|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное автономное образовательное учреждение высшего образования  «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ» |

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | Синёв Н. И. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №6 |
| ДВУМЕРНЫЕ МАССИВЫКУРС 1 |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

****РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 1941 |  |  |  | Князюк Р.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2020

1. **Постановка задачи.**

В двумерном целочисленном массиве размера N на K обнулить столбцы, в которых чётных элементов было больше чем нечётных.

1. **Формализация.**

* **Список переменных в данной программе:**
  + Список указателей типа **int**:

***\*\*mass*** – указатель на указатель для выделения памяти под двумерный массив и последующей работы с массивом;

* + Переменные типа **int**:

***cnt\_even*** – переменная, которая хранит количество чётных элементов в столбце;

***k*** – переменная, которая хранит количество столбцов;

***n*** – переменная, которая хранит количество строк;

***i*** – счетчик ,используемый в циклах для работы с двумерным массивом;

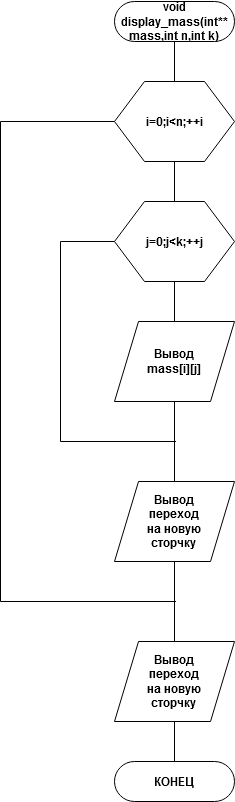
***j*** – счетчик ,используемый в циклах для работы с двумерным массивом;

* Все данные вводятся с клавиатуры.
* Алгоритм вывода и ввода двумерного массива будет отдельной подпрограммой.
* Так как размер массива изначально не определен, то сначала программа получает количество строк и столбцов.
* После получения кол-ва строки столбцов производится выделение памяти (используя функцию *malloc* (memory allocation)) под массив в **heap**. Таким образом, программа может работать с массивами любого размера. Для работы с массивом используется указатель на указатель (***\*\*mass***).
* *Примечание. (Куча - название структуры данных, с помощью которой реализована динамически распределяемая память приложения.)*.
* Ввод массива производится циклически, если кол-во строки и столбцов меньше 10, иначе заполнить массив случайно сгенерированными числами от -5 до 5(использована библиотека *time.h* функция *time,srand,rand*). Массив вводится построчно.
* Так как массив двумерный, то для ввода необходимо два цикла (один вложенный). При вводе массива внешний цикл по строкам, а внутренний по столбцам. (Так как количество итераций известно, будет использован цикл **for**).
* После ввода всех данных, программа выведет массив в удобном формате и начнет его обработку.
* Так как необходимо обнулить столбцы , в которых четных элементов больше чем нечетных, то внешний цикл будет по столбцам, а внутренний по строкам.
* Внутри вложенного цикла проверяется четность элемента. Если элемент четный , то счетчик ***cnt\_even*** увеличить на 1.
* Во внешнем цикле производится проверка cnt\_even. Если cnt\_even (количество четных элементов) больше n/2 (половина количества строк), то производится обнуление столбца.
* В конце программа должны вывести обработанный массив
* Операционная система: Windows 7 Максимальная
* Написание кода: Visual Studio Code
* Компилятор: MinGW GCC-8.2.0-5
* Запуск программы через \*.exe

**Таблица 1 – Тестовые примеры**

|  |  |  |
| --- | --- | --- |
| **Пример** | **№1** | **№2** |
| **Данные** | **Массив 4(строки) на 5(столбцов)**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **1** | **1** | **1** | **1** | **1** | | **2** | **3** | **4** | **5** | **6** | | **6** | **7** | **8** | **9** | **10** | | **4** | **5** | **6** | **7** | **9** | | **Массив 3 на 3**   |  |  |  | | --- | --- | --- | | **13** | **22** | **54** | | **46** | **33** | **24** | | **2** | **11** | **5** | |
| **Результат** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **0** | **1** | **0** | **1** | **1** | | **0** | **3** | **0** | **5** | **6** | | **0** | **7** | **0** | **9** | **10** | | **0** | **5** | **0** | **7** | **9** | | |  |  |  | | --- | --- | --- | | **0** | **22** | **0** | | **0** | **33** | **0** | | **0** | **11** | **0** | |

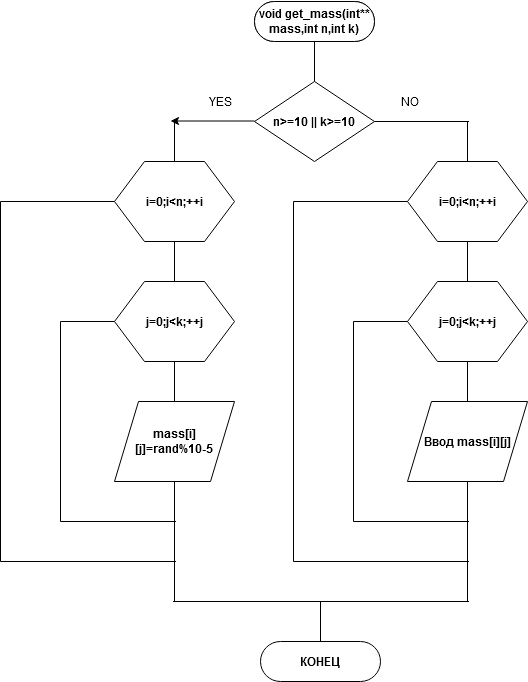
1. **Блок схемы программы.**
   1. **Подпрограмма вывода массива.**



**Рисунок 1 – подпрограммы вывода**

**display\_mass(int\*\* mass,int n,int k)**

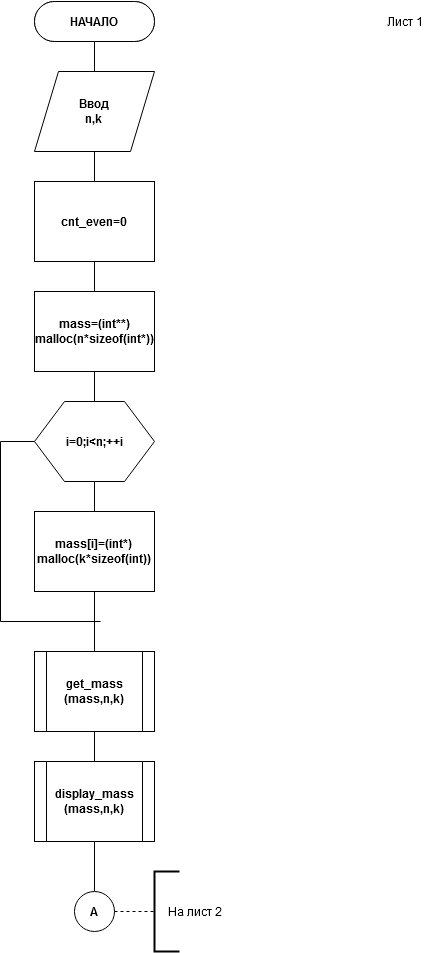
* 1. **Подпрограмма ввода массива.**

****

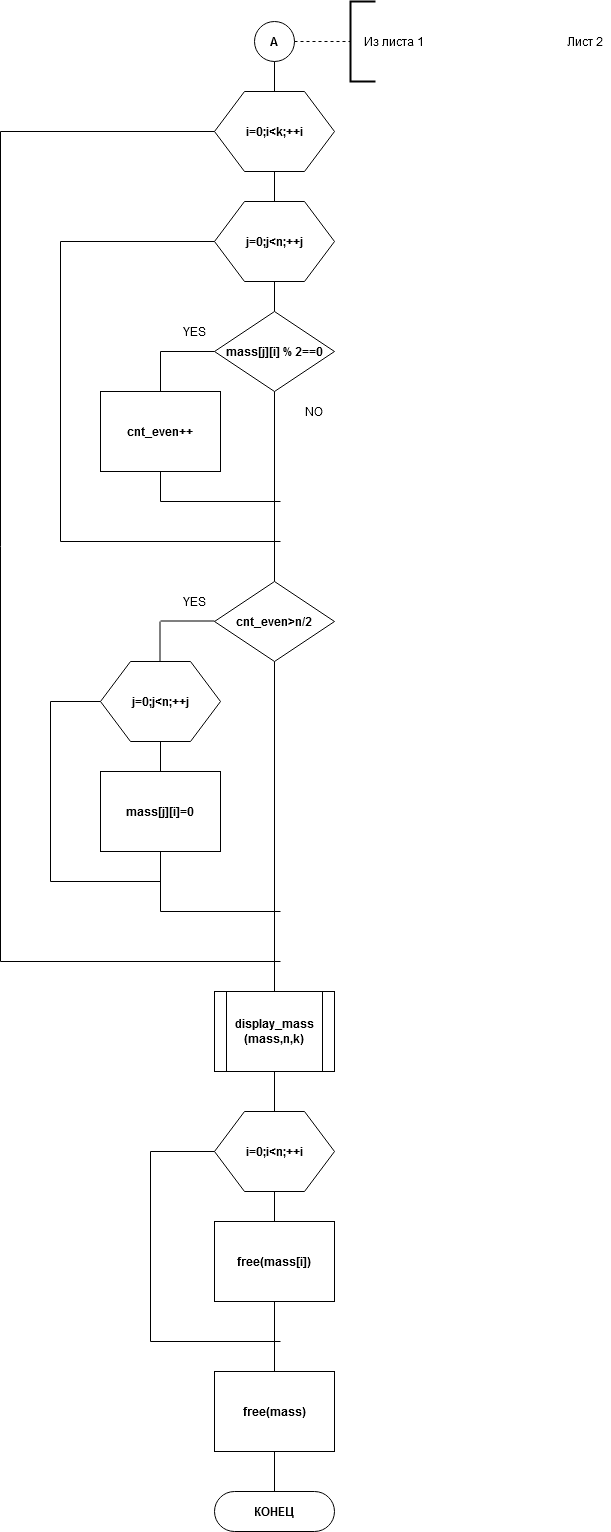
**Рисунок 2 – подпрограммы ввода**

**get\_mass(int\*\* mass,int n,int k)**

* 1. **Основная программа.**

****

**Рисунок 3 – Блок-схема основной программы**

****

**Рисунок 3**

1. **Код программы на Си.**

#include <stdio.h>

#include <stdlib.h> //для system() и malloc()

#include <time.h>   //для функции time(0) (time нужно для генерации случайных чисел)

//------------------------------------------------------------------------

//------------------------------------------------------------------------

void display\_mass(int\*\* mass,int n,int k){

    int i,j;

    for(i=0;i<n;++i){

        for(j=0;j<k;++j)

            printf("mass[%d,%d]=%2d ",i,j,mass[i][j]);

        printf("\n\n");

    }

    printf("\n");

}

//------------------------------------------------------------------------

//------------------------------------------------------------------------

void get\_mass(int\*\* mass,int n,int k){

    int i,j;

    if(n>=10 || k>=10){

        printf("Array is too big.\nWill be used random generation...\n");

        //инициализация генератора случайных чисел

        srand(time(0));

        //заполнение массива значениями от -5 до 5

        for(i=0;i<n;++i)

            for(j=0;j<k;++j){

                mass[i][j]=rand() % 10 - 5;

            }

    }

    else{

        for(i=0;i<n;++i)

            for(j=0;j<k;++j){

                printf("mass[%d,%d] = ",i,j);

                scanf("%d",&mass[i][j]);

            }

    }

}

//------------------------------------------------------------------------

//------------------------------------------------------------------------

int main(void){

    //объявление переменных

    int \*\*mass;

    int k,n,i,j,cnt\_even=0;

    //вводим данные о кол-ве рядов и столбцов

    printf("Enter nums of rows: ");

    scanf("%d",&n);

    printf("Enter nums of columns: ");

    scanf("%d",&k);

    //выделяем память под массив в heap

    //будем хранить строками

    mass=(int\*\*)malloc(n\*sizeof(int\*));

    for(i=0;i<n;++i){

        mass[i]=(int\*)malloc(k\*sizeof(int));

    }

    printf("Heap was created...\nPress any key...");

    \_flushall(); //очищаем поток ввода

    getchar();

    printf("Start working...\n");

    //ввод массива

    get\_mass(mass,n,k);

    //вывод массива

    display\_mass(mass,n,k);

    printf("Processing...\n");

    //обработка массива

    for(i=0;i<k;++i){

        for(j=0;j<n;++j){

            if(mass[j][i] % 2 == 0)

                cnt\_even+=1;

        }

        if(cnt\_even>n/2)

            for(j=0;j<n;++j)

                mass[j][i]=0;

        cnt\_even=0;

    }

    printf("Array after processing...\n");

    //вывод массива

    display\_mass(mass,n,k);

    //очищаем память

    for(i=0;i<n;++i)

        free(mass[i]);

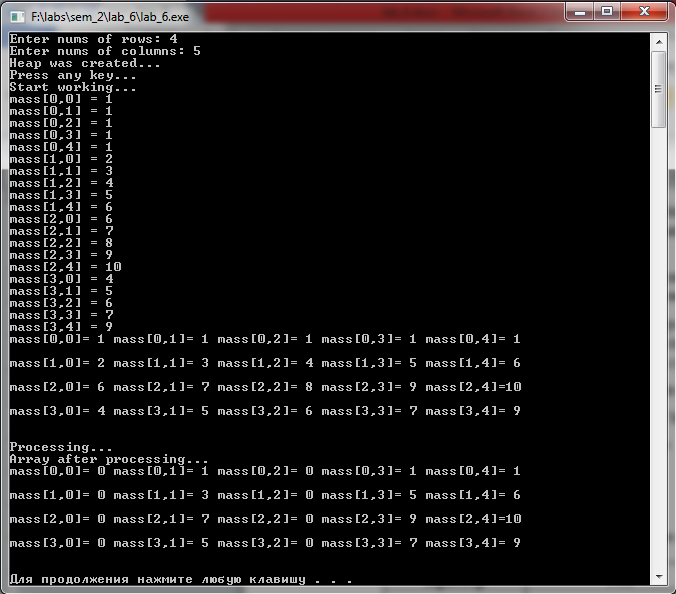
    free(mass);

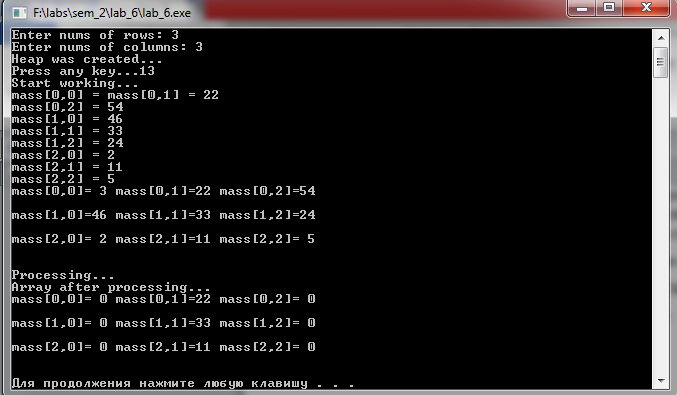
    system("pause");

    return 0;

}

1. **Проверка работы программы на тестовых примерах.**

****

****

**Рисунок 5 – Тестовый пример 2**

**Рисунок 4 – Тестовый пример 1**

1. **Вывод.**

Тестовые примеры (5 пункт отчета) доказывают правильность работы алгоритма и программы. Таким образом, программа способна в двумерном целочисленном массиве размера (произвольного) N на K обнулить столбцы, в которых чётных элементов было больше чем нечётных, а также вывести массив до и после обработки.