**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра «АПУ»**

**ОТЧЁТ**

**по лабораторной работе № 4**

**по дисциплине «Программирование»**

**«ДВУМЕРНЫЕ И ДИНАМИЧЕСКИЕ МАССИВЫ»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 4391 | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | Мухачев Д. О. |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Писарев А.С. |

Санкт-Петербург

2024

**Цель работы:**

Задачей было ввести матрицу определенного размера и выполнить некоторые операции с ней в интерактивном режиме.

**Ход работы:**

Первым шагом была реализация функции перемножения матриц:

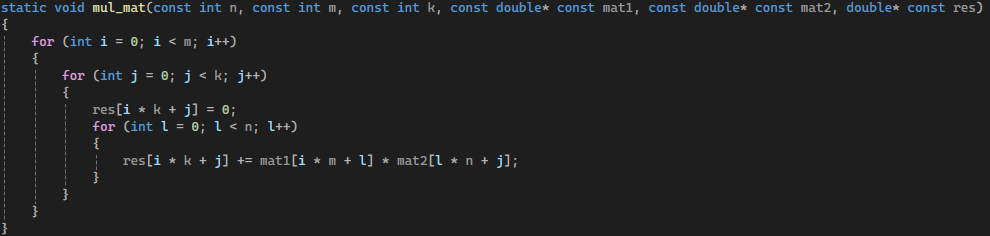
****

Рисунок 1

Вторым шагом – возведение квадратной матрицы в степень, с консервативным использованием памяти:

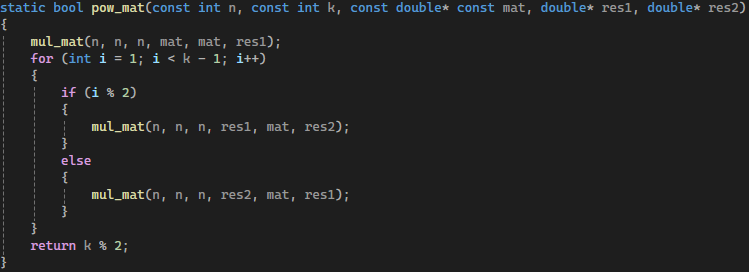


Рисунок 2

Также была написана функция вывода матрицы:

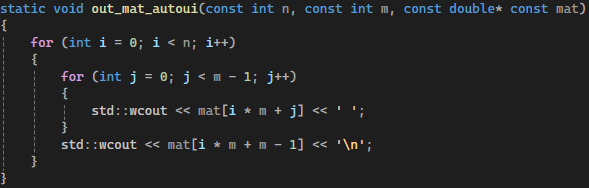
****

Рисунок 3

И функция ввода матрицы с клавиатуры:

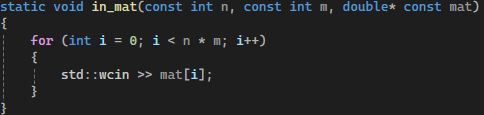


Рисунок 4

Далее был написан основной цикл программы, включающий в себя интерфейс, позволяющий вводить матрицу и команды интерактивного управления в соответствии с условиями задачи и использующий Unicode для вывода кириллицы:

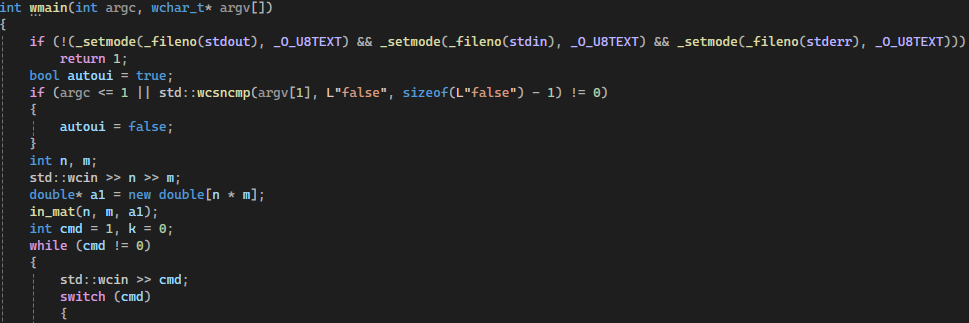


Рисунок 5

При введении команды 0 программа завершает работу, при введении команды 1 выводит матрицу на экран:

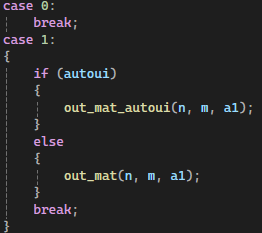


Рисунок 6

При введении команды 2 программа принимает на вход размерность и матрицу для вычисления произведения с исходной матрицей:

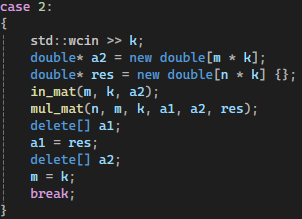


Рисунок 7

При введении команды 3 программа возводит матрицу в степень, полученную с клавиатуры с помощью ранее написанной функции. Программа проверяет возможность возведения матрицы в степень:

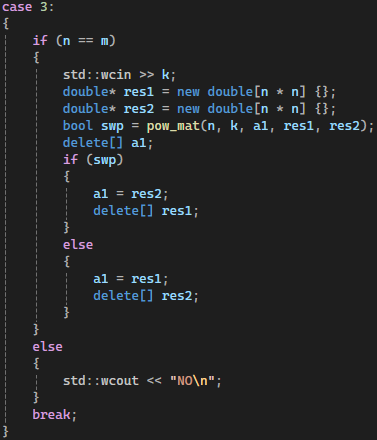


Рисунок 8

При вводе любой другой команды программа завершает работу с ошибкой:

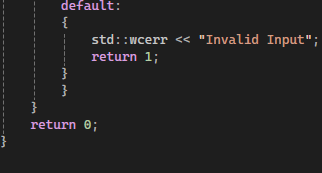


Рисунок 9

В завершении, были подключены все необходимые для работы программы библиотеки:

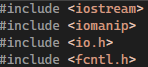


Рисунок 10

**Проверка работоспособности программы:**

Для теста были использованы входные данные из методических материалов:

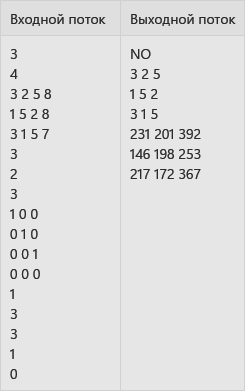


Рисунок 11

Полученные результаты:

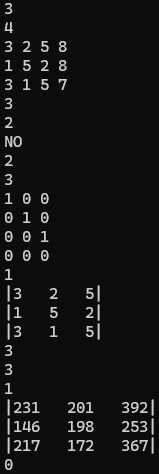


Рисунок 12

**Вывод:** В ходе выполнения лабораторной работы, целью которой была разработка программы для выполнения некоторых действий над матрицами, был получен практический опыт в нескольких ключевых областях. Было освоено создание различных типов интерфейсов, использование циклов, условных операторов и разработка функций для решения математических задач. Теоретические знания, полученные на лекциях, помогли в написании алгоритма умножения матриц.

Приложение 1

Код программы

#include <iostream>

#include <iomanip>

#include <io.h>

#include <fcntl.h>

static void mul\_mat(const int n, const int m, const int k, const double\* const mat1, const double\* const mat2, double\* const res)

{

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

{

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

{

res[i \* k + j] = 0;

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

{

res[i \* k + j] += mat1[i \* m + l] \* mat2[l \* n + j];

}

}

}

}

static void pow\_mat(const int n,const int k, const double\* const mat, double\* res1, double\* res2)

{

mul\_mat(n, n, n, mat, mat, res1);

for (int i = 1; i < k - 1; i++)

{

if (i % 2)

{

mul\_mat(n, n, n, res1, mat, res2);

}

else

{

mul\_mat(n, n, n, res2, mat, res1);

}

}

return k % 2;

}

static void out\_mat(const int n, const int m, const double\* const mat)

{

int maxlen = INT32\_MIN;

for (int i = 0; i < n \* m; i++)

{

int len = std::snprintf(nullptr, 0, "%.2f", mat[i]);

if (len > maxlen)

{

maxlen = len;

}

}

const int ALIGN\_W = maxlen;

std::wcout << '|';

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

{

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

{

if (j < m - 1)

{

std::wcout << std::left << std::setw(ALIGN\_W) << std::setfill(L' ') << mat[i \* m + j];

}

else

{

std::wcout << mat[i \* m + j] << "|\n";

if (i < n - 1)

{

std::wcout << "|";

}

}

}

}

}

static void out\_mat\_autoui(const int n, const int m, const double\* const mat)

{

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

{

for (int j = 0; j < m - 1; j++)

{

std::wcout << mat[i \* m + j] << ' ';

}

std::wcout << mat[i \* m + m - 1] << '\n';

}

}

static void in\_mat(const int n, const int m, double\* const mat)

{

for (int i = 0; i < n \* m; i++)

{

std::wcin >> mat[i];

}

}

int wmain(int argc, wchar\_t\* argv[])

{

if (!(\_setmode(\_fileno(stdout), \_O\_U8TEXT) && \_setmode(\_fileno(stdin), \_O\_U8TEXT) && \_setmode(\_fileno(stderr), \_O\_U8TEXT)))

return 1;

bool autoui = true;

if (argc <= 1 || std::wcsncmp(argv[1], L"false", sizeof(L"false") - 1) != 0)

{

autoui = false;

}

int n, m;

std::wcin >> n >> m;

double\* a1 = new double[n \* m];

in\_mat(n, m, a1);

int cmd = 1, k = 0;

while (cmd != 0)

{

std::wcin >> cmd;

switch (cmd)

{

case 0:

break;

case 1:

{

if (autoui)

{

out\_mat\_autoui(n, m, a1);

}

else

{

out\_mat(n, m, a1);

}

break;

}

case 2:

{

std::wcin >> k;

double\* a2 = new double[m \* k];

double\* res = new double[n \* k] {};

in\_mat(m, k, a2);

mul\_mat(n, m, k, a1, a2, res);

delete[] a1;

a1 = res;

delete[] a2;

m = k;

break;

}

case 3:

{

if (n == m)

{

std::wcin >> k;

double\* res1 = new double[n \* n] {};

double\* res2 = new double[n \* n] {};

bool swp = pow\_mat(n, k, a1, &res1, &res2);

delete[] a1;

if (swp)

{

a1 = res2;

delete[] res1;

}

else

{

a1 = res1;

delete[] res2;

}

}

else

{

std::wcout << "NO\n";

}

break;

}

default:

{

std::wcerr << "Invalid Input";

return 1;

}

}

}

return 0;

}