ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

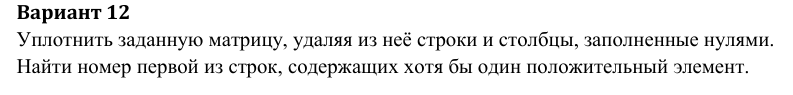
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| Обработка числовых матриц |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4135К |  |  |  | Столяров Н.С. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

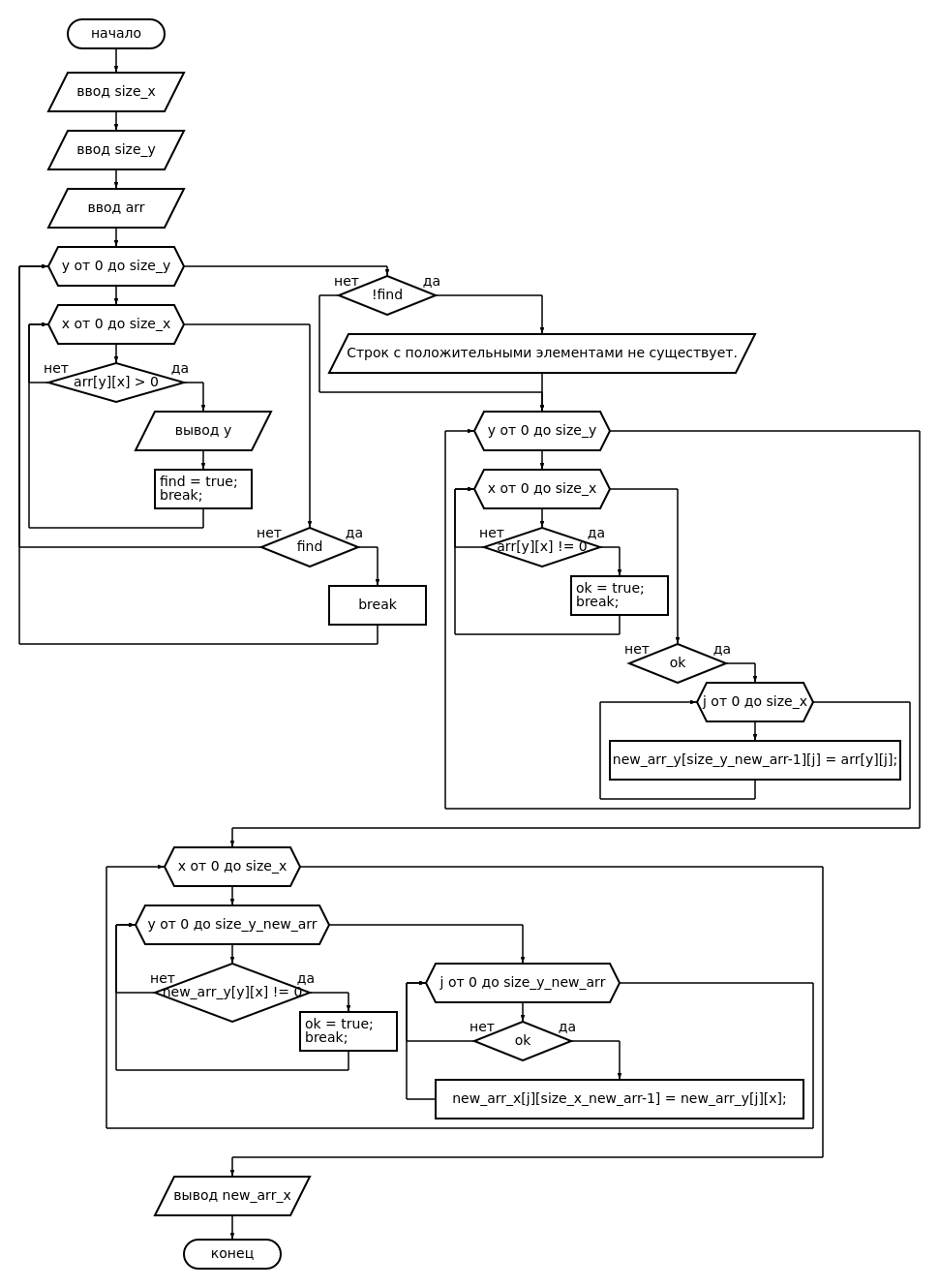
**1.Цель работы:** Целью работы является изучение структуры данных двумерный массив.  
**2.Задачи работы:**  
  
**3.Описание функций:  
1)Имя: main**   
**Назначение:** Вычисление номера первой строки содержащей положительный элемент. Удаление столбцов и строк заполненными нулями.  
**Входные данные:** Высота, ширина и массив.   
**Выходные данные:** Индекс и массив.  
**Побочный эффект:** отсутствует.   
**Тестовые данные:**

|  |  |  |  |
| --- | --- | --- | --- |
| Ширина | Высота | Массив | Ответ |
| 4 | 4 | 1 0 3 4 0 0 0 0 4 0 2 1 5 0 7 8 | Индекс первой строки содержащей положительный элемент: 0 1       3       4  4 2 1  5 7 8 |

**Прототип:** int main()  
**Псевдокод:**  
Ввод ширины и высоты массивами  
Ввод массива  
Нахождение номера строки содержащей хотя бы 1 положительный элемент  
Вывод номера строки  
Удаление из массива строк состоящих из нулей  
Удаление из массива столбцов состоящих из нулей

Вывод очищенного от нулей массива

**Блок-схема:**

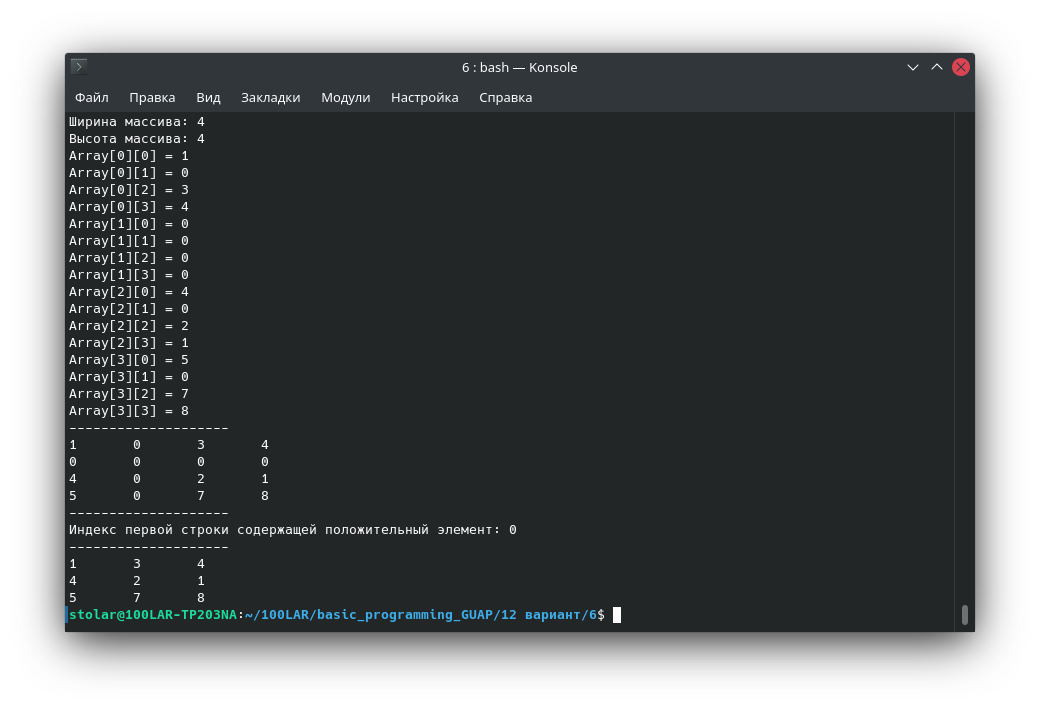


**4. Имя: read\_value**  
**Назначение:** проверка переменной a на корректность ввода  
**Входные данные:** Отсутствуют.   
**Выходные данные:** x(double).  
**Побочный эффект:** отсутствует.   
**Прототип:** double read\_value()

|  |  |
| --- | --- |
| **Псевдокод:** | **Блок-схема:** |
| Бесконечный цикл  ввод строки  проверка на пробелы и удаление их  проверка на запятые и замена их на точки  проверка на минус  проверка на лишние символы  если не было проблем, то останавливаем бесконечный цикл­  конец бесконечного цикла  перевод строки в число  вывод числа |  |

**5.ЛИСТИНГ КОДА:  
/\*   
 Задания на лабораторную работу приводятся в каждом варианте. При написании программ   
 можно использовать как динамические, так и нединамические массивы. Размерность последних   
 задаѐтся именованной константой   
  
 Вариант 12   
   Уплотнить заданную матрицу, удаляя из неѐ строки и столбцы, заполненные нулями.   
   Найти номер первой из строк, содержащих хотя бы один положительный элемент.   
  
\*/   
  
#define RANDOM\_NUMS false   
  
#include <iostream>   
using namespace std;   
  
#include "libs/lib.h"   
#include <cmath>   
#include <time.h>   
  
// проверка ввода   
#include "libs/simple\_char.h"   
#include "libs/input\_validation.h"   
  
// работа с массивами   
#include "libs/array.h"   
  
int read\_size\_arr(const char \*promt = "") {   
 int size;   
 while (true) {   
   size = read\_value(promt, false, false, false);   
   if (size > 0) {   
     break;   
   } else {   
     cout << "Размер должен быть больше 1." << endl;   
   }   
 }   
 return size;   
}   
  
int main() {   
       // смена кодировки   
 system("chcp 65001");   
  
 // очистка терминала   
 clear\_scr();   
  
 // рандом   
 srand(time(NULL));   
  
 int x, y;   
 int size\_x, size\_y;   
  
 // ввод размеров массива   
 while (true) {   
   size\_x = read\_size\_arr("Ширина массива: ");   
   size\_y = read\_size\_arr("Высота массива: ");   
  
   if (size\_x < 1 || size\_y < 0) {   
     cout << "Массив не может содержать 0 элементов." << endl;   
   } else break;   
 }   
  
 // создаём новый массив   
 double \*\*arr = (double\*\*)malloc(size\_y \* sizeof(double\*));   
 for(int i = 0; i < size\_y; i++) {   
     arr[i] = (double\*)malloc(size\_x \* sizeof(double));   
 }   
  
 // ввод значений массива   
 for (y = 0; y < size\_y; y++)   
   for (x = 0; x < size\_x; x++) {   
     if (RANDOM\_NUMS) {   
       arr[y][x] = random\_int(-10, 10);   
     } else {   
       cout << "Array[" << y << "][" << x << "] = ";   
       arr[y][x] = read\_value("", true, true, false);   
     }   
   }   
  
 draw\_line(20);   
 draw\_float\_double\_array(arr, size\_x, size\_y);   
 draw\_line(20);   
  
 // Найти номер первой из строк, содержащих хотя бы один положительный элемент.   
 // строка с положительным элементом   
 bool find = false;   
 for (y = 0; y < size\_y; y++) {   
   for (x = 0; x < size\_x; x++) {   
     if (arr[y][x] > 0) {   
       cout << "Индекс первой строки содержащей положительный элемент: " << y << endl;   
       find = true;   
       break;   
     }   
   }   
   if (find) break;   
 }   
  
 if (!find) {   
   cout << "Строк с положительными элементами не существует." << endl;   
 }   
  
 draw\_line(20);   
  
 // Уплотнить заданную матрицу, удаляя из неѐ строки и столбцы, заполненные нулями.   
 // уплотняем матрицу удаляя строки   
 double \*\*new\_arr\_y = (double\*\*)malloc(sizeof(double\*));   
 int size\_y\_new\_arr = 0;   
 bool ok = false;   
 for (y = 0; y < size\_y; y++) {   
   ok = false;   
   for (x = 0; x < size\_x; x++) {   
     if (arr[y][x] != 0) {   
       ok = true;   
       break;   
     }   
   }   
   if (ok) {   
     new\_arr\_y = (double\*\*) realloc(new\_arr\_y, (++size\_y\_new\_arr) \* sizeof(double\*));   
     new\_arr\_y[size\_y\_new\_arr-1] = (double\*)malloc(size\_x \* sizeof(double));   
  
     for (int j = 0; j < size\_x; j++) {   
       new\_arr\_y[size\_y\_new\_arr-1][j] = arr[y][j];   
     }   
   }   
 }   
    
 // для отладки   
 //draw\_float\_double\_array(new\_arr\_y, size\_x, size\_y\_new\_arr);    
 //draw\_line(20);   
  
 double \*\*new\_arr\_x = (double\*\*)malloc(size\_y\_new\_arr \* sizeof(double\*));   
 int size\_x\_new\_arr = 0;   
 for (x = 0; x < size\_x; x++) {   
   ok = false;   
   for (y = 0; y < size\_y\_new\_arr; y++) {   
     if (new\_arr\_y[y][x] != 0) {   
       ok = true;   
     break;   
     }    
   }   
   if (ok) {   
     size\_x\_new\_arr++;   
     for (int j = 0; j < size\_y\_new\_arr; j++) {   
       new\_arr\_x[j] = (double\*) realloc(new\_arr\_x[j], (size\_x\_new\_arr) \* sizeof(double));   
       new\_arr\_x[j][size\_x\_new\_arr-1] = new\_arr\_y[j][x];   
     }   
   }   
 }   
  
 draw\_float\_double\_array(new\_arr\_x, size\_x\_new\_arr, size\_y\_new\_arr);   
    
 // очистка памяти   
 for(y = 0; y < size\_y; y++) {   
     free(arr[y]);   
 }   
 free(arr);   
  
 for(y = 0; y < size\_y\_new\_arr; y++) {   
     free(new\_arr\_y[y]);   
 }   
 free(new\_arr\_y);   
  
 for(y = 0; y < size\_y\_new\_arr; y++) {   
     free(new\_arr\_x[y]);   
 }   
 free(new\_arr\_x);   
    
       return 0;   
}**

**10.Пример выполнения программы:**



Видно, что результаты расчётов совпадают с тестовыми данными.

**11.Анализ результатов и выводы:**

В ходе этой лабораторной работы я устал перебирать динамические массивы.  
Из плюсов можно отметить её работоспособность.

Из минусов ужасный ввод элементов массива. Пока её протестируешь, уже можно стать дедом.