**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**"Национальный исследовательский университет**

**"Высшая школа экономики"**

Московский институт электроники и математики им. А.Н. Тихонова НИУ ВШЭ

Департамент компьютерной инженерии

**Курс: Алгоритмизация и программирование**

|  |  |  |  |
| --- | --- | --- | --- |
| **Раздел** | **Max оценка** | **Итог. Оценка 1** | **Итог. Оценка 2** |
| Работа программы | 1 |  |  |
| Тесты | 1 |  |  |
| Правильность алгоритма | 3 |  |  |
| Ответы на вопросы | 2 |  |  |
| Дополнительное задание | 3 |  |  |

**ОТЧЕТ**

**по лабораторной работе № 11**

**Студент:\_ Иванилов Леонид Михайлович**

**Группа:\_БИВ231**

**Вариант:\_143 (1,16)**

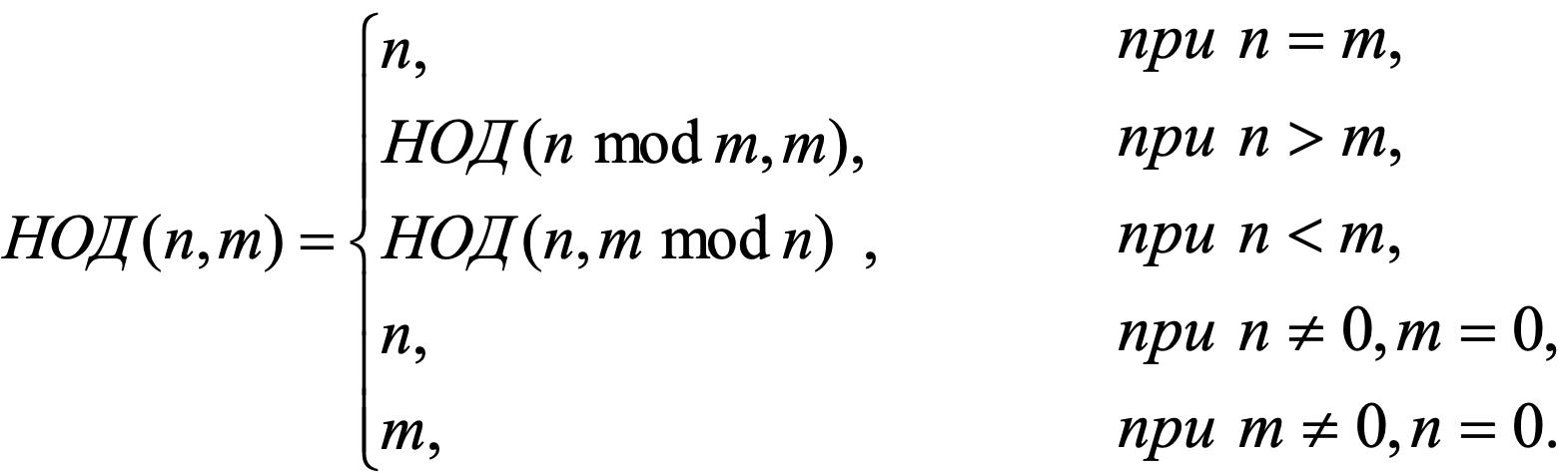
**Руководитель:\_Ерохина Елена Альфредовна**

**Оценка:\_\_\_\_\_\_\_\_\_\_**

**Дата сдачи\_\_\_\_\_\_\_\_\_\_**

**Задания:**

**I.**



**II.** Формирует новое бинарное дерево, состоящее из отрицательных чисел исходного дерева.

**МОСКВА 2024**

**Листинг №1**

#include <iostream>  
#include <limits>  
#include <ctime>  
using namespace std;  
  
// Рекурсивная функция для нахождения НОД по второму алгоритму Евклида  
int gcd\_recursion(int n, int m) {  
 if (n == 0) return m;  
 if (m == 0) return n;  
 if (n == m) return n;  
 if (n > m) return gcd\_recursion(n % m, m);  
 return gcd\_recursion(n, m % n);  
}  
  
// Итеративная функция для нахождения НОД по второму алгоритму Евклида  
int gcd\_iteration(int n, int m) {  
 while (n != 0 && m != 0) {  
 if (n > m) {  
 n = n % m;  
 } else {  
 m = m % n;  
 }  
 }  
 return n + m; // Одно из чисел станет 0, возвращаем сумму, так как другое число - НОД  
}  
  
// Функция для ввода целого и неотрицательного числа  
int input\_positive\_integer(const string& prompt) {  
 int number;  
 while (true) {  
 cout << prompt;  
 cin >> number;  
  
 // Проверка, что ввод корректный и число является целым и неотрицательным  
 if (cin.fail() || number < 0 || cin.peek() != '\n') {  
 cout << "Please enter a non-negative integer." << endl;  
 cin.clear();  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 } else {  
 return number;  
 }  
 }  
}  
  
int main() {  
 while (true) {  
 int n = input\_positive\_integer("Input the first number: ");  
 int m = input\_positive\_integer("Input the second number: ");  
  
 long long time1 = clock();  
 cout << "GCD (Recursion): " << gcd\_recursion(n, m) << endl;  
 long long time2 = clock();  
  
 cout << "GCD (Iteration): " << gcd\_iteration(n, m) << endl;  
 long long time3 = clock();  
  
 cout << "Recursion time: " << (time2 - time1) \* 1000 / CLOCKS\_PER\_SEC << " ms" << endl;  
 cout << "Iteration time: " << (time3 - time2) \* 1000 / CLOCKS\_PER\_SEC << " ms" << endl;  
 }  
}

**Листинг №2**

#include <iostream>  
#include <iomanip>  
#include <limits>  
#define L 20  
  
  
using namespace std;  
  
void input(int A[][L], int \*n, int \*m) {  
 cout << "Матрица A: " << endl;  
 do {  
 cout << "Введите n и m по заданным условиям 0 < n, m <= " << L << ": ";  
 cin >> \*n >> \*m;  
 if (\*n <= 0 || \*n > L || \*m <= 0 || \*m > L)  
 cout << "заново" << endl;  
 } while (\*n <= 0 || \*n > L || \*m <= 0 || \*m > L);  
 int \*ui, \*uj;  
 cout << "Введите элементы матрицы А: " << endl;  
 for (ui = \*A; ui < \*A + (\*n \* L); ui += L)  
 for (uj = ui; uj < ui + \*m; ++uj)  
 cin >> \*uj;  
}  
  
  
void swap(int A[][L], int n, int m, int \*flag\_pos, int \*flag\_neg) {  
  
 int \*ui, \*uj, \*min\_pos, \*max\_neg, c;  
  
 min\_pos = NULL;  
 max\_neg = NULL;  
  
 for (ui = \*A; ui < \*A + (n \* L); ui += L) {  
 for (uj = ui; uj < ui + m; uj++) {  
 if (\*uj > 0) {  
 \*flag\_pos = 1;  
 if (min\_pos == NULL || \*min\_pos > \*uj) {  
 min\_pos = uj;  
 }  
 } else if (\*uj < 0) {  
 \*flag\_neg = 1;  
 if (max\_neg == NULL || \*max\_neg < \*uj) {  
 max\_neg = uj;  
 }  
 }  
 }  
 }  
  
 if (\*flag\_neg && \*flag\_pos) {  
 c = \*max\_neg;  
 \*max\_neg = \*min\_pos;  
 \*min\_pos = c;  
 }  
}  
  
void output(int A[][L], int n, int m) {  
 int \*ui, \*uj;  
 for (ui = \*A; ui < \*A + (n \* L); ui += L)  
 for (uj = ui; uj < ui + m; ++uj)  
 cout << setw(3) << \*uj;  
 cout << endl;  
}  
  
int main() {  
 int A[L][L], n, m;  
 int flag\_pos = 0;  
 int flag\_neg = 0;  
  
 input(A, &n, &m);  
 cout << "Исходная матрица " << endl;  
 output(A, n, m);  
 swap(A, n, m, &flag\_pos, &flag\_neg);  
  
 if(!flag\_pos && !flag\_neg){  
 cout << "Матрица состоит только из нулей, изменений нет" << endl;  
 }  
 else if (!flag\_pos){  
 cout << "В матрице нет положительных чисел" << endl;  
 }  
 else if(!flag\_neg){  
 cout << "В матрице нет отрицательных чисел" << endl;  
 }  
 else{  
 cout << "Изменённая матрица" << endl;  
 output(A, n, m);  
 }  
  
  
 return 0;  
}

**Тесты №1**

|  |  |
| --- | --- |
| ВВОД | ВЫВОД |
| Массив A:  Введите n по заданным условиям 0 < n <= 20: 5  Введите элементы массива А:  1 2 3 4 5  Массив B:  Введите m по заданным условиям 0 < m <= 20: 10  Введите элементы массива В:  1 1 2 2 3 3 4 4 6 6 | Массив A:  1 2 3 4 5  Массив B:  1 1 2 2 3 3 4 4 6 6  Массив C:  1 2 3 4 |
| Задание 1  Массив A:  Введите n по заданным условиям 0 < n <= 20: 5  Введите элементы массива А:  1 2 3 4 5  Массив B:  Введите m по заданным условиям 0 < m <= 20: 5  Введите элементы массива В:  1 2 3 4 5 | Массив A:  1 2 3 4 5  Массив B:  1 2 3 4 5  Массив C пуст. |
| Задание 1  Массив A:  Введите n по заданным условиям 0 < n <= 20: 5  Введите элементы массива А:  1 1 2 3 4  Массив B:  Введите m по заданным условиям 0 < m <= 20: 2  Введите элементы массива В:  1 1 | Массив A:  1 1 2 3 4  Массив B:  1 1  Массив C:  1 1 |

Тесты №2

|  |  |
| --- | --- |
| ВВОД | ВЫВОД |
| Матрица A:  Введите n и m по заданным условиям 0 < n, m <= 20:  3  3  Введите элементы матрицы А:  0 0 0  0 0 0  0 0 0 | Исходная матрица  0 0 0 0 0 0 0 0 0  Матрица состоит только из нулей, изменений нет |
| Матрица A:  Введите n и m по заданным условиям 0 < n, m <= 20: 3 3  Введите элементы матрицы А:  1 1 1  1 1 1  1 1 1 | Исходная матрица  1 1 1 1 1 1 1 1 1  В матрице нет отрицательных чисел |
| Матрица A:  Введите n и m по заданным условиям 0 < n, m <= 20: 3 3  Введите элементы матрицы А:  -1 -1 -1  -1 -1 -2  -3 -4 -5 | Исходная матрица  -1 -1 -1 -1 -1 -2 -3 -4 -5  В матрице нет положительных чисел |
| Матрица A:  Введите n и m по заданным условиям 0 < n, m <= 20: 3 3  Введите элементы матрицы А:  1 0 0  0 0 0  0 0 -1 | Исходная матрица  1 0 0 0 0 0 0 0 -1  Изменённая матрица  -1 0 0 0 0 0 0 0 1 |

Листинг доп задания

#include <iostream>  
#include <limits>  
#define lmax 20  
  
using namespace std;  
  
int input\_value();  
int input();  
void input\_matrix(int Matrix[][lmax], int \*strings, int \*elements);  
void output\_matrix(int Matrix[][lmax], int strings, int elements);  
void swap(int Matrix[][lmax], int strings, int elements, int \*\*chet, int \*\*max);  
  
int main()  
{  
 int Matrix[lmax][lmax];  
 int strings, elements;  
 input\_matrix(Matrix, &strings, &elements);  
  
 cout << "Введённая матрица:" << endl;  
 output\_matrix(Matrix, strings, elements);  
  
 int \*chet = nullptr, \*max = nullptr;  
 swap(Matrix, strings, elements, &chet, &max);  
 if (chet == nullptr && max == nullptr){  
 cout << "Изменений не было, так как не было найдено оба элемента" << endl;  
 }  
 else if (chet == nullptr) {  
 cout << "Изменений не было, так как максимальный нечетный элемент отсутствует в чётных столбцах" << endl;  
 } else {  
 if (max == nullptr) {  
 cout << "Изменений не было, так как 1-й четный элемент отсутствует в нечётных столбцах" << endl;  
 } else {  
 if (\*chet == \*max) {  
 cout << "Изменений не было, так как 1-й чётный в нечётных столбцах и максимальный нечётный в чётных столбцах" << endl;  
 } else {  
 cout << "Изменённая матрица:" << endl;  
 output\_matrix(Matrix, strings, elements);  
 }  
 }  
 }  
  
 return 0;  
}  
  
int input\_value() {  
 float value;  
 do {  
 int fail = 1;  
 while (fail == 1) {  
 cin >> value;  
 if (cin.fail()) {  
 cin.clear();  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 cout << "Некорректные данные, повторите ввод: ";  
 } else {  
 fail = 0;  
 }  
 }  
 if (value != int(value) || value > INT\_MAX || value < INT\_MIN)  
 cout << "Некорректные данные, повторите ввод: ";  
 } while (value != int(value) || value > INT\_MAX || value < INT\_MIN);  
 return value;  
}  
  
int input() {  
 float value;  
 do {  
 int fail = 1;  
 while (fail == 1) {  
 cin >> value;  
 if (cin.fail()) {  
 cin.clear();  
 cin.ignore(numeric\_limits<streamsize>::max(), '\n');  
 cout << "Некорректные данные, повторите ввод: ";  
 } else {  
 fail = 0;  
 }  
 }  
 if (value != int(value) || value > lmax || value <= 0)  
 cout << "Некорректные данные, повторите ввод: ";  
 } while (value != int(value) || value > lmax || value <= 0);  
 return value;  
}  
  
void input\_matrix(int Matrix[][lmax], int \*strings, int \*elements)  
{  
 cout << "Введите количество строк матрицы: ";  
 \*strings = input();  
  
 cout << "Введите количество столбцов матрицы: ";  
 \*elements = input();  
  
 cout << "Введите элементы матрицы " << \*strings << " на " << \*elements << " элементов:" << endl;  
  
 int \*strings\_num, \*elements\_num;  
  
 for (strings\_num = \*Matrix; strings\_num < \*Matrix + (lmax \* \*strings); strings\_num += lmax)  
 {  
 for (elements\_num = strings\_num; elements\_num < strings\_num + \*elements; elements\_num++)  
 {  
 \*elements\_num = input\_value();  
 }  
 }  
}  
  
void swap(int Matrix[][lmax], int strings, int elements, int \*\*chet, int \*\*max)  
{  
 int \*strings\_num, \*elements\_num;  
 int count = 0;  
 for (elements\_num = \*Matrix; elements\_num < \*Matrix + elements; elements\_num++)  
 {  
 count++;  
 if (count % 2 != 0)  
 {  
 for (strings\_num = elements\_num; strings\_num < \*Matrix + (lmax \* strings); strings\_num += lmax)  
 {  
 if (\*strings\_num % 2 != 0)  
 {  
 if (\*max == nullptr)  
 {  
 \*max = strings\_num;  
 }  
 else  
 {  
 if (\*strings\_num > \*\*max)  
 {  
 \*max = strings\_num;  
 }  
 }  
 }  
 }  
 }  
 else  
 {  
 for (strings\_num = elements\_num; strings\_num < \*Matrix + (lmax \* strings) && \*chet == nullptr; strings\_num += lmax)  
 {  
 if (\*strings\_num % 2 == 0)  
 {  
 \*chet = strings\_num;  
 }  
 }  
 }  
 }  
  
 if (\*chet != nullptr && \*max != nullptr)  
 {  
 int value = \*\*max;  
 \*\*max = \*\*chet;  
 \*\*chet = value;  
 }  
}  
  
void output\_matrix(int Matrix[][lmax], int strings, int elements)  
{  
 int \*strings\_num, \*elements\_num;  
 for (strings\_num = \*Matrix; strings\_num < \*Matrix + (lmax \* strings); strings\_num += lmax)  
 {  
 for (elements\_num = strings\_num; elements\_num < strings\_num + elements; elements\_num++)  
 {  
 cout << \*elements\_num << " ";  
 }  
 cout << endl;  
 }  
}

Тесты для доп задания

|  |  |
| --- | --- |
| ВВОД | ВЫВОД |
| Введите количество строк матрицы: 2  Введите количество столбцов матрицы: 2  Введите элементы матрицы 2 на 2 элементов:  1 1  1 1 | Изменений не было, так как максимальный нечетный элемент отсутствует в чётных столбцах |
| Введите количество строк матрицы: 2  Введите количество столбцов матрицы: 2  Введите элементы матрицы 2 на 2 элементов:  0 0  0 0 | Изменений не было, так как 1-й четный элемент отсутствует в нечётных столбцах |
| Введите количество строк матрицы: 2  Введите количество столбцов матрицы: 2  Введите элементы матрицы 2 на 2 элементов:  0 1  0 1 | Изменений не было, так как не было найдено оба элемента |
| Введите количество строк матрицы: 4  Введите количество столбцов матрицы: 4  Введите элементы матрицы 4 на 4 элементов:  1 2 3 4  5 6 7 8  9 10 11 12  13 14 15 16 | Изменённая матрица:  1 15 3 4  5 6 7 8  9 10 11 12  13 14 2 16 |
|  |  |
|  |  |