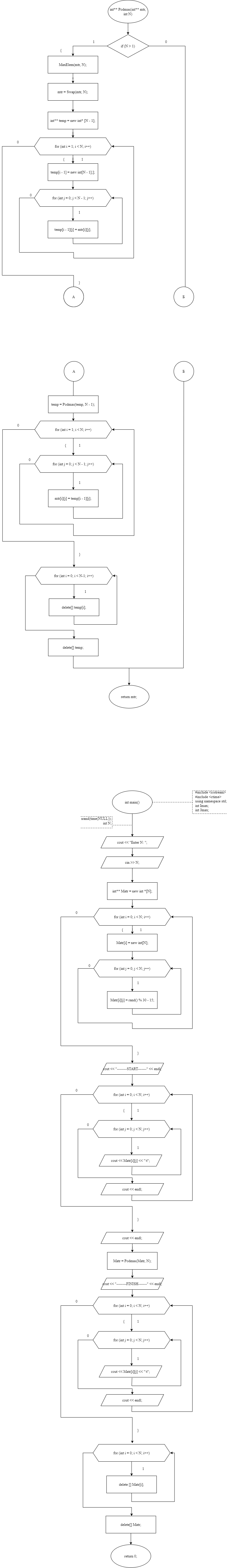
**

B) Функция *Swap:*

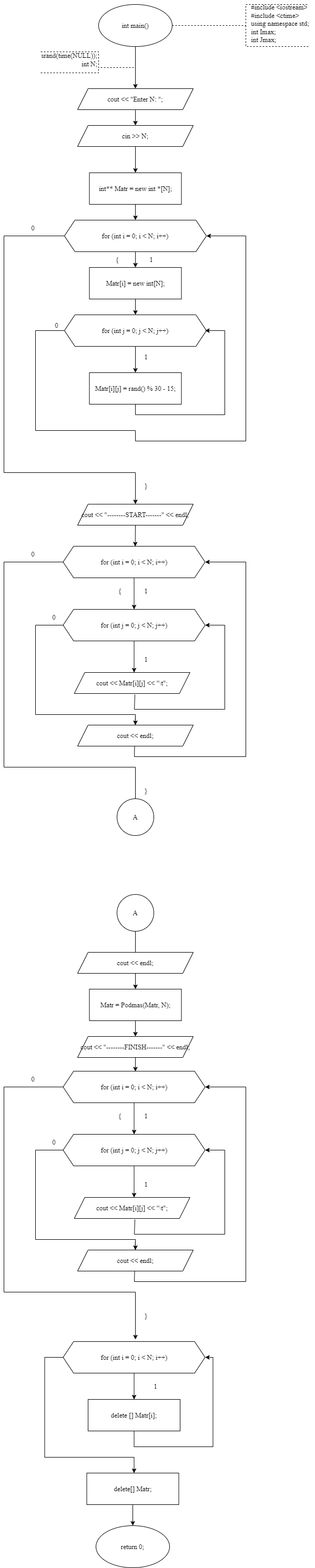


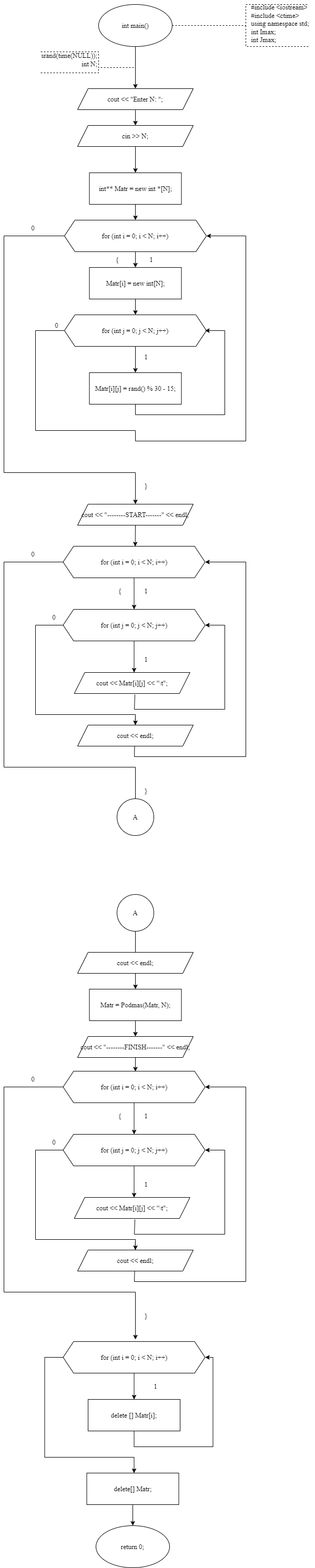
C) Функция *Podmas:*

**



D) Функция *main:*

**

****

# 

# **Приложение А**

**Листинг программы**

#include <iostream>

#include <ctime>

using namespace std;

int Imax;

int Jmax;

void MaxElem(int\*\* mtx, int N) // нахождение макимального элемента

{

int max = mtx[0][0];

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

{

for (int j = 0; j < N; j++)

{

if (mtx[i][j] > max)

{

max = mtx[i][j];

Imax = i;

Jmax = j;

}

}

}

}

int\*\* Swap(int \*\* mtr, int N)

{

#pragma region Row

if (Imax != 0) // поменять i-ю строку c первой

{

int\* temp = \*mtr; // запоминаем 0ю строку

\*mtr = \*(mtr + Imax); // 0ю строку заполняем iой

\*(mtr + Imax) = temp; //i-ю строку заполняем 0ой

temp = nullptr;

}

#pragma endregion

#pragma region Column

if (Jmax != N-1)

{

int elem;

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

{

elem = mtr[i][N - 1]; // элемент запоминаем последний столбец

mtr[i][N - 1] = mtr[i][Jmax];

mtr[i][Jmax] = elem;

}

}

#pragma endregion

return mtr;

}

int\*\* Podmas(int\*\* mtr, int N)

{

if (N > 1)

{

MaxElem(mtr, N);

mtr = Swap(mtr, N);

int\*\* temp = new int\* [N - 1]; // выделяется подмассив

for (int i = 1; i < N; i++) // первая строка не рассматривается

{

temp[i - 1] = new int[N - 1];

for (int j = 0; j < N - 1; j++) // не рассматривается последний столбец

{

temp[i - 1][j] = mtr[i][j];

}

}

temp = Podmas(temp, N - 1);

for (int i = 1; i < N; i++) // первая строка не рассматривается

{

for (int j = 0; j < N - 1; j++) // не рассматривается последний столбец

{

mtr[i][j] = temp[i - 1][j]; // возвращение отсортированного подмассива в массив

}

}

for (int i = 0; i < N-1; i++) delete[] temp[i]; // сначала удаляются вложенные массивы, затем массив, содержащий их

delete[] temp;

}

return mtr;

}

int main()

{

#pragma region Matrix creation

srand(time(NULL));

int N;

cout << "Enter N: ";

cin >> N;

int\*\* Matr = new int \*[N]; // массив, содержащий N указателей

for (int i = 0; i < N; i++) // цикл отвечает за количество строк в матрице

{

Matr[i] = new int[N]; // создается массив, который будет содержать в себе элементы

for (int j = 0; j < N; j++)

{

Matr[i][j] = rand() % 30 - 15;

}

}

#pragma endregion

#pragma region Start Matrix output

cout << "-------------------------------START-----------------------------------" << endl;

for (int i = 0; i < N; i++) // вывод массива

{

for (int j = 0; j < N; j++)

{

cout << Matr[i][j] << "\t";

}

cout << endl;

}

cout << endl;

#pragma endregion

Matr = Podmas(Matr, N);

#pragma region End Matrix output

cout << "-------------------------------FINISH-----------------------------------" << endl;

for (int i = 0; i < N; i++) // вывод массива

{

for (int j = 0; j < N; j++)

{

cout << Matr[i][j] << "\t";

}

cout << endl;

}

#pragma endregion

for (int i = 0; i < N; i++) delete [] Matr[i]; // сначала удаляются вложенные массивы, затем массив, содержащий их

delete[] Matr;

}

# 

# **Приложение Б**

**Результаты выполнения программы**

