Министерство образования и науки России

Федеральное государственное автономное образовательное

учреждение высшего образования

“Санкт-Петербургский государственный электротехнический

университет им. В. И. Ульянова (Ленина)”

(СПБГЭТУ “ЛЭТИ”)

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

**ОТЧЕТ**

**по лабораторной работе № 5 на тему**

**“Функции языка С / C++” по**

**дисциплине “Программирование”**

**Выполнил:** *Студент группы 5307 Грицай Н. Д.*

**Проверила:** *к.т.н., доцент Сискович Т.И.*

**Подпись преподавателя:\_\_\_\_\_\_\_\_**

Санкт-Петербург

2015

Оглавление

[Цель работы 3](#_Toc434764990)

[1.Задание 3](#_Toc434764991)

[2.Уточнение задания 3](#_Toc434764992)

[3.Контрольные примеры 3](#_Toc434764993)

[4.Описание переменных главной функции 4](#_Toc434764994)

[5.Схема главной функции 4](#_Toc434764995)

[6.Описание функций 4](#_Toc434764996)

[6.1.Функция menu 4](#_Toc434764997)

[6.2.Функция matrixi 4](#_Toc434764998)

[6.3.Функция resmatrix 5](#_Toc434764999)

[6.4.Функция matrixo 7](#_Toc434765000)

[6.5.Функция reference 8](#_Toc434765001)

[6.6.Функция dsfree 8](#_Toc434765002)

[6.7.Функция ismatch 9](#_Toc434765003)

[6.8.Функция addcopystr 10](#_Toc434765004)

[7.Иерархия вызовов функций программы 11](#_Toc434765005)

[8.Краткое описание алгоритма 12](#_Toc434765006)

[9.Исходный код программы 12](#_Toc434765007)

[10.Результат 16](#_Toc434765008)

[Вывод 16](#_Toc434765009)

[*Приложение* 17](#_Toc434765010)

# Цель работы

Получить практические навыки использования функций языка С / C++.

# 1.Задание

Разработать алгоритм и написать программу, выполняющую многократно по желанию пользователя ввод матрицы размера mxn, контрольный вывод, обработку исходной матрицы и вывод результата.

Результирующая матрица формируется из строк исходной, в которых есть хотя бы 2 совпадающих элемента.

# 2.Уточнение задания

В программе матрицу реализовать в виде динамической структуры.

# 3.Контрольные примеры

Контрольные примеры представлены в таблице 1.

*Таблица 1. Контрольные примеры*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ примера** | **Исходные данные** | | | **Результат** |
| **Кол-во строк** | **Кол-во столбцов** | **Матрица arr1** | **Матрица arr2** |
| 1 | 1 | 2 | 2 3 | “Матрица не сформирована” |
| 2 | 2 | 2 | 1 2  1 1 | 1 1 |
| 3 | 2 | 3 | 2 3 4  1 1 5 | 1 1 5 |
| 4 | 1 | 5 | 1 2 4 535 535 | 1 2 4 535 535 |
| 5 | 2 | 1 | 0  0 | “Матрица не сформирована” |
| 6 | 1 | 3 | 0 1 0 | 0 1 0 |

# 4.Описание переменных главной функции

Описание переменных представлено в таблице 2.

*Таблица 2. Описание переменных главной функции*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки исходной матрицы |
| arr2 | int\*\* | Указатель на адрес первого элемента первой строки результирующей матрицы |
| clmct | short | Количество столбцов исходной и результирующей матриц |
| str1ct, str2ct | short | Количество строк исходной и результирующей матриц |
| isarr1, isarr2 | short | Флажки наличия исходной и результирующей матриц(1–есть/0-нет) |
| isprogend | short | Флажок выхода из программы(1-выйти/0-нет) |

# 5.Схема главной функции

Схема главной функции представлена на рисунках 1.1 и 1.2 в Приложении.

# 6.Описание функций

## 6.1.Функция menu

**Назначение функции:** вывод меню на экран

**Прототип функции:** char menu();

**Возврат функции:** символ, введенный пользователем

**Пример вызова функции:** char c = menu();

**Схема функции:** схема функции menu представлена на рисунке 2 в Приложении

## 6.2.Функция matrixi

**Назначение функции:** ввод матрицы

**Прототип функции:** int \*\*matrixi(short \*strct, short \*clmct);

**Возврат функции:** указатель на адрес первого элемента первой строки матрицы

**Описание формальных переменных функции:** описание формальных переменных функции

matrixi представлено в таблице 3

**Пример вызова функции:** arr1 = matrixi(&str1ct, &clmct);

**Описание фактических параметров:** описание фактических параметров функции matrixi представлено в таблице 5

**Описание локальных переменных функции:** описание локальных переменных функции matrixi представлено в таблице 4

**Схема функции:** схема функции matrixi представлена на рисунке 3 в Приложении

*Таблица 3. Описание формальных переменных функции matrixi*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| strct | short\* | Адрес переменной, хранящей количество строк матрицы |
| clmct | short\* | Адрес переменной, хранящей количество столбцов матрицы |

*Таблица 4. Описание локальных переменных функции matrixi*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки матрицы |

*Таблица 5. Описание фактических параметров функции matrixi*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| &str1ct | short\* | Адрес переменной, хранящей количество строк исходной матрицы |
| &clmct | short\* | Адрес переменной, хранящей количество столбцов исходной и результирующей матриц |

## 6.3.Функция resmatrix

**Назначение функции:** формирование результирующей матрицы

**Прототип функции:** int \*\*resmatrix(int \*\*arr1, short str1ct, short clmct, short \*str2ct);

**Возврат функции:** указатель на адрес первого элемента первой строки результирующей матрицы

**Описание формальных переменных функции:** описание формальных переменных функции resmatrix представлено в таблице 6

**Пример вызова функции:** arr2 = resmatrix(arr1, str1ct, clmct, &str2ct);

**Описание фактических параметров функции:** описание фактических параметров функции resmatrix представлено в таблице 8

**Описание локальных переменных функции:** описание локальных переменных функции resmatrix представлено в таблице 7

**Схема функции:** схема функции resmatrix представлена на рисунке 4 в Приложении

*Таблица 6. Описание формальных переменных функции resmatrix*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки исходной матрицы |
| str1ct | short | Количество строк исходной матрицы |
| clmct | short | Количество столбцов исходной и результирующей матриц |
| &str2ct | short\* | Адрес переменной, хранящей количество строк результирующей матрицы |

*Таблица 7. Описание локальных переменных функции resmatrix*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr2 | int\*\* | Указатель на адрес первого элемента первой строки результирующей матрицы |
| i | short | Вспомогательная переменная |

*Таблица 8. Описание фактических параметров функции resmatrix*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки исходной матрицы |
| str1ct | short | Количество строк исходной матрицы |
| clmct | short | Количество столбцов исходной и результирующей матриц |
| &str2ct | short\* | Адрес переменной, хранящей количество строк результирующей матрицы |

## 6.4.Функция matrixo

**Назначение функции:** вывод на экран матрицы

**Прототип функции:** void matrixo(int \*\*arr, short strct, short clmct);

**Описание формальных переменных функции:** описание формальных переменных функции matrixo представлено в таблице 9

**Пример вызова функции:** matrixo(arr1, str1ct, clmct);

**Описание фактических параметров функции:** описание фактических параметров функции matrixo представлено в таблице 11

**Описание локальных переменных функции:** описание локальных переменных функции matrixo представлено в таблице 10

**Схема функции:** схема функции matrixo представлена на рисунке 5 в Приложении

*Таблица 9. Описание формальных переменных функции matrixo*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr | int\*\* | Указатель на адрес первого элемента первой строки матрицы |
| strct | short | Количество строк матрицы |
| clmct | short | Количество столбцов матрицы |

*Таблица 10. Описание локальных переменных функции matrixo*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i, j | short | Вспомогательные переменные |

*Таблица 11. Описание фактических параметров функции matrixo*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки исходной матрицы |
| str1ct | short | Количество строк исходной матрицы |
| clmct | short | Количество столбцов исходной матрицы |

## 6.5.Функция reference

**Назначение функции:** вывод справки на экран

**Прототип функции:** void reference();

**Пример вызова функции:** reference();

**Схема функции:** схема функции reference представлена на рисунке 6 в Приложении

## 6.6.Функция dsfree

**Назначение функции:** освобождение памяти, выделенной под динамическую структуру

**Прототип функции:** int \*\*dsfree(int \*\*arr, short strct);

**Возврат функции:** NULL

**Описание формальных переменных функции:** описание формальных переменных функции dsfree представлено в таблице 12

**Пример вызова функции:** arr1 = dsfree(arr1, str1ct);

**Описание фактических параметров функции:** описание фактических параметров функции dsfree представлено в таблице 14

**Описание локальных переменных функции:** описание локальных переменных функции dsfree представлено в таблице 13

**Схема функции:** схема функции dsfree представлена на рисунке 7 в Приложении

*Таблица 12. Описание формальных переменных функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки динамической структуры |
| strct | short | Количество первых индексов динамической структуры |

*Таблица 13. Описание локальных переменных функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i | short | Вспомогательная переменная |

*Таблица 14. Описание фактических параметров функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| arr1 | int\*\* | Указатель на адрес первого элемента первой строки исходной матрицы |
| str1ct | short | Количество строк исходной матрицы |

## 6.7.Функция ismatch

**Назначение функции:** проверка наличия совпадающих элементов в массиве или динамической структуре размерности один

**Прототип функции:** bool ismatch(int \*arr, short strlen);

**Возврат функции:** 1, если есть совпадающие элементы и ноль в противном случае

**Описание формальных переменных функции:** описание формальных переменных функции ismatch представлено в таблице 15

**Пример вызова функции:** bool istrue = ismatch(\*(arr1 + i), clmct);

**Описание фактических параметров функции:** описание фактических параметров функции ismatch представлено в таблице 17

**Описание локальных переменных функции:** описание локальных переменных функции ismatch представлено в таблице 16

**Схема функции:** схема функции ismatch представлена на рисунке 8 в Приложении

*Таблица 15. Описание формальных переменных функции ismatch*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr | int\* | Адрес первого элемента массива или динамической структуры |
| strlen | short | Длина массива |

*Таблица 16. Описание локальных переменных функции ismatch*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| isres | short | Флажок результата (1-в массиве есть совпадающие элементы/0-нет) |
| j, k | short | Вспомогательные переменные |

*Таблица 17. Описание фактических параметров функции ismatch*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| \*(arr1 + i) | int\* | Адрес первого элемента i-той строки исходной матрицы |
| clmct | short | Количество столбцов исходной матрицы |

## 6.8.Функция addcopystr

**Назначение функции:** добавление новой строки в матрицу и копирование в нее элементов массива или динамической структуры размерности один

**Прототип функции:** int \*\*addcopystr(int \*\*arr2, int \*arr1, short \*strct, short clmct);

**Возврат функции:** указатель на адрес первого элемента первой строки матрицы

**Описание формальных переменных функции:** описание формальных переменных функции addcopystr представлено в таблице 18

**Пример вызова функции:** arr2 = addcopystr(arr2, \*(arr1 + i), str2ct, clmct);

**Описание фактических параметров функции:** описание фактических параметров функции addcopystr представлено в таблице 20

**Описание локальных переменных функции:** описание локальных переменных функции addcopystr представлено в таблице 19

**Схема функции:** схема функции addcopystr представлена на рисунке 9 в Приложении

*Таблица 18. Описание формальных переменных функции addcopystr*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| arr2 | int\*\* | Указатель на адрес первого элемента первой строки матрицы |
| arr1 | int\* | Адрес первого элемента массива или динамической структуры |
| strct | short\* | Адрес переменной, хранящей количество строк матрицы |
| clmct | short | Количество столбцов матрицы и размер массива или динамической структуры |

*Таблица 19. Описание локальных переменных функции addcopystr*

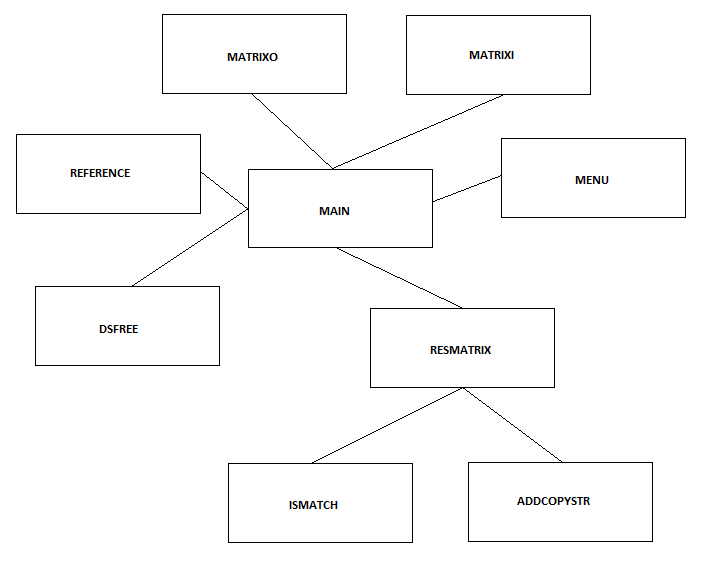
|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| j | short | Вспомогательная переменная |

*Таблица 20. Описание формальных параметров функции addcopystr*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| arr2 | int\*\* | Указатель на адрес первого элемента первой строки результирующей матрицы |
| \*(arr1 + i) | int\* | Адрес первого элемента i-той строки исходной матрицы |
| str2ct | short\* | Адрес переменной, хранящей количество строк результирующей матрицы |
| clmct | short | Количество столбцов результирующей матрицы и размер i-той строки исходной матрицы |

# 7.Иерархия вызовов функций программы

Иерархия вызовов функций программы представлена на рисунке 10



*Рисунок 10. Иерархия вызовов функций программы*

# 8.Краткое описание алгоритма

1. Вывод меню на экран и переход к пункту, выбранному пользователем с помощью функции menu:

1)Ввод матрицы с помощью функции matrixi. Переход в пункт 2

2)Вывод исходной матрицы с помощью функции matrix или сообщение о том, что ее нет. Переход в пункт 2

3)Обработка исходной матрицы с помощью функции resmatrix, сообщение о том, что нет исходной матрицы или сообщение о том, что матрица уже обработана. Переход в пункт 2

4)Вывод результирующей матрицы с помощью функции matrix или сообщение о том, что ее нет. Переход в пункт 2

5)Вывод справки на экран с помощью функции reference. Переход в пункт 2

6)Ввод isprogend

Если пользователь выбрал неверный пункт меню, то сообщение об этом

1. Если(!isprogend), то: переход в пункт 1, иначе: выход из программы

# 9.Исходный код программы

#include "stdio.h"

#include "Windows.h"

#include "conio.h"

//вывод на экран меню и возврат выбранного пользователем пункта

char menu();

//ввод матрицы и возврат указателя на адрес ее первого элемента

int \*\*matrixi(short \*strct, short \*clmct);

//Формирование результирующей матрицы и возврат указателя на адрес ее первого элемента

int \*\*resmatrix(int \*\*arr1, short str1ct, short clmct, short \*str2ct);

//вывод на экран матрицы

void matrixo(int \*\*arr, short strct, short clmct);

//вывод на экран справки

void reference();

//освобождение памяти динамической структуры и возврат NULL

int \*\*dsfree(int \*\*arr, short strct);

//проверка наличия совпадающих элементов массива

bool ismatch(int \*arr, short strlen);

//добавление новой строки результирующей матрицы

int \*\*addcopystr(int \*\*arr2, int \*arr1, short \*strct, short clmct);

void NDGlogo(char str[59]);

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int \*\*arr1 = NULL, \*\*arr2 = NULL;

short clmct, str1ct, str2ct, isarr1 = 0, isarr2 = 0, isprogend = 0;

do

{

system("title 1.5 Динамическая Матрица");

system("cls");

switch (menu())

{

case '1': //ввод матрицы

system("cls");

printf\_s("==Ввод матрицы==\n");

isarr1 = 1;

isarr2 = 0;

if (arr1 != NULL)

arr1 = dsfree(arr1, str1ct);

if (arr2 != NULL)

arr2 = dsfree(arr2, str2ct);

arr1 = matrixi(&str1ct, &clmct);

printf\_s("\nВвод завершен!\nДля продолжения нажмите клавишу Enter");

rewind(stdin);

getchar();

break;

case '2': //вывод исходной матрицы

system("cls");

if (isarr1)

{

printf\_s("==Исходная матрица==\n");

matrixo(arr1, str1ct, clmct);

}

else

printf\_s("Ошибка! Сначала введите матрицу");

printf\_s("\nДля продолжения нажмите клавишу Enter");

rewind(stdin);

getchar();

break;

case '3': //обработка исходной матрицы

system("cls");

if (isarr1)

{

if (!isarr2)

{

isarr2 = 1;

arr2 = resmatrix(arr1, str1ct, clmct, &str2ct);

}

else

printf\_s("Матрица уже обработана!");

}

else

printf\_s("Ошибка! Сначала введите матрицу");

printf\_s("\nДля продолжения нажмите клавишу Enter");

rewind(stdin);

getchar();

break;

case '4': //вывод результирующей матрицы

system("cls");

if (isarr2)

{

if (str2ct)

{

printf\_s("==Результирующая матрица==\n");

matrixo(arr2, str2ct, clmct);

}

else

printf\_s("В результирующей матрице нет ни одной

строки.\n");

}

else

printf\_s("Ошибка! Сначала обработайте исходную матрицу");

printf\_s("\nДля продолжения нажмите клавишу Enter");

rewind(stdin);

getchar();

break;

case '5': //справка

system("cls");

reference();

rewind(stdin);

getchar();

break;

case '6': //выход

system("cls");

printf\_s("Вы уверены, что хотите выйти?(1-Да/0-Нет): ");

scanf\_s("%hi", &isprogend);

rewind(stdin);

break;

default:

printf\_s("Ошибка! Введите существующий пункт меню\n\n");

rewind(stdin);

getchar();

}

} while (!isprogend);

if(arr1 != NULL)

arr1 = dsfree(arr1, str1ct);

if(arr2 != NULL)

arr2 = dsfree(arr2, str2ct);

return 0;

}

//Описание: вывод меню на экран

//Возврат: символ, введенный пользователем

char menu()

{

printf\_s("==Меню==\n");

printf\_s("1.Ввод матрицы.\n");

printf\_s("2.Вывод исходной матрицы\n");

printf\_s("3.Обработка исходной матрицы\n");

printf\_s("4.Вывод результирующей матрицы\n");

printf\_s("5.Справка\n");

printf\_s("6.Выход\n");

printf\_s("\nВведите пункт меню: ");

rewind(stdin);

return getchar();

}

//Описание: ввод матрицы

//Возврат: указатель на адрес первого элемента матрицы

int \*\*matrixi(short \*strct, short \*clmct)

{

int \*\*arr;

//ввод кол-ва строк

do

{

printf("Введите количество строк исходной матрицы: ");

scanf\_s("%hi", strct);

if (\*strct < 1)

printf\_s("Ошибка! Количество столбцов не может быть меньше 1\n");

} while (\*strct < 1);

//ввод кол-ва столбцов

do

{

printf("Введите количество столбцов исходной матрицы: ");

scanf\_s("%hi", clmct);

if (\*clmct < 1)

printf\_s("Ошибка! Количество строк не может быть меньше 1\n");

} while (\*clmct < 1);

//формирование исходной матрицы

arr = (int\*\*)malloc(sizeof(int\*) \* \*strct);

for (short i = 0; i < \*strct; i++)

{

\*(arr + i) = (int\*)malloc(sizeof(int) \* \*clmct);

for (short j = 0; j < \*clmct; j++)

{

printf\_s("Введите элемент[%hi][%hi]: ", i + 1, j + 1);

scanf\_s("%d", \*(arr + i) + j);

}

}

return arr;

}

//Описание: вывод матрицы

void matrixo(int \*\*arr, short strct, short clmct)

{

for (short i = 0; i < strct; i++)

{

for (short j = 0; j < clmct; j++)

printf\_s("%5d", \*(\*(arr + i) + j));

printf\_s("\n");

}

}

//Описание: вывод справки на экран

void reference()

{

printf\_s("==Справка==\n");

printf\_s("-Результирующая матрица формируется из строк новой, в которых есть

совпадающие элементы.\n");

printf\_s("\nДля продолжения нажмите клавишу Enter");

}

//Описание: освобождение памяти двумерной динамической структуры

//Возврат: указатель на адрес динамичекой структуры

int \*\*dsfree(int \*\*arr, short strct)

{

for (short i = 0; i < strct; i++)

free(\*(arr + i));

free(arr);

return NULL;

}

//Описание: формирование результирующей матрицы

//Возврат: указатель на адрес первого элемента результирующей матрицы

int \*\*resmatrix(int \*\*arr1, short str1ct, short clmct, short \*str2ct)

{

int \*\*arr2 = NULL;

\*str2ct = 0;

for (short i = 0; i < str1ct; i++)

{

if (ismatch(\*(arr1 + i), clmct))

{

arr2 = addcopystr(arr2, \*(arr1 + i), str2ct, clmct);

}

}

printf\_s("Результирующая матрица сформирована");

return arr2;

}

//Описание: проверка на наличие совпадающих элементов в массиве

//Возврат: 1 если есть совпадающие элементы, 0 в противном случае

bool ismatch(int \*arr, short strlen)

{

bool isres = 0;

for (short j = 0; (j < strlen - 1) && !isres; j++)

for (short k = j + 1; (k < strlen) && !isres; k++)

if (\*(arr + j) == \*(arr + k))

isres = 1;

return isres;

}

//Описание: Добавляет в матрицу новую строку и копирует в нее элементы исходной строки

//Возврат: Указатель на адрес первого элемента матрицы

int \*\*addcopystr(int \*\*arr2, int \*arr1, short \*strct, short clmct)

{

arr2 = (int\*\*)realloc(arr2, sizeof(int\*) \* (\*strct + 1));

\*(arr2 + \*strct) = (int\*)malloc(sizeof(int) \* clmct);

for (short j = 0; j < clmct; j++)

\*(\*(arr2 + \*strct) + j) = \*(arr1 + j);

\*strct = \*strct + 1;

return arr2;

}

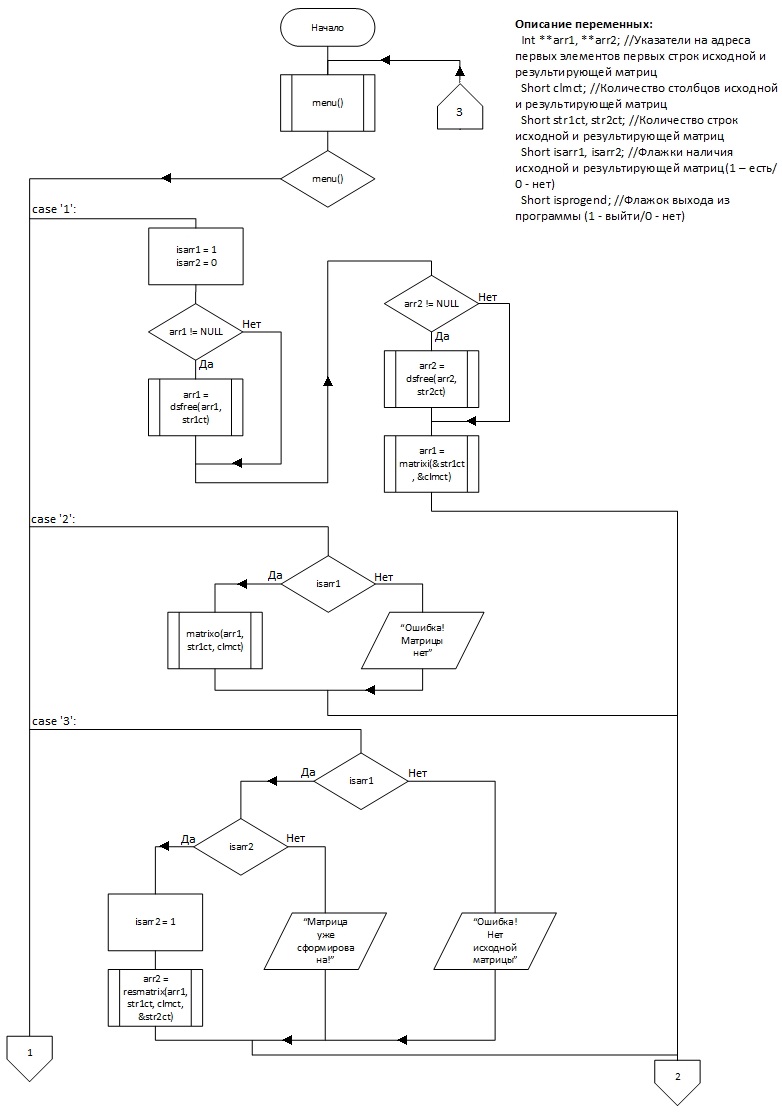
# 10.Результат

При выполнении программы получены результаты, совпадающие с контрольными примерами, представленными в таблице 1. Ошибки не обнаружены.

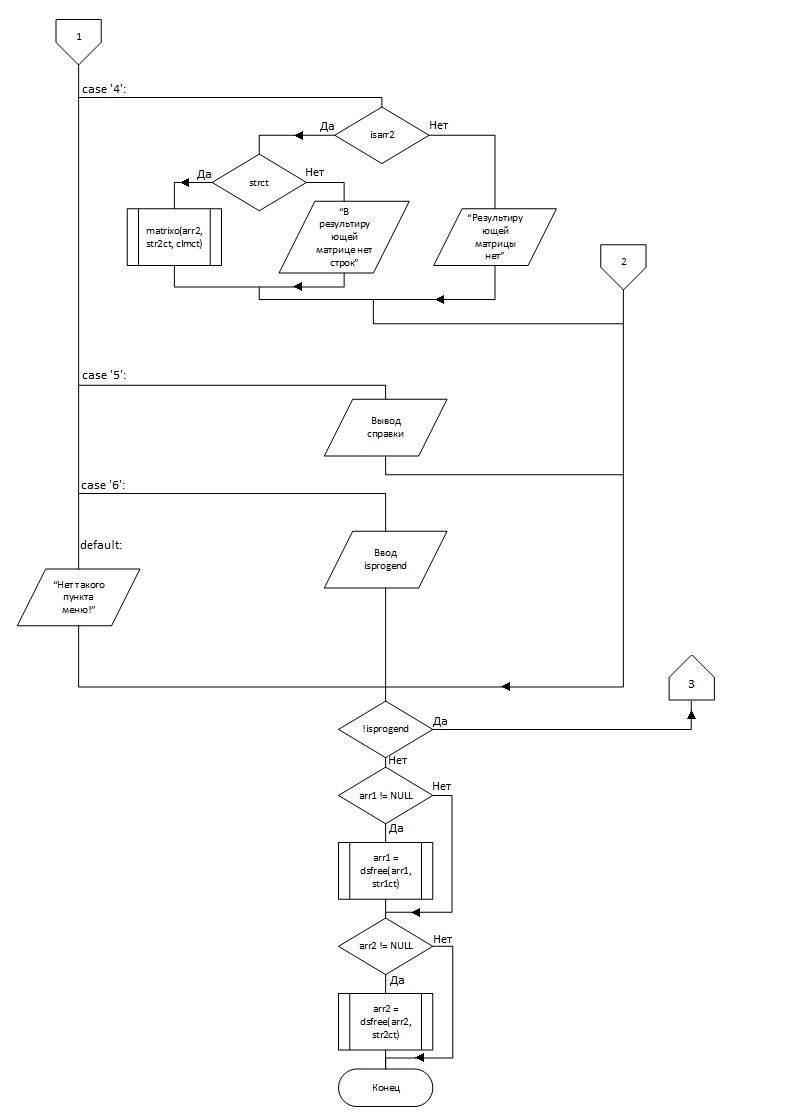
# Вывод

В ходе выполнения работы были получены практические навыки использования функций языка С / C++.

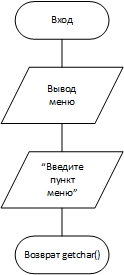
# *Приложение*



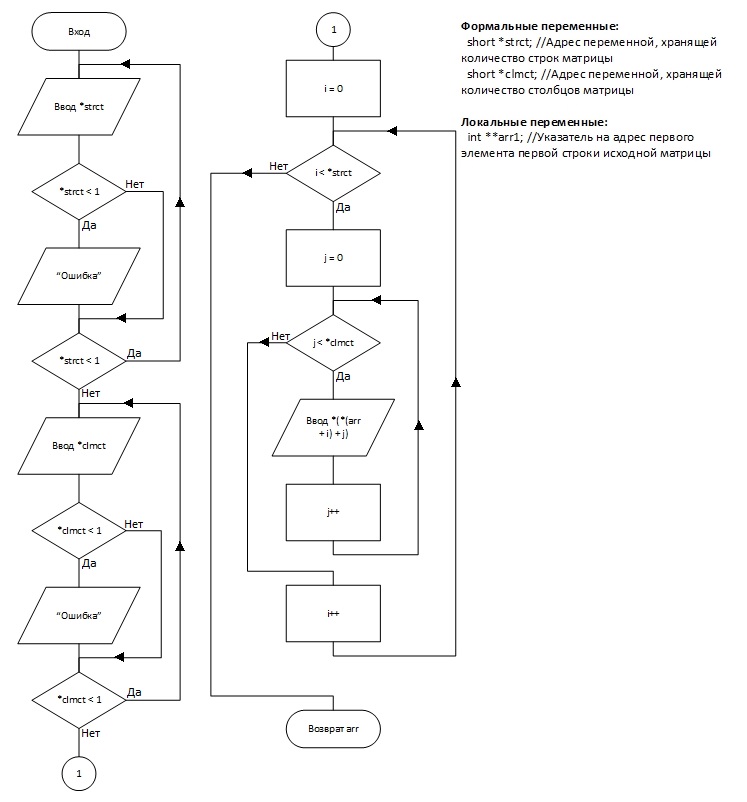
*Рисунок 1.1. Схема главной функции*

**

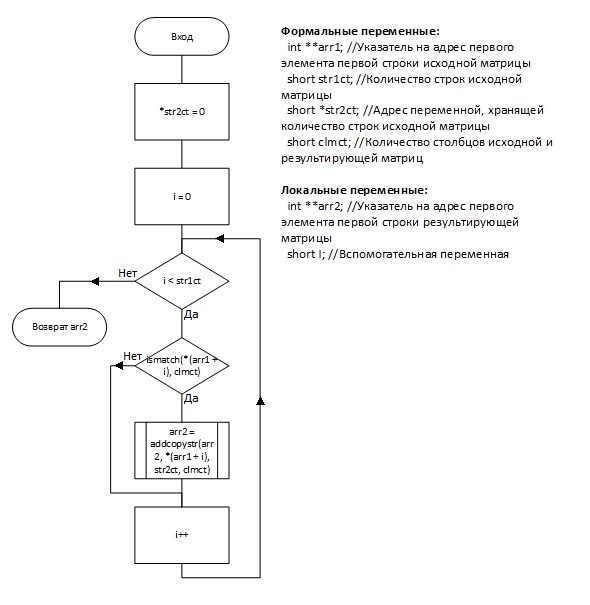
*Рисунок 1.2. Схема главной функции*

**

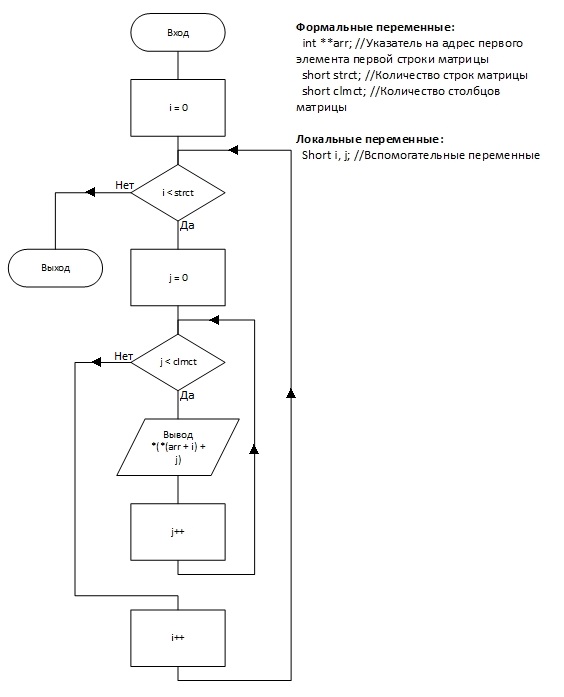
*Рисунок 2. Схема функции menu*

**

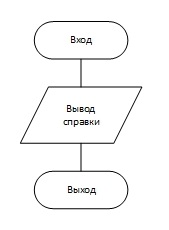
*Рисунок 3. Схема функции matrixi*

**

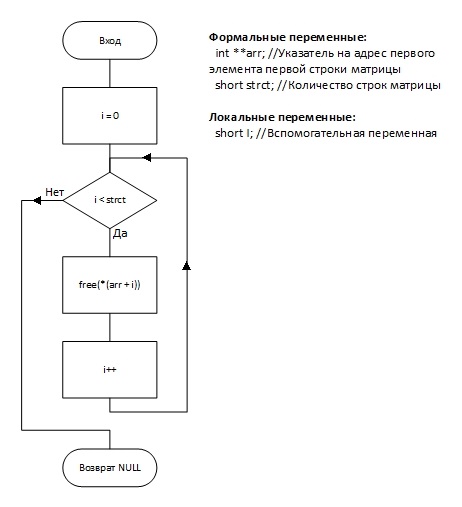
*Рисунок 4. Схема функции resmatrix*

**

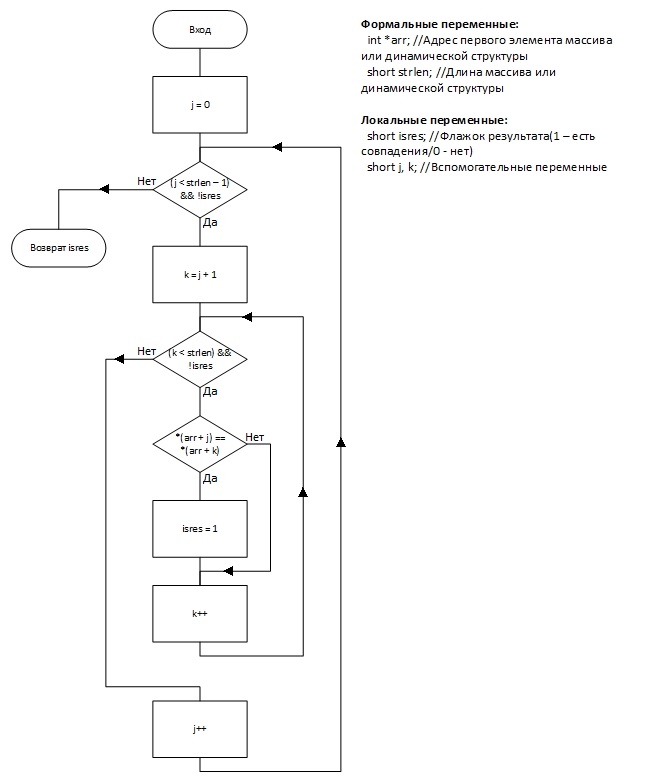
*Рисунок 5. Схема функции matrixo*

**

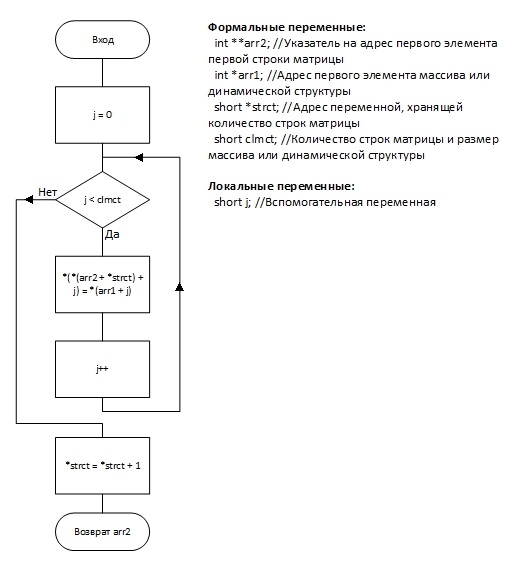
*Рисунок 6. Схема функции reference*

**

*Рисунок 7. Схема функции dsfree*

**

*Рисунок 8. Схема функции ismatch*

**

*Рисунок 9. Схема функции addcopystr*