**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**Выполнила**: Асташкина Мария Николаевна   
**Группа**: БПИ 196  
**Вариант: 2**

**ЗАДАНИЕ**

Найти определитель матрицы А. Входные данные: целое положительное число *n*, произвольная матрица А размерности *n* х *n*. Количество потоков является входным параметром, при этом размерность матриц может быть не кратна количеству потоков.

**ИСХОДНЫЙ КОД**

#include <iostream>

#include <omp.h>

#include <locale.h>

using namespace std;

void GetMatr(int\*\* mas, int\*\* p, int i, int j, int m) {

int ki, kj, di, dj;

di = 0;

#pragma omp for private(ki)

for (ki = 0; ki < m - 1; ki++) { // проверка индекса строки

if (ki == i) di = 1;

dj = 0;

for (kj = 0; kj < m - 1; kj++) { // проверка индекса столбца

if (kj == j) dj = 1;

p[ki][kj] = mas[ki + di][kj + dj];

}

}

}

// Рекурсивное вычисление определителя

int Determinant(int\*\* mas, int m) {

int i, j, d, k, n;

int\*\* p;

p = new int\* [m];

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

p[i] = new int[m];

j = 0; d = 0;

k = 1; //(-1) в степени i

n = m - 1;

if (m < 1) cout << "Определитель вычислить невозможно!";

if (m == 1) {

d = mas[0][0];

return(d);

}

if (m == 2) {

d = mas[0][0] \* mas[1][1] - (mas[1][0] \* mas[0][1]);

return(d);

}

if (m > 2) {

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

GetMatr(mas, p, i, 0, m);

d = d + k \* mas[i][0] \* Determinant(p, n);

k = -k;

}

}

return(d);

}

/// функция для проверки является ли чар цифрой

bool is\_digit(const char\* S) {

const char\* temp = "0123456789-+";

unsigned point\_count = 0;

for (int i = 0; i < strlen(S); i++) {

if ((i > 0) && (S[i] == '+' || S[i] == '-')) return false;

if (S[i] == '.') {

point\_count++;

if (point\_count > 1) return false;

}

if (!strchr(temp, S[i])) return false;

}

return true;

}

/// функция для проверки введенных данных на корректность

int CheckInput() {

const int N = 256;

char S[N];

// проверяем на то что введелнное число - не строка и записываем его в переменнуб n

cin.getline(S, N);

while (!is\_digit(S)) {

cerr << "Неправильный ввод, попробуйте еще раз!\n";

cin.getline(S, N);

}

return atof(S);

}

int main()

{

setlocale(LC\_ALL, "Russian");

int n, k;

cout << "Введите размерность матрицы: ";

n = CheckInput();

cout << "\nВведите количестов потоков: ";

k = CheckInput();

int\*\* A = new int\* [n];

for (int i = 0; i < n; i++)

{

A[i] = new int[n];

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

{

A[i][j] = CheckInput();

}

}

omp\_set\_num\_threads(k);

#pragma omp parallel

cout << "Матрица, которую вы ввели: " << endl;

for (int i = 0; i < n; i++) {

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

cout << A[i][j] << " ";

cout << endl;

}

cout << "Детерминант введенной матрицы = " << Determinant(A, n);

return 0;

}

**ТЕСТЫ**

1. Изображение выглядит как текст

   Автоматически созданное описание
2. Изображение выглядит как текст

   Автоматически созданное описание проверка на корректность входных данных
3. Изображение выглядит как текст

   Автоматически созданное описание проверка на ввод отрицательных данных
4. Изображение выглядит как текст

   Автоматически созданное описание проверка на ввод данных матрицы (надо вводить по одному числу, а для удобства уже после ввода программа выдаст матрицу в правильном виде)