ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА» (РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ О ПРАКТИЧЕСКОЙ РАБОТЕ № 1

По дисциплине «Языки программирования»

Выполнил: ст. гр. ТКИ – 111

Белавинская М.А.

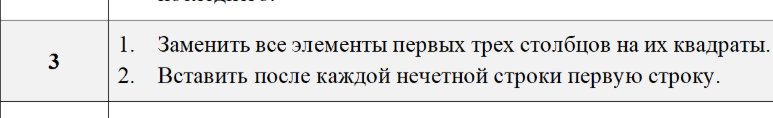
Проверил: к.т.н., доц.

Васильева М. А.

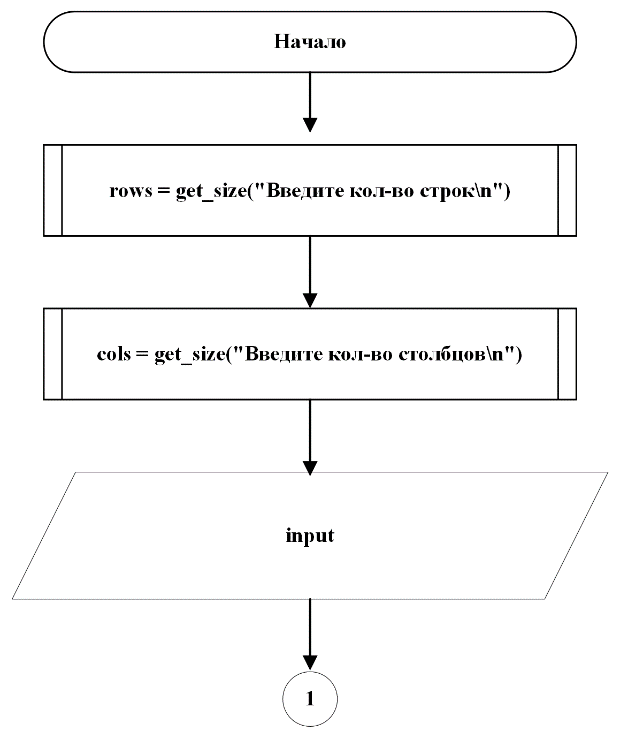
Москва 2022

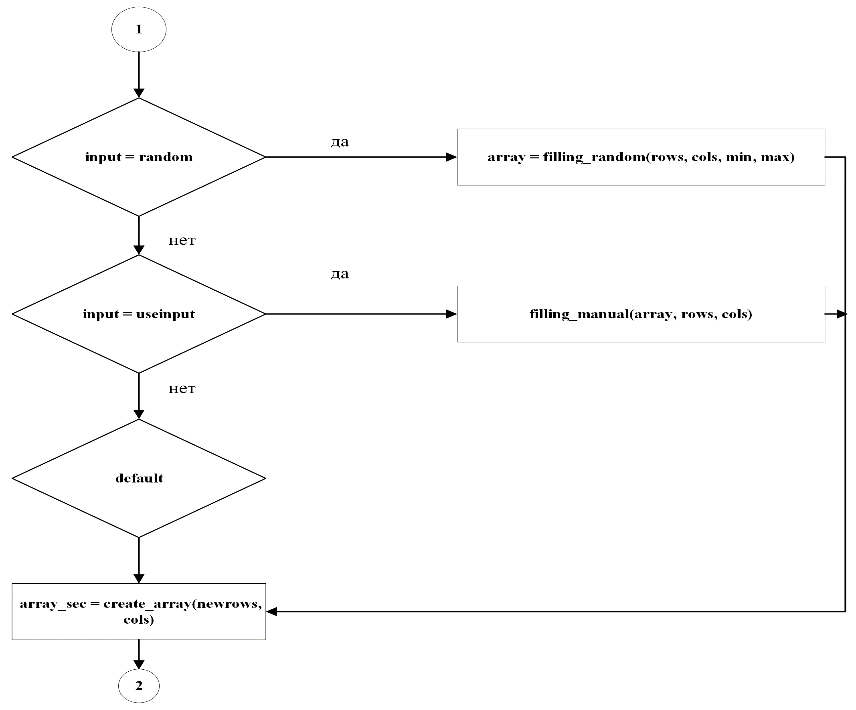
**Задание 4-3**

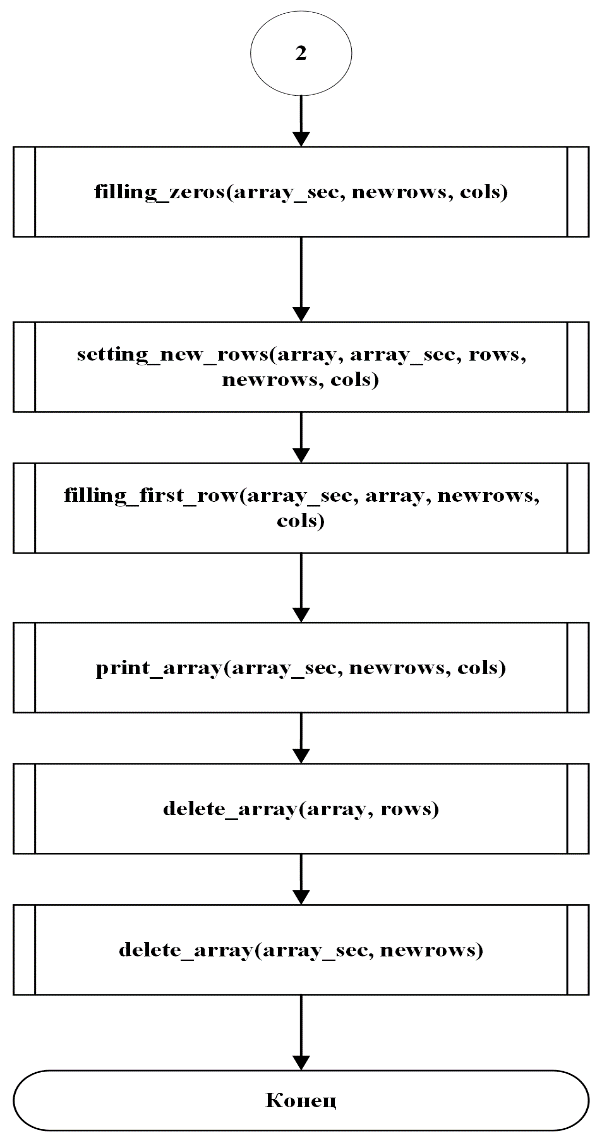
1. **Формулировка задачи**

Создать многомерный массив nˣm из n целых чисел. Реализовать возможность заполнения массива, как случайными числами, так и с помощью клавиатуры по желанию пользователя. Пункты задания организовать в виде функций (методов). Вывести массив на экран. ****

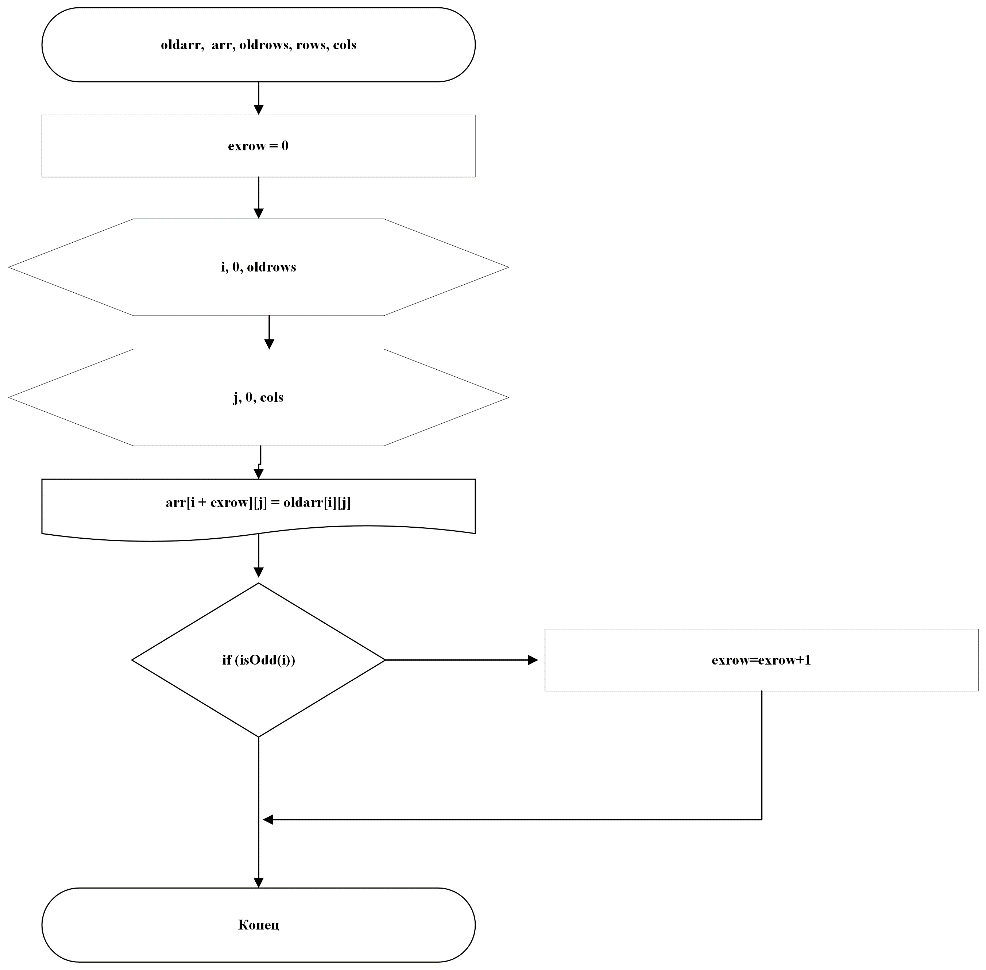
1. **Блок-схемы алгоритмов функции представлены на рисунках (Рисунок 1-11)**

****

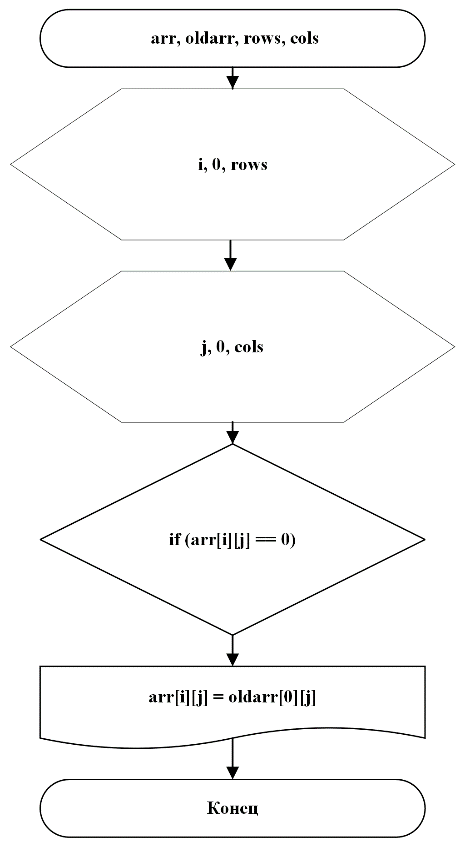
****

****

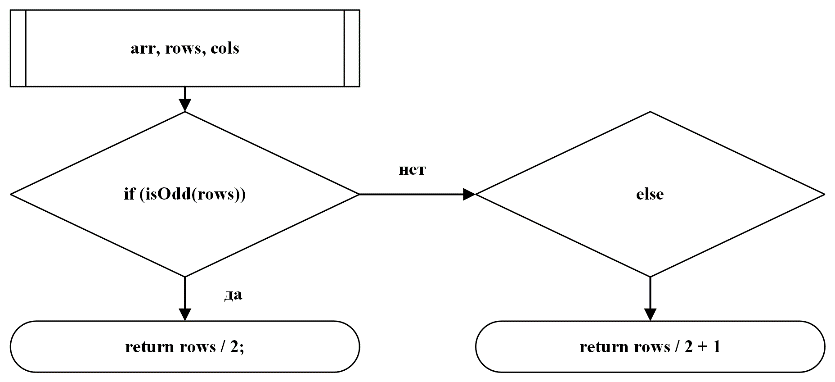
**Рисунок 1 – Блок-схема функции setting\_new\_rows**

****

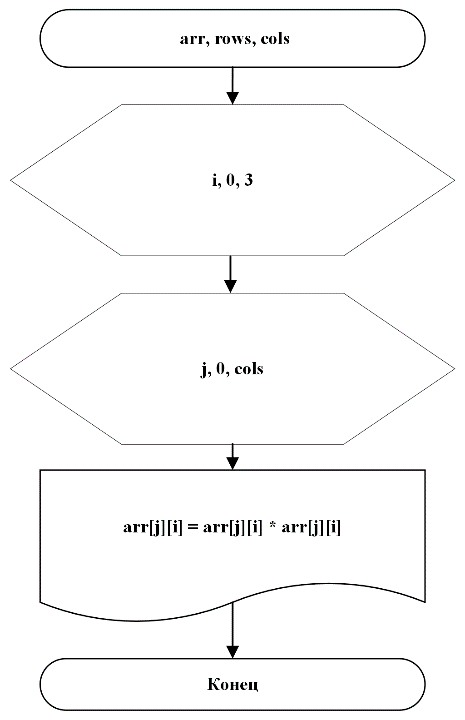
**Рисунок 2 – Блок-схема функции filling\_first\_row**



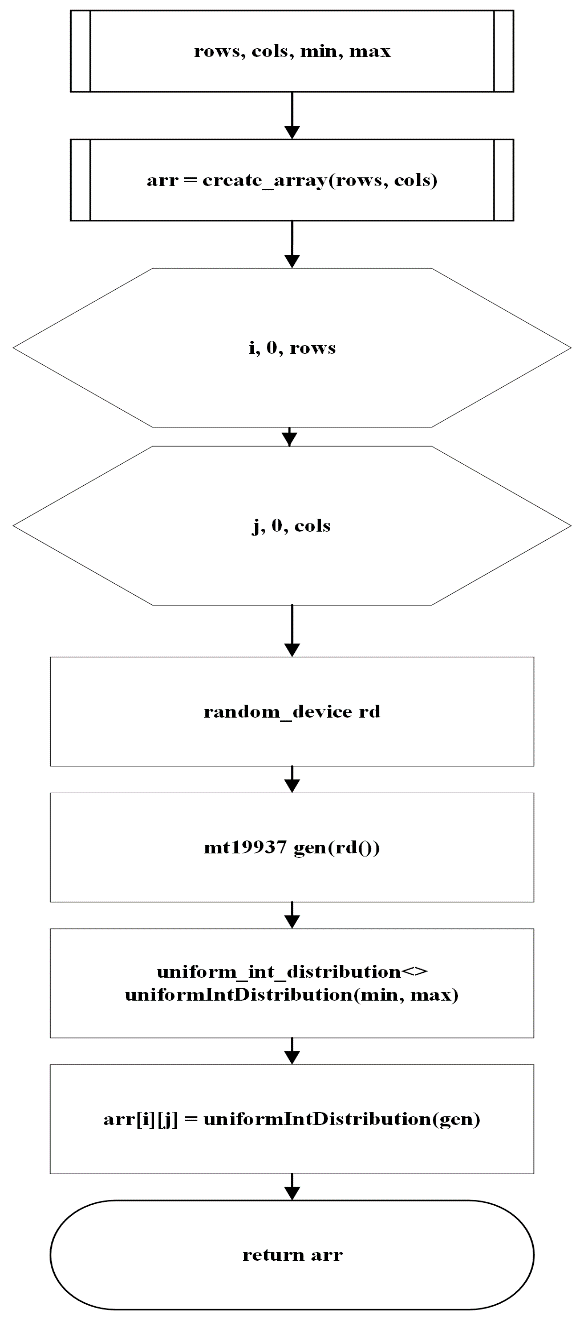
**Рисунок 3 – Блок-схема функции counting\_extra\_rows**

****

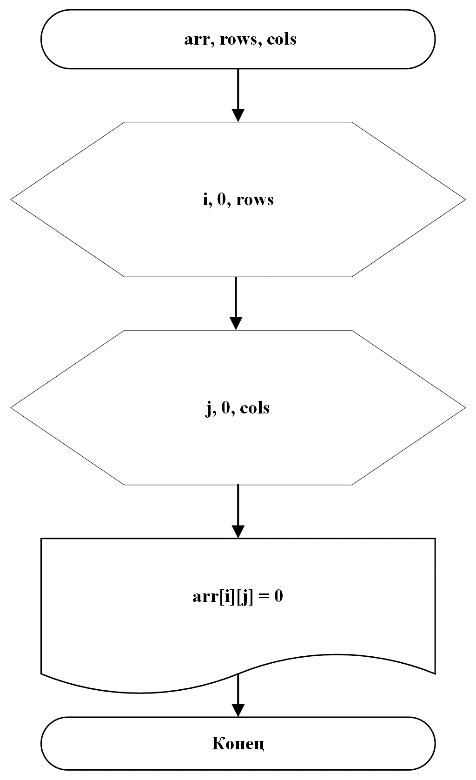
**Рисунок 4 – Блок-схема функции square\_three\_cols**

****

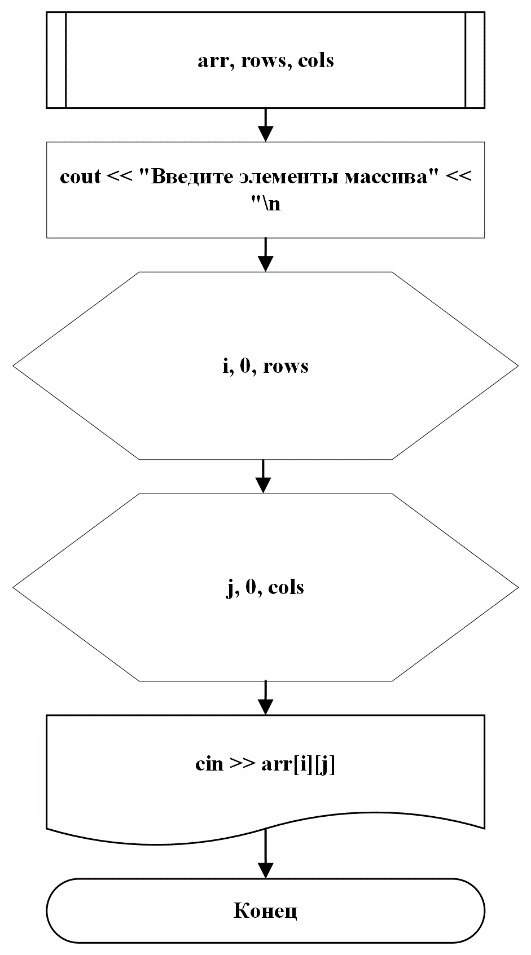
**Рисунок 5 – Блок-схема функции filling\_random**

****

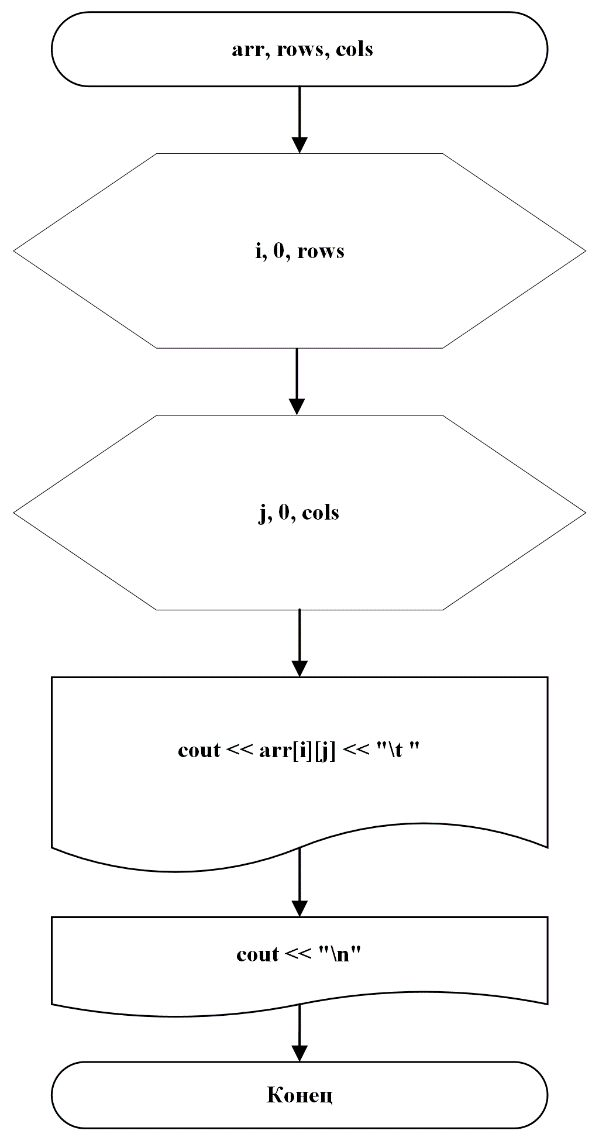
**Рисунок 6 – Блок-схема функции filling\_zeros**

****

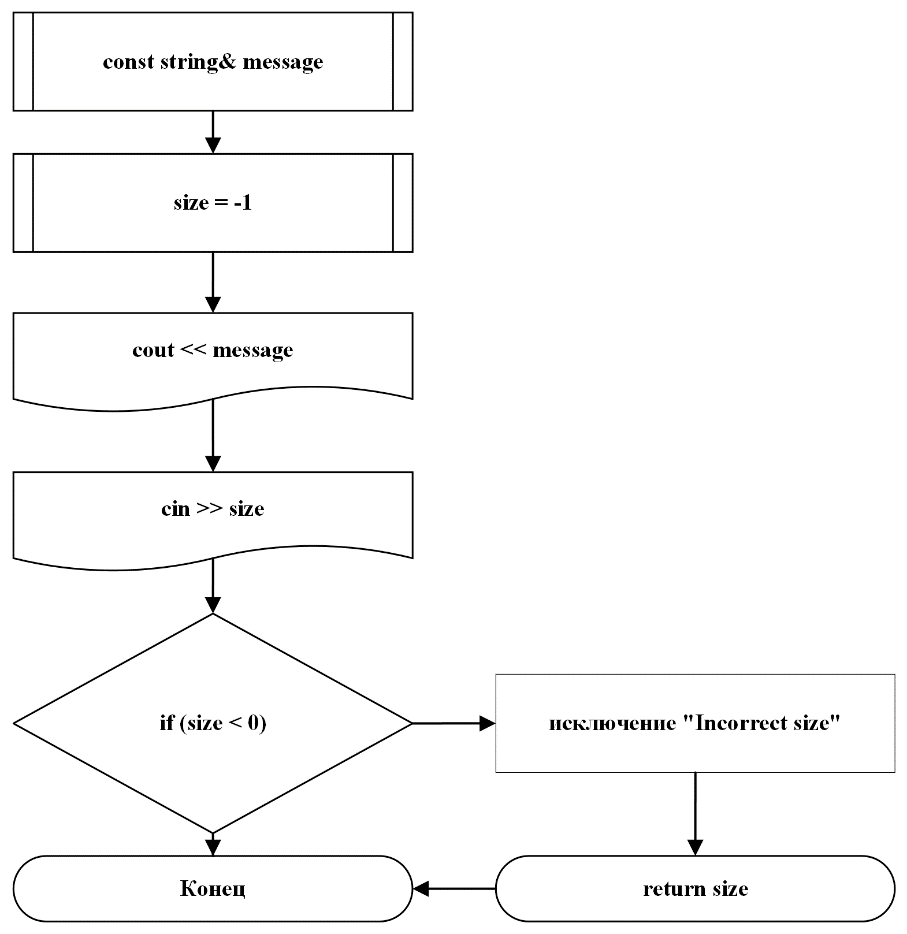
**Рисунок 7 – Блок-схема функции filling\_manual**

****

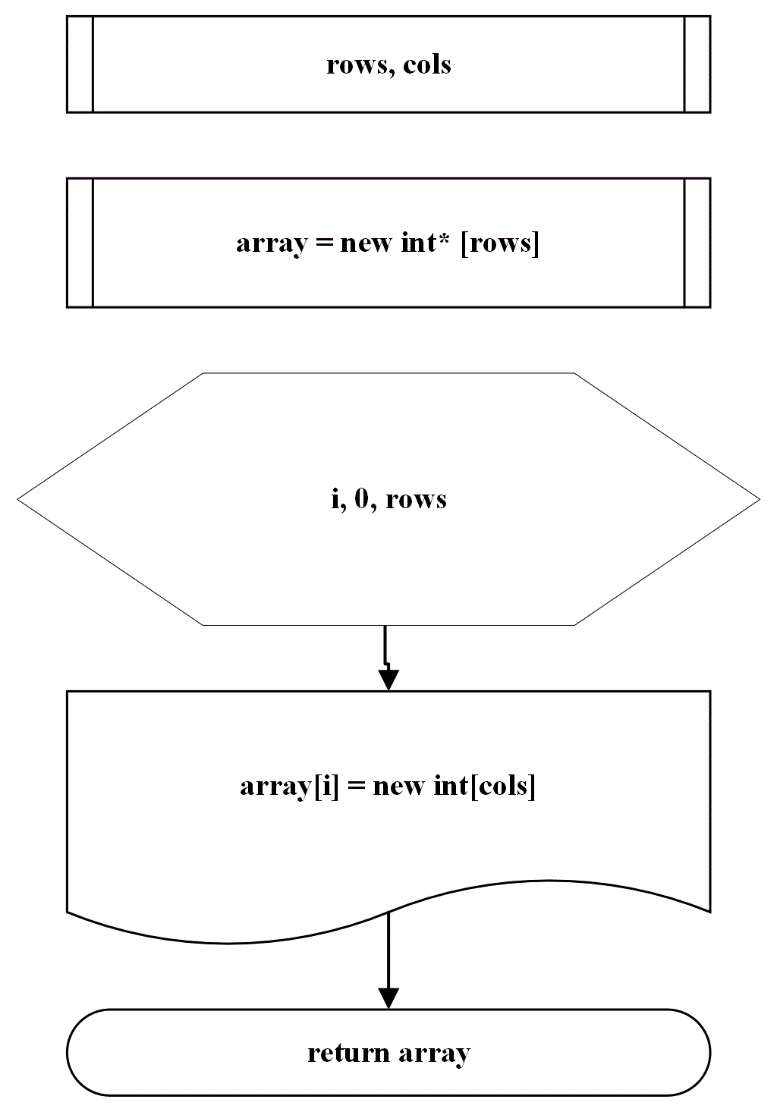
**Рисунок 8 – Блок-схема функции print\_array**

****

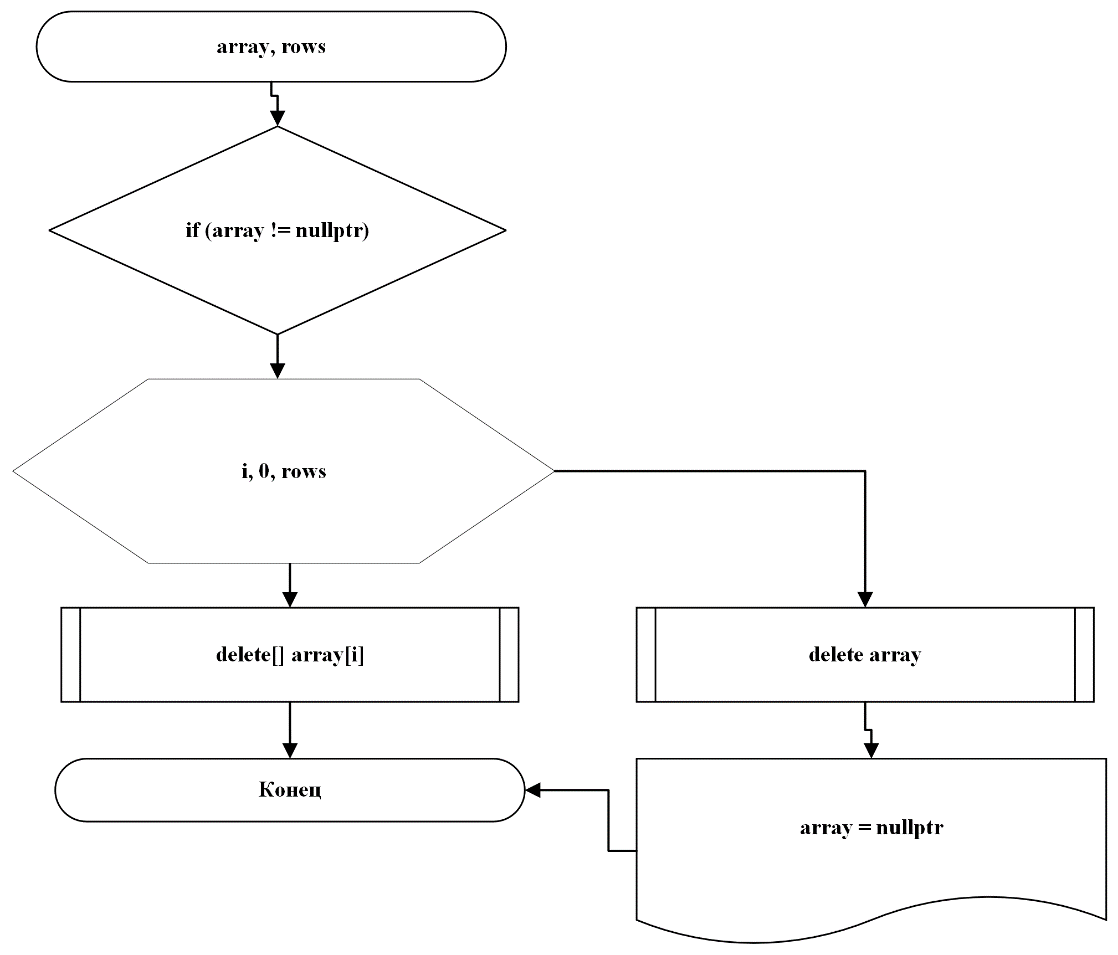
**Рисунок 9 – Блок-схема функции get\_size**

****

**Рисунок 10 – Блок-схема функции create\_array**

****

**Рисунок 11 – Блок-схема функции delete\_array**

****

**Решение задачи на языке программирования С++**

**#include <iostream>**

**#include <ctime>**

**#include <random>**

**using namespace std;**

**/\***

**\*\brief Выбор способа заполнения массива**

**\*\param c\_filling\_random - рандомные числа**

**\*\param с\_filling\_manual - с клавиатуры**

**\*/**

**enum class fillingarray {**

**c\_filling\_random,**

**c\_filling\_manual**

**};**

**/\***

**\*\brief Функция реализующая заполнение массива случайными числами**

**\*\param rows - кол-во строк массива**

**\*\param cols - кол-во столбцов массива**

**\*/**

**int\*\* filling\_random(const size\_t rows, const size\_t cols, const int min, const int max);**

**/\***

**\*\brief Фукция заполняющая массив данными с клавиатуры**

**\*\param arr указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**int filling\_manual(int\*\* arr, const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Фукция выводящая на экран массив**

**\*\param arr - указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**void print\_array(int\*\* arr, const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Фукция удаляющая массив**

**\*\param array - указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*/**

**void delete\_array(int\*\*& array, const size\_t rows);**

**/\***

**\*\brief Фукция создающая двухмерный массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*\return Возвращает указатель на массив**

**\*/**

**int\*\* create\_array(const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Функция считывающая размеры массива**

**\*\param message - подсказка для пользователя**

**\*\return Возвращает размеры массива**

**\*/**

**size\_t get\_size(const string& message);**

**/\***

**\*\brief Фукция заполняющая массив нулями**

**\*\param arr - указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**void filling\_zeros(int\*\* arr, const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Фукция заменяющая элементы первых трех столбцов их квадратами**

**\*\param arr - указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**void square\_three\_cols(int\*\* arr, const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Фукция заполняющая пустые строки элементами первой строки**

**\*\param arr - указатель на массив**

**\*\param oldarr - указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**void filling\_first\_row(int\*\* arr, int\*\* oldarr, const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Фукция считающая сколько доп строк нужно в массиве**

**\*\param arr - указатель на массив**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**int counting\_extra\_rows(int\*\* arr, const size\_t rows, const size\_t cols);**

**/\***

**\*\brief Фукция заполняющая пустые строки элементами первой строки**

**\*\param arr - указатель на массив**

**\*\param oldarr - указатель на массив**

**\*\param oldrows - кол-во строк в изначальном массиве**

**\*\param rows - кол-во строк в массиве**

**\*\param cols - кол-во столбцов в массиве**

**\*/**

**void setting\_new\_rows(int\*\* oldarr, int\*\* arr, const size\_t oldrows, const size\_t rows, const size\_t cols);**

**bool isOdd(size\_t i);**

**/\***

**\*\brief Точка входа в программу**

**\*\return возвращает 0 в случае успеха**

**\*/**

**int main() {**

**setlocale(LC\_ALL, "rus");**

**size\_t rows = 1;**

**size\_t cols = 1;**

**try {**

**rows = get\_size("Введите кол-во строк\n");**

**cols = get\_size("Введите кол-во столбцов\n");**

**}**

**catch (const std::out\_of\_range& e) {**

**std::cout << e.what();**

**return 1;**

**}**

**int input = 0;**

**int\*\* array = create\_array(rows, cols);**

**cout << "Выберите способ заполнения массива (Рандом = " << static\_cast<int>(fillingarray::c\_filling\_random) << ", самостоятельно = " << static\_cast<int>(fillingarray::c\_filling\_manual) << ") " << "\n";**

**cin >> input;**

**const auto choise = static\_cast<fillingarray>(input);**

**switch (choise) {**

**case(fillingarray::c\_filling\_random):**

**int min, max;**

**cout << "Введите минимальное случайное число\n";**

**cin >> min;**

**cout << "Введите максимальное случайное число\n";**

**cin >> max;**

**array = filling\_random(rows,**

**cols, min, max);**

**break;**

**case(fillingarray::c\_filling\_manual):**

**filling\_manual(array, rows, cols);**

**cout << "\n";**

**break;**

**default:**

**cout << "Error! Нет такой функции";**

**return 1;**

**}**

**print\_array(array, rows, cols);**

**square\_three\_cols(array, rows, cols);**

**print\_array(array, rows, cols);**

**int newrows = counting\_extra\_rows(array, rows, cols) + rows;**

**int\*\* array\_sec = create\_array(newrows, cols);**

**filling\_zeros(array\_sec, newrows, cols);**

**setting\_new\_rows(array, array\_sec, rows, newrows, cols);**

**filling\_first\_row(array\_sec, array, newrows, cols);**

**print\_array(array\_sec, newrows, cols);**

**delete\_array(array, rows);**

**delete\_array(array\_sec, newrows);**

**return 0;**

**}**

**bool isOdd(size\_t i) {**

**return i % 2 == 0;**

**}**

**void setting\_new\_rows(int\*\* oldarr, int\*\* arr, const size\_t oldrows, const size\_t rows, const size\_t cols) {**

**size\_t exrow = 0;**

**for (size\_t i = 0; i < oldrows; i++) {**

**for (size\_t j = 0; j < cols; j++) {**

**arr[i + exrow][j] = oldarr[i][j];**

**}**

**if (isOdd(i)) {**

**exrow++;**

**}**

**}**

**}**

**void filling\_first\_row(int\*\* arr, int\*\* oldarr, const size\_t rows, const size\_t cols) {**

**for (size\_t i = 0; i < rows; i++) {**

**for (size\_t j = 0; j < cols; j++) {**

**if (arr[i][j] == 0) {**

**arr[i][j] = oldarr[0][j];**

**}**

**}**

**}**

**}**

**int counting\_extra\_rows(int\*\* arr, const size\_t rows, const size\_t cols) {**

**if (isOdd(rows)) {**

**return rows / 2;**

**}**

**else {**

**return rows / 2 + 1;**

**}**

**}**

**void square\_three\_cols(int\*\* arr, const size\_t rows, const size\_t cols) {**

**for (size\_t i = 0; i < 3; i++) {**

**for (size\_t j = 0; j < rows; j++) {**

**arr[j][i] = arr[j][i] \* arr[j][i];**

**}**

**}**

**}**

**int\*\* filling\_random(const size\_t rows, const size\_t cols, const int min, const int max) {**

**int\*\* arr = create\_array(rows, cols);**

**for (size\_t i = 0; i < rows; i++) {**

**for (size\_t j = 0; j < cols; j++) {**

**random\_device rd;**

**mt19937 gen(rd());**

**uniform\_int\_distribution<> uniformIntDistribution(min, max);**

**arr[i][j] = uniformIntDistribution(gen);**

**}**

**}**

**return arr;**

**}**

**void filling\_zeros(int\*\* arr, const size\_t rows, const size\_t cols)**

**{**

**for (size\_t i = 0; i < rows; i++) {**

**for (size\_t j = 0; j < cols; j++) {**

**arr[i][j] = 0;**

**}**

**}**

**}**

**int filling\_manual(int\*\* arr, const size\_t rows, const size\_t cols) {**

**cout << "Введите элементы массива" << "\n";**

**for (size\_t i = 0; i < rows; i++) {**

**for (size\_t j = 0; j < cols; j++) {**

**cin >> arr[i][j];**

**}**

**}**

**}**

**void print\_array(int\*\* arr, const size\_t rows, const size\_t cols) {**

**for (size\_t i = 0; i < rows; i++) {**

**for (size\_t j = 0; j < cols; j++) {**

**cout << arr[i][j] << "\t ";**

**}**

**cout << "\n";**

**}**

**cout << '\n';**

**}**

**size\_t get\_size(const string& message) {**

**int size = -1;**

**cout << message;**

**cin >> size;**

**if (size < 0)**

**{**

**throw out\_of\_range("Incorrect size");**

**}**

**return size;**

**}**

**int\*\* create\_array(const size\_t rows, const size\_t cols) {**

**int\*\* array = new int\* [rows];**

**for (size\_t i = 0; i < rows; i++) {**

**array[i] = new int[cols];**

**}**

**return array;**

**}**

**void delete\_array(int\*\*& array, const size\_t rows) {**

**if (array != nullptr) {**

**for (size\_t i = 0; i < rows; i++) {**

**delete[] array[i];**

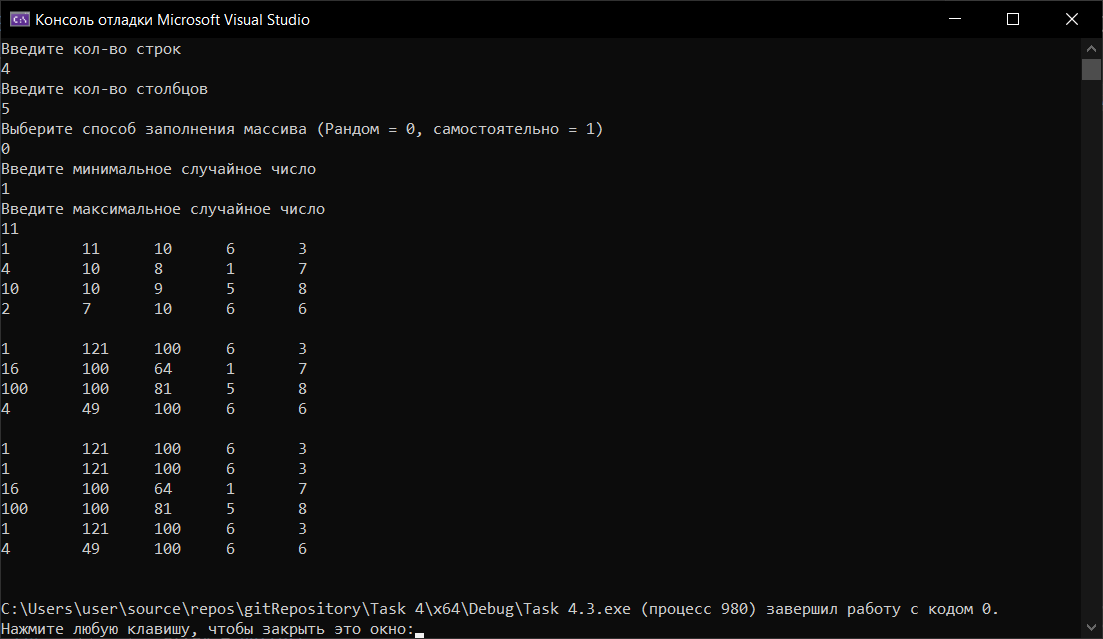
**}**

**delete array;**

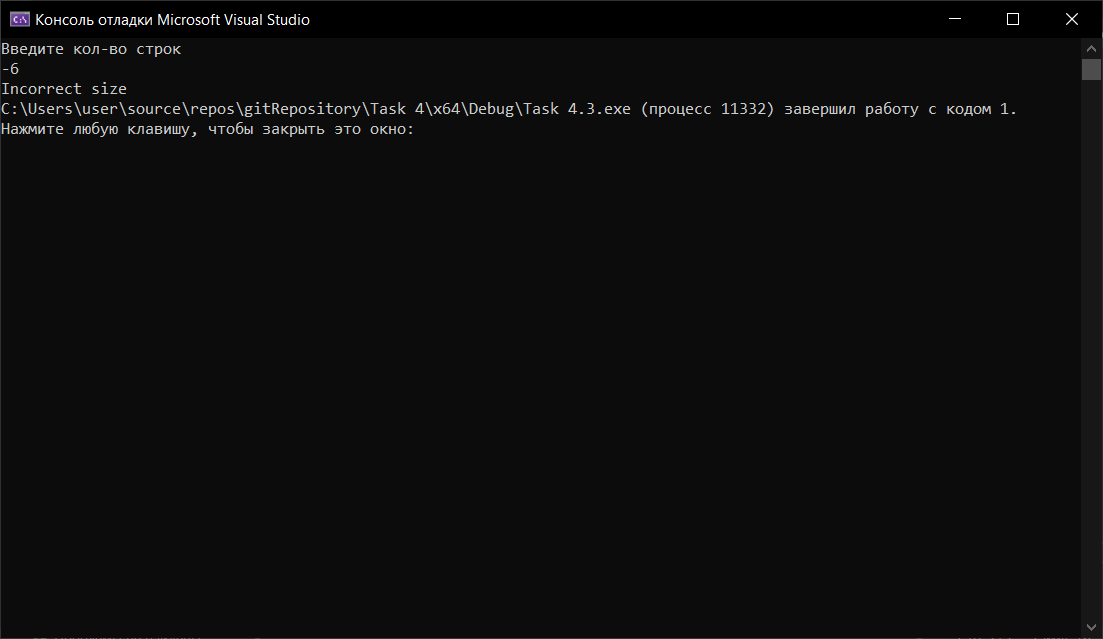
**array = nullptr;**

**}**

**3} Решение тестовых примеров**

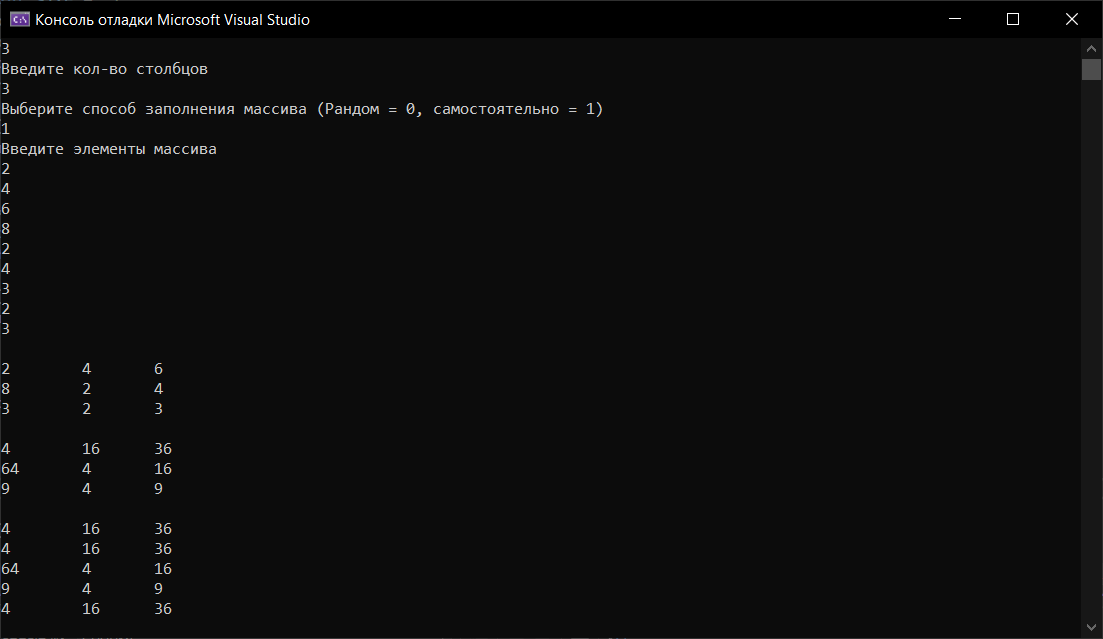


**Рисунок 12. Решение тестового примера**



**Рисунок 13. Отрицательный размер массива**

**Рисунок 14. Заполнение массива ручное**



**Зачет задания в GitHub**

