**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное образовательное учреждение высшего профессионального образования**

**Национальный исследовательский университет**

**«Высшая школа экономики»**

**Московский институт электроники и математики Национального исследовательского университета**

**«Высшая школа экономики»**

**Кафедра информационных технологий**

**и автоматизированных систем**

**Курс: Информатика и программирование**

**ОТЧЕТ**

**по лабораторной работе № 2\_**

**Студент:\_Лёвин О.В.\_ \_\_\_\_\_\_\_\_**

**Группа:\_ИВТ-13\_\_\_\_\_**

**Вариант:\_92**

**Руководитель:\_Ерохина Е.А \_\_**

**Оценка:\_\_\_\_\_\_\_\_\_\_**

|  |  |  |
| --- | --- | --- |
| **Раздел** | **Маx** | Рез. |
|  | **оценка** | **оценка** |
|  |  |  |
| Оформление программы | 1 |  |
|  |  |  |
| Тесты | 1 |  |
|  |  |  |
| Алгоритм | 3 |  |
|  |  |  |
| Вопросы | 2 |  |
|  |  |  |
| Доп. задание | 3 |  |
|  |  |  |

Оглавлени

[Задание 3](#_Toc409646054)

[Листинг программы (C) 4](#_Toc409646055)

[Листинг программы (C++) 5](#_Toc409646056)

[Распечатка тестов к программе и результатов 7](#_Toc409646057)

Задание

Дан двумерный массив A размерностью m\*n. Заменить местами первый и последний положительные элементы.

Листинг программы (C)

/\*

\* lab2.c

\* 28.01.15

\*/

#include <stdio.h>

#include <stdlib.h>

#include <errno.h>

#include <time.h>

#define STDIN\_INPUT

enum {

CHANGED,

ONE\_POSITIVE,

NO\_POSITIVE

};

/\*

\* A - исходная матрица

\* m и n - количество строк и столбцов соответственно

\*/

int MatrixChange (int\*\* A, const size\_t m, const size\_t n);

int main(void)

{

size\_t m, n, i, j;

printf ("Введите число строк и число столбцов матрицы: ");

scanf ("%ld %ld", &m, &n);

//выделение памяти под матрицу

int\*\* A = (int\*\*) malloc (sizeof(int\*) \* m);

if (A == NULL) {

perror (NULL);

return errno;

}

for (i = 0; i < m; i++)

if ((A[i] = (int\*) malloc (sizeof(int) \* n)) == NULL) {

perror (NULL);

return errno;

}

#ifdef STDIN\_INPUT

printf ("Введите элементы матрицы A: ");

for (i = 0; i < m; i++)

for (j = 0; j < n; j++)

scanf ("%ld", &A[i][j]);

#else

srand (time(NULL));

printf ("Элементы матрицы: \n");

for (i = 0; i < m; i++) {

for (j = 0; j < n; j++) {

A[i][j] = - rand() / 10000;

printf ("%d\t", A[i][j]);

}

puts("");

}

#endif

int res = MatrixChange (A, m, n);

//анализ результата замены

switch (res) {

case CHANGED:

printf ("\nПерестановка удачна. Полученная матрица:\n");

for (i = 0; i < m; i++) {

for (j = 0; j < n; j++)

printf ("%d\t", A[i][j]);

puts("");

}

break;

case ONE\_POSITIVE:

printf ("\nМатрица содержит только один положительнй элемент. Изменений нет.\n");

break;

case NO\_POSITIVE:

printf ("\nПоложительные элементы отсутствуют. Изменений нет.\n");

break;

}

//освобождение динамической памяти

for (i = 0; i < m; i++)

free (A[i]);

free (A);

return 0;

}

int MatrixChange (int\*\* A, const size\_t m, const size\_t n)

{

//координаты первого и последнего положительных

struct {

int row, col;

} firstPositive = {-1, -1},

lastPositive = {-1, -1};

//нахождение первого и последнего положительных

size\_t i, j;

for (i = 0; i < m; i++)

for (j = 0; j < n; j++)

if (A[i][j] > 0) {

if (firstPositive.row == -1)

firstPositive.row = i, firstPositive.col = j;

else

lastPositive.row = i, lastPositive.col = j;

}

//возвращаем результат

if (firstPositive.row != -1 && lastPositive.row != -1) {

//перестановка их местами

int c = A[firstPositive.row][firstPositive.col];

A[firstPositive.row][firstPositive.col] = A[lastPositive.row][lastPositive.col];

A[lastPositive.row][lastPositive.col] = c;

return CHANGED;

}

else if (firstPositive.row != -1 && lastPositive.col == -1)

return ONE\_POSITIVE;

else

return NO\_POSITIVE;

}

Листинг программы (C++)

/\*

\* lab2.cxx

\* 28.01.15

\*/

#include <cstdio>

#include <iostream>

#include <cstdlib>

#include <cerrno>

#include <ctime>

using std::cin;

using std::cout;

using std::cerr;

using std::endl;

#define STDIN\_INPUT

enum {

CHANGED,

ONE\_POSITIVE,

NO\_POSITIVE

};

/\*

\* A - исходная матрица

\* m и n - количество строк и столбцов соответственно

\*/

int MatrixChange (int\*\* A, const size\_t m, const size\_t n);

int main(void)

{

size\_t m, n;

printf ("Введите число строк и число столбцов матрицы: ");

cin >> m >> n;

//выделение памяти под матрицу

int\*\* A;

try {

A = new int\*[m];

} catch (std::bad\_alloc& e){

cerr << e.*what*() << endl;

return errno;

}

for (int\*\* i = A; i != A+m; i++)

try {

\*i = new int[n];

} catch (std::bad\_alloc& e) {

cerr << e.*what*() << endl;

}

#ifdef STDIN\_INPUT

cout << "Введите элементы матрицы A: ";

for (size\_t i = 0; i < m; i++)

for (int\* j = A[i]; j != A[i]+n; j++)

cin >> \*j;

#else

srand (time(NULL));

cout << "Элементы матрицы: \n";

for (int\*\* i = A; i != A+m; i++) {

for (int\* j = \*i; j != \*i+n; j++) {

\*j = rand() / 10000;

cout << \*j << "\t";

}

cout << endl;

}

#endif

int res = MatrixChange (A, m, n);

//анализ результата замены

switch (res) {

case CHANGED:

cout << "\nПерестановка удачна. Полученная матрица:\n";

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

for (const int\* j = A[i]; j != A[i]+n; j++)

cout << \*j << "\t";

cout << endl;

}

break;

case ONE\_POSITIVE:

cout << "\nМатрица содержит только один положительнй элемент. Изменений нет.\n";

break;

case NO\_POSITIVE:

cout << "\nПоложительные элементы отсутствуют. Изменений нет.\n";

break;

}

//освобождение динамической памяти

for (size\_t i = 0; i < m; i++)

delete[] A[i];

delete[] A;

return 0;

}

int MatrixChange (int\*\* A, const size\_t m, const size\_t n)

{

//координаты первого и последнего положительных

struct {

int row, col;

} firstPositive = {-1, -1},

lastPositive = {-1, -1};

//нахождение первого и последнего положительных

for (size\_t i = 0; i < m; i++)

for (size\_t j = 0; j < n; j++)

if (A[i][j] > 0) {

if (firstPositive.row == -1)

firstPositive.row = i, firstPositive.col = j;

else

lastPositive.row = i, lastPositive.col = j;

}

//возвращаем результат

if (firstPositive.row != -1 && lastPositive.row != -1) {

//перестановка их местами

int c = \*(\*(A + firstPositive.row) + firstPositive.col);

\*(\*(A + firstPositive.row) + firstPositive.col) = \*(\*(A + lastPositive.row) + lastPositive.col);

\*(\*(A + lastPositive.row) + lastPositive.col) = c;

return CHANGED;

}

else if (firstPositive.row != -1 && lastPositive.col == -1)

return ONE\_POSITIVE;

else

return NO\_POSITIVE;

}

Распечатка тестов к программе и результатов

|  |  |  |
| --- | --- | --- |
| № | Исходные данные | Результат |
| 1 | -1 -1 | Cannot allocate memory |
| 2 | N = 2; M = 2  A[][] = {{1,2}, {3,4}} | Перестановка удачна. Полученная матрица:  A[][] = {{4,2}, {3, 1}} |
| 3 | N = 2; M = 2  A[][] = {{1,-2}, {3,-4}} | Перестановка удачна. Полученная матрица:  A[][] {{3,-2},{1,-4}} |
| 4 | N = 2; M = 2  A[][] = {{1,-2},{-3,-4}} | Матрица содержит только один положительный элемент. Изменений нет. |
| 5 | N = 2; M = 2  A[][] = {{-1,-2},{-3,-4}} | Положительные элементы отсутствуют. Изменений нет. |