МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5

по дисциплине: Основы программирования тема: «Использование функций при работе с двумерными массивами»

Выполнил: ст. группы ПВ-201 Машуров Дмитрий Русланович

Проверил:

Притчин Иван Сергеевич Брусенцева Валентина

Станиславовна

Белгород 2021 г.

**Лабораторная работа №5**

«Использование функций при работе с двумерными массивами»

#### Цель работы: получение навыков работы с функциями и двумерными массивами

#### Задания для подготовки к работе

1. Изучить способы описания и инициализации многомерных массивов, правила передачи массивов функциям.
2. Разбить задачу соответствующего варианта на подзадачи таким образом, чтобы решение каждой подзадачи описывалось функцией, а основная программа состояла бы из последовательности вызовов функций. Размеры матриц задать константами.

5. Для каждой подзадачи описать спецификацию и блок-схему алгоритма. Спецификация содержит заголовок функции и ее назначение, из которого должен быть понятен смысл каждого параметра.

6. Подобрать наборы тестовых данных.

**Задание варианта №17**

Дана квадратная матрица. Определить *k* – количество "особых" элементов матрицы, считая элемент "особым", если он больше суммы остальных элементов своего столбца.

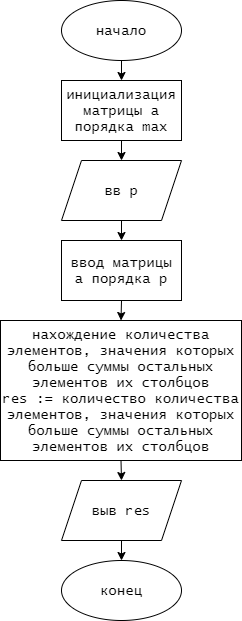
**Выполнение работы:**

1. **Описание алгоритма и выделение подзадач**

Исходя из условия задачи, нам нужно находить сумму столбца матрицы и сравнивать каждый элемент с суммой своего столбца матрицы, который не включает в себя данный элемент ()

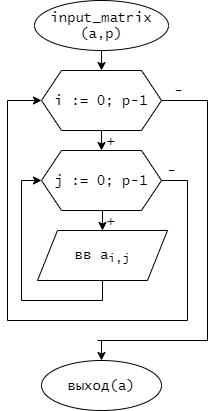
Выделим следующие подзадачи:

1. Нахождение суммы элементов столбца матрицы
2. Нахождение количества элементов, значения которых больше суммы остальных элементов их столбца
3. Ввод матрицы
4. **Блок-схема с укрупнёнными блоками**

****

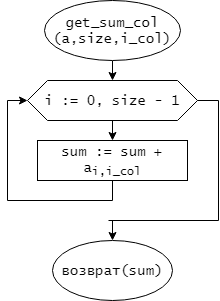
1. **Спецификации функций**
2. Ввод матрицы
   1. Заголовок: void input\_matrix(int a[][MAX], size\_t p)
   2. Назначение: ввод матрицы a порядка p

Блок-схема:



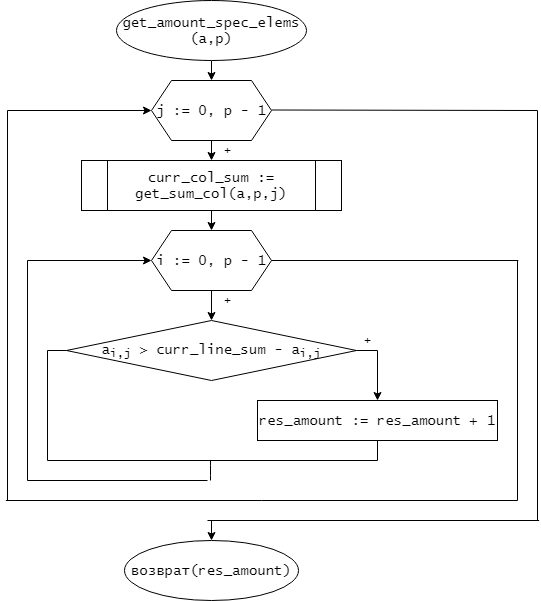
1. Нахождение суммы элементов столбца матрицы
   1. Заголовок: int get\_sum\_col(const int a[], size\_t size)
   2. Назначение: возвращает сумму элементов столбца под индексом i\_col матрицы a порядка size

Блок-схема:



1. Нахождение количества элементов, значения которых больше суммы остальных элементов их столбца
   1. Заголовок: size\_t get\_amount\_spec\_elems(const int a[][MAX], size\_t p)
   2. Назначение: возвращает количество элементов, каждый из которых превышает сумму остальных элементов своего столбца, матрицы a порядка p

Блок-схема:



1. **Тестовые данные**

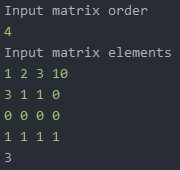
|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Вход** | **Выход** |
| **1** | p = 4  1 2 3 10  3 1 1 0  0 0 0 0  1 1 1 1 | 3 |
| **2** | p = 3  1 2 3  3 2 1  2 2 2 | 0 |
| **3** | p = 3  -1 2 4  -1 -2 3  -1 -1 -1 | 5 |

1. **Текст программы**

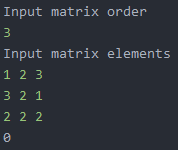
#### /\* \* Дана квадратная матрица. Определить k – количество "особых" \* элементов матрицы, считая элемент "особым", если он больше \* суммы остальных элементов своего столбца \*/ #include <stdio.h> #include <stdlib.h> #define MAX 100 /\* ввод матрицы a порядка p \*/ void input\_matrix(int a[][MAX], size\_t p) { for (size\_t i = 0; i < p; i++) { for (size\_t j = 0; j < p; j++) { scanf("%d", &a[i][j]); } } } /\* \* возвращает сумму элементов столбца под индексом i\_col \* матрицы a порядка size \*/ int get\_sum\_col(const int a[][MAX], size\_t size, size\_t i\_col) { int sum = 0; for (size\_t i = 0; i < size; i++) { sum += a[i][i\_col]; } return sum; } /\* \* возвращает количество элементов, каждый из которых \* превышает сумму остальных элементов своего столбца, матрицы a порядка p \*/ size\_t get\_amount\_spec\_elems(const int a[][MAX], size\_t p) { size\_t res\_amount = 0; int curr\_col\_sum; for (size\_t j = 0; j < p; j++) { curr\_col\_sum = get\_sum\_col(a, p, j); for (size\_t i = 0; i < p; i++) { if (a[i][j] > curr\_col\_sum - a[i][j]) res\_amount++; } } return res\_amount; } int main() { size\_t p; printf("Input matrix order\n"); scanf("%u", &p); printf("Input matrix elements\n"); int a[MAX][MAX]; input\_matrix(a, p); size\_t res = get\_amount\_spec\_elems(a, p); printf("%u", res); }

1. **Результаты работы**

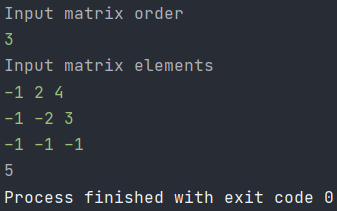
*Пример №1*



*Пример №2*



*Пример №3*



1. **Анализ ошибок**

* Изначально инициализировал матрицу динамически – невнимательное прочтение условия задания