**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчЁт

**по лабораторной работе №5**

**по дисциплине «Программирование»**

Тема: Обработка двумерного массива с использованием функций

|  |  |  |  |
| --- | --- | --- | --- |
| Студент гр. 4352 |  |  | Н.А. Чехонадских |
| Преподаватель |  |  | В.А. Калмычков |

# Оглавление

[Оглавление 2](#_Toc181895887)

[1 Задание 3](#_Toc181895888)

[2 Неясности, предложения их решения и уточнения 3](#_Toc181895889)

[3 Математическая постановка 3](#_Toc181895890)

[3.1 Данные 3](#_Toc181895891)

[3.2 Заштрихованная область 3](#_Toc181895892)

[3.3 Способ решения 3](#_Toc181895893)

[4 Контрольные примеры 4](#_Toc181895894)

[5 Ограничения, обусловленные вычислительным устройством 4](#_Toc181895895)

[6 Организация интерфейса пользователя 4](#_Toc181895896)

[7 Формат файлов 5](#_Toc181895897)

[7.1 Организация входного файла 5](#_Toc181895898)

[7.2 Организация выходного файла 6](#_Toc181895899)

[8 Реализация ввода/вывода 6](#_Toc181895900)

[9 Внутреннее представление данных 6](#_Toc181895901)

[10 Описание внутренних функций 7](#_Toc181895902)

[10.1 Определение функций 7](#_Toc181895903)

[10.2 Назначение функций и им сопутствующего 7](#_Toc181895904)

[11 Описание алгоритма 8](#_Toc181895905)

[11.1 Общее описание 8](#_Toc181895906)

[11.2 Блок-схема (сделать) 8](#_Toc181895907)

[12 Текст программы 9](#_Toc181895908)

[13 Результат работы программы 11](#_Toc181895909)

[14 Вывод 12](#_Toc181895910)

# 1 Задание

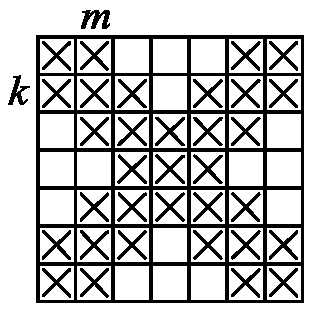


Рисунок 1.

Найти наибольший элемент заштрихованной области, зависящей от параметров и , квадратной матрицы размера (рисунок 1).

Необходимые данные: заявление о размере матрицы, параметры заштрихованной области, значения для заполнения матрицы — считать из файла.

# 2 Неясности, предложения их решения и уточнения

Принято решение игнорировать пустые строки при вводе матрицы из файла, а также игнорировать значения, непомещающиеся в матрицу с учётом незаполненных (пустых) элементов матрицы.

Принято решение брать модуль значений введённых параметров и заявления о размере матрицы, а также отбрасывать дробную часть в случае, если введённое значение нецелое, с расчётом на то, что минус (отрицательное значение) и/или дробная часть — опечатка(-и) при вводе данных.

# 3 Математическая постановка

## 3.1 Данные

.

## 3.2 Заштрихованная область

Заштрихованная область представляет из себя область исходной матрицы, которая не включает треугольноподобные области, находящиеся выше линий с их пересечением, параллельных прямым, образованным парой точек на матричной плоскости и или и , и содержащих в себе соответственно точки на матричной плоскости и , или ниже линий с их пересечением, параллельных тем же прямым, но соответственно содержащих в себе точки на матричной плоскости и , а также те треугольноподобные области, что находятся левее линий с их пересечением, параллельным тем же прямым и соответственно содержащих точки на матричной плоскости и , либо правее линий с их пересечением, параллельным тем же прямым и соответственно содержащих точки на матричной плоскости и .

**Замечания.** Ввиду неналичия на матричной плоскости точки с индексом 0 параметры ограничены положительными значениями, в противном случае заштрихованная область будет не определена.

При треугольноподобная область выходит за границы матричной плоскости (по вертикали), аналогичное наблюдается и при (по горизонтали).

## 3.3 Способ решения

Перебрать все элементы данной матрицы , обходя (исключая) треугольноподобные области, описанные в подразделе 3.2, и найти среди них наибольшее — оно и будет ответом на задание.

# 4 Контрольные примеры

| № | Содержание входного файла | Рассматриваемые значения в матрице | Ответ |
| --- | --- | --- | --- |
| 1 | 6 2 1  1 2 3 4 5 99 99 99  6 7 8 9 10  11 12 13 14 15 32 33 34  16 17 40 19 20 35 36 37 |  | 19 |
| 2 |  | — | Нет (недопустимый формат файла) |
| 3 | 7 1 3  78 40 50 60 50 11 44  18 80 45 11 21 23 10  17 89 45 62 32 62 28  33 22 22 22 12 19 46  48 12 21 54 63 12 49  47 89 85 12 45 30 57  82 61 33 57 90 66 45 |  | 90 |
| 4 | 2 2 -2.100  7.12 4.16 7.16  -2.100 |  | 7.12 |

# 5 Ограничения, обусловленные вычислительным устройством

Значения .

В связи с небесконечной точностью хранимых вещественных значений различие между ними распознаётся с точностью до 30 знаков после запятой, а именно проверка вещественных значений на равенство проводится с использованием многомалого (бесконечномалого) значения , что в общем случае проводится по формуле (1):

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

где , — сравниваемые на тождественность значения.

Положим за максимальный объем памяти, выделенный под (двумерный) массив обрабатываемых данных, 512 000 байт, тогда максимальный размер такого массива — квадратная матрица размером , т.е. и, следственно, .

# 6 Организация интерфейса пользователя

Макет ввода/вывода:

O1: Лабораторная работа №5

Автор: Н.А. Чехонадских

Версия: 1.0.2 Даты: начало 24.10.2024, конец 06.11.2024

Ограничения: -10^10 < a(i,j) < 10^10 с точностью до 30 знаков после запятой; N,k,m < 357.

Задание: Найти наибольший элемент заштрихованной области, зависящей от параметров m и k, квадратной матрицы A размера NxN.

m

+-+-+-+-+-+-+-+

|X|X| | | |X|X|

+-+-+-+-+-+-+-+

k|X|X|X| |X|X|X|

+-+-+-+-+-+-+-+

| |X|X|X|X|X| |

+-+-+-+-+-+-+-+

| | |X|X|X| | |

+-+-+-+-+-+-+-+

| |X|X|X|X|X| |

+-+-+-+-+-+-+-+

|X|X|X| |X|X|X|

+-+-+-+-+-+-+-+

|X|X| | | |X|X|

+-+-+-+-+-+-+-+

O2\*: Введите путь к файлу, который требуется открыть: \_

I2\*: [ ]["]t\_t.t\_t["][ ]

O2.1\*: Не удалось открыть файл "t\_t.t\_t".

O2.2\*: Превышено время ожидания входного файла. Попробуйте ещё раз позже.

O3\*\*\*: Недопустимый формат файла.

O4.1\*\*: Заявленный размер матрицы не может быть отрицательным и/или нецелым числом.

O4.2\*\*: Считанная матрица меньше того размера, который был заявлен.

O4.3\*\*: Значение \_ должно быть положительным.

O4.4\*\*: Значение \_ должно быть целым.

O5\*\*\*\*: Недостаточно памяти для продолжения работы программы.

O6\*\*\*\*\*: Не удалось открыть выходной файл для записи.

\* Вывод O2 и ввод I2 используются не более 4 раз, при этом каждый новый раз предоставляется пользователю по прошествию неудачной попытки открытия файла, заданного пользователем, о чём выводится сообщение об ошибке O2.1. По прошествии 4 неудачных попыток открытия заданных пользователем файлов выводится сообщение O2.2 и программа завершает свою работу.

\*\* При обнаружении недопустимых данных выводится предупреждение: при вводе отрицательного и/или нецелого числа для — O4.1, при невозможности считать матрицу заявленного размера — O4.2, при вводе отрицательного числа для — O4.3 с наименованием параметра заместо нижнего подчёркивания, нецелого числа для этих параметров — O4.4 с наименованием параметра заместо нижнего подчёркивания.

\*\*\* При невозможности считать заявление о размере матрицы и значений для параметров выводится сообщение об ошибке O3 и программа завершает свою работу.

\*\*\*\* По прошествии неудачной попытки выделить память под считанные из файла данные выводится сообщение об ошибке O5 и программа завершает свою работу.

\*\*\*\*\* По прошествии неудачной попытки открыть выходной файл для записи выводится сообщение об ошибке O6 и программа завершает свою работу.

# 7 Формат файлов

## 7.1 Организация входного файла

Макет организации файла (… — один или несколько пробелов и/или знаков табуляции, N — , K — , M — , A — ):

...N...K...M...

[...A][...A][...][

...A][...A][...]

Макет чисел N, K, M:

d\_d

Макет чисел A (вывод производится с минимальной шириной в 7 символов):

sd\_d[.d\_d][esd\_d]

## 7.2 Организация выходного файла

Макет организации файла:

Введённая матрица (d\_dxd\_d):

[sd\_d[.d\_d][esd\_d]][ sd\_d[.d\_d][esd\_d]][

sd\_d[.d\_d][esd\_d]][ sd\_d[.d\_d][esd\_d]]

Введённые параметры заштрихованной области: k=d\_d, m=d\_d.

[Максимальное значение в заштрихованной области: sd\_d[.d\_d][esd\_d].]

[Максимального значения нет. Заштрихованная область представляет из себя пустое множество или не является определённой.]

# 8 Реализация ввода/вывода

Использованы потоки ввода (cin>>) и вывода (cout<<) библиотеки «iostream», а также функции объекта cin: getline, clear, eof.

Использованы потоки чтения из файла (>>) и записи в файл (<<) библиотеки «fstream», а также функции объекта типа fstream: open, is\_open, fail, seekg, eof, close. Для настройки ввода были использованы функции setiosflags, resetiosflags библиотеки «iomanip».

Была использована библиотека «Windows.h» с предшествующим определением макроса WIN32\_LEAN\_AND\_MEAN для использования функций SetConsoleCP и SetConsoleOutputCP, настраивающих соответственно язык ввода и вывода в консоль (русский).

# 9 Внутреннее представление данных

| Имя переменной | Тип | Назначение |
| --- | --- | --- |
| f | fstream | Интерфейс ввода в файл и вывода из файла. |
| a | float\*\* | Считанная матрица . |
| res | float | Ответ на задание. |
| hr | int | Результат работы функции (успешность выполнения). |
| n | unsigned int | Значение (фактического) . |
| k | Значение параметра . |
| m | Значение параметра . |

Значения элементов считанной матрицы хранятся внутри программы следующим образом:

Изображение выглядит как текст, рукописный текст, Шрифт, письмо

Автоматически созданное описание

# 10 Описание внутренних функций

## 10.1 Определение функций

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя функции | Аргументы | | | | Возвращаемое значение |
| Входные | Выходные | Модифи-цируемые | Транзитные |
| MAX | a, b |  |  |  | float |
| MAX |  |  |  | a, b, c, d | float |
| InputFile |  | f |  |  | int |
| AnalyzeFile |  | size\_container, k\_container, m\_container | f |  | int |
| ReadArray | size |  | f |  | float\*\* |
| PrintArrayLine | line, size |  | f |  | void |
| PrintArray | array, size |  | f |  | void |
| MainOutput | size, k, m |  | f | array | void |
| AnalyzeArray | array, k, size, m | result\_container |  |  | int |
| PrintResult | result |  | f |  | void |
| EndArray | array, size |  |  |  | void |

## 10.2 Назначение функций и описание им сопутствующего

В данном разделе под словом «параметры» («параметр») подразумеваются «параметры заштрихованной области» (подраздел 3.2).

**Функция MAX** с двумя аргументами a и b типа float, передающихся по значению, определяет наибольшее среди них и возвращает его значение.

**Функция MAX** с четырьмя аргументами a, b, c и d типа float, передающихся по значению, определяет наибольшее среди них и возвращает это (максимальное) значение.

**Функция InputFile** отвечает за ввод пути ко входному файлу пользователем и открывает его с помощью переданного по ссылке объекта f типа fstream, при этом в случае 4 неудачных попыток открытия входного файла возвращается ненулевое значение, сигнализирующее о неудачном вводе файла пользователем, в ином случае — нулевое значение, сигнализирующее об успешном выполнении функции.

**Функция AnalyzeFile** анализирует данные во входном файле, открытом с помощью передаваемого по ссылке через аргумент f объекта типа fstream, и определяет размер матрицы, которую необходимо считать, в переданную по ссылке через аргумент size\_container переменную типа unsigned int с учётом заявления о размере вводимой матрицы и фактического наполнения файла, а также считывает скорректированные значения введённых параметров в переданные по ссылке через аргументы k\_container и m\_container переменные типа unsigned int. В случае обнаружения неверного формата файла, чем является отсутствие заявления о размере вводимой матрицы и значений для каждого из двух параметров, 0-ой бит (отсчёт битов ведётся с индекса 0 справа налево) возвращаемого значения будет иметь нулевое значение; при допустимом формате файла 0-ой бит возвращаемого значения будет иметь ненулевое значение. При невозможности считать матрицу заявленного размера за размер матрицы будет взят размер наибольшей квадратной матрицы, доступной для чтения, а также 27-ой бит в возвращаемом значении будет иметь ненулевое значение. Корректировка считанных значений заявления о размере вводимой матрицы и параметров следующая:

* + при отрицательном и/или нецелом значении в заявлении о размере вводимой матрицы будет использовано абсолютное значение без дробной части, а также 30-ый бит возвращаемого значения будет иметь ненулевое значение, в ином случае — нулевое;
  + при вводе отрицательного значения для параметра или будет использовано абсолютное значение, а также 29-ый или 28-ой соответственно бит возвращаемого значения будет иметь ненулевое значение, в ином случае — нулевое;
  + при вводе нецелого значения для параметра или дробная часть этого значения будет отброшена, а также 26-ой или 25-ый соответственно бит возвращаемого значения будет иметь ненулевое значение, в ином случае — нулевое.

**Функция ReadArray** выделяет память под матрицу необходимого размера, переданного по значению через аргумент size типа unsigned int, и возвращает (ненулевой) указатель на заполненную считанными из файла, открытого с помощью переданного по ссылке через аргумент f объекта типа fstream, данными матрицу (подробнее об организации хранения элементов матрицы в разделе 9), в случае невозможности выделения требуемой памяти возвращается нулевое значение.

**Функция PrintArrayLine** выводит строку из матрицы, указатель на которую передан по значению через аргумент line типа float\* (подробнее об организации хранения элементов матрицы в разделе 9), размера, переданного по значению через аргумент size типа unsigned int, в файл, открытый с помощью переданного по ссылке через аргумент f объекта типа fstream.

**Функция PrintArray** выводит матрицу, указатель на которую передан по значению через аргумент array типа float\*\* (подробнее об организации хранения элементов матрицы в разделе 9), размера, переданного по значению через аргумент size типа unsigned int, в файл, открытый с помощью переданного по ссылке через аргумент f объекта типа fstream.

**Функция MainOutput** выводит в файл, открытый с помощью переданного по ссылке через аргумент f объекта типа fstream, заголовок с обозначением размера считанной матрицы, переданного по значению через аргумент size типа unsigned int, и введённых параметров и , значения которых переданы по значению типа unsigned int через аргументы k и m соответственно, и считанную матрицу, указатель на которую передан по значению через аргумент array типа float\*\* (подробнее об организации хранения элементов матрицы в разделе 9).

**Функция AnalyzeArray** просматривает матрицу, указатель на которую передан по значению через аргумент array типа float\*\* (подробнее об организации хранения элементов матрицы в разделе 9), размера, переданного по значению через аргумент size типа unsigned int, и выполняет задание — ищет наибольшее значение в заштрихованной области с учётом параметров и , переданных по значению типа unsigned int через аргументы k и m соответственно. Найденное наибольшее значение записывается в переменную, адрес которой передан по указателю через аргумент result\_container типа float\*, и возвращается нулевое значение; в случае, если заштрихованная область представляет из себя пустое множество или не определена, возвращается ненулевое значение.

**Функция PrintResult** выводит в файл, открытый с помощью переданного по ссылке через аргумент f объекта типа fstream, результат выполнения задания (поиска наибольшего значения в заштрихованной области при её определённости), переданного с помощью указателя на переменную через аргумент result типа float\*, если аргумент result имеет ненулевое значение, в ином случае — сообщение о том, что заштрихованная область представляет из себя пустое множество или не определена.

**Функция EndArray** освобождает память, выделенную под матрицу, указатель на которую передан через аргумент array типа float\*\* (подробнее об организации хранения элементов матрицы в разделе 9), размера, переданного по значению через аргумент size типа unsigned int, если аргумент array имеет ненулевое значение, в ином случае не делает ничего.

# 11 Описание алгоритма

## 11.1 Общее описание

Получить путь к открываемому файлу от пользователя. Проанализировать файл, определить размер матрицы для чтения. Считать матрицу в динамически выделенный буфер. Вывести считанные данные в выходной файл. Перебрать числа в заштрихованной области в поиске максимального значения. Вывести ответ на задание в файл.

## 11.2 Блок-схема (сделать)

# 12 Текст программы

#define WIN32\_LEAN\_AND\_MEAN

#include <Windows.h>

#include <iostream>

#include <iomanip>

#include <fstream>

using namespace std;

float MAX(float a, float b);

float MAX(float a, float b, float c, float d);

int InputFile(fstream& f);

int AnalyzeFile(fstream& f, unsigned int& size\_container, unsigned int& k\_container, unsigned int& m\_container);

float\*\* ReadArray(fstream& f, unsigned int size);

void PrintArrayLine(fstream& f, float\* line, unsigned int size);

void PrintArray(fstream& f, float\*\* array, unsigned int size);

void MainOutput(fstream& f, float\*\* array, unsigned int size, unsigned int k, unsigned int m);

int AnalyzeArray(float\*\* array, unsigned int size, unsigned int k, unsigned int m, float\* result\_container);

void PrintResult(fstream& f, float\* result);

void EndArray(float\*\* array, unsigned int size);

int main() {

SetConsoleCP(1251); SetConsoleOutputCP(1251);

fstream f; float\*\* a {}; float res; int hr = 0; unsigned int n = 0, k, m;

cout << " Лабораторная работа №5\nАвтор: Н.А. Чехонадских\nВерсия: 1.0.2 Даты: начало 24.10.2024, конец 06.11.2024\nОграничения: -10^10 < a(i,j) < 10^10 с точностью до 30 знаков после запятой; N,k,m < 357.\nЗадание: Найти наибольший элемент заштрихованной области, зависящей от параметров m и k, квадратной матрицы A размера NxN.\n m \n +-+-+-+-+-+-+-+\n |X|X| | | |X|X|\n +-+-+-+-+-+-+-+\nk|X|X|X| |X|X|X|\n +-+-+-+-+-+-+-+\n | |X|X|X|X|X| |\n +-+-+-+-+-+-+-+\n | | |X|X|X| | |\n +-+-+-+-+-+-+-+\n | |X|X|X|X|X| |\n +-+-+-+-+-+-+-+\n |X|X|X| |X|X|X|\n +-+-+-+-+-+-+-+\n |X|X| | | |X|X|\n +-+-+-+-+-+-+-+\n\n";

if (hr = InputFile(f)) cout << "Превышено время ожидания файла. Попробуйте ещё раз позже.\n";

if (!hr) {

hr = AnalyzeFile(f, n, k, m);

if (hr & 1) {

if (hr & (1 << 30)) cout << "Заявленный размер матрицы не может быть отрицательным и/или нецелым числом.\n";

else if (hr & (1 << 27)) cout << "Считанная матрица меньше того размера, который был заявлен.\n";

if (hr & (1 << 29)) cout << "Значение " << 'k' << " должно быть положительным.\n";

if (hr & (1 << 28)) cout << "Значение " << 'm' << " должно быть положительным.\n";

if (hr & (1 << 26)) cout << "Значение " << 'k' << " должно быть целым.\n";

if (hr & (1 << 25)) cout << "Значение " << 'm' << " должно быть целым.\n";

hr = 0;

} else {

cout << "Недопустимый формат файла.\n";

hr = 2;

}

}

if (!hr) {

if (!(a = ReadArray(f, n)) && n) {

cout << "Недостаточно памяти для продолжения работы программы.\n";

hr = 1;

}

}

if (!hr) {

f.close();

f.open("результат.txt", ios::out);

if (!f.is\_open()) {

cout << "Не удалось открыть выходной файл для записи.\n";

hr = 1;

}

}

if (!hr) {

MainOutput(f, a, n, k, m);

PrintResult(f, (AnalyzeArray(a, n, k, m, &res) ? NULL : &res));

}

EndArray(a, n);

if (f.is\_open()) f.close();

EXIT: return hr;

}

int InputFile(fstream& f) {

char filename[101] {}; char\* start; unsigned int len; char attempts = 0;

while (attempts < 4) {

cout << "Введите путь к файлу, который требуется открыть: ";

cin.getline(filename, 101);

for (start = filename; \*start && \*start == ' '; ++start);

len = 0;

if (\*start == '\"') {

++start;

while (start[len] && start[len] != '\"') ++len;

start[len] = '\0';

} else if (\*start) {

while (start[len] && start[len] != ' ') ++len;

start[len] = '\0';

}

if (len) {

f.open(start, ios::in);

if (f.is\_open()) break;

else {

cout << "Не удалось открыть файл \"" << start << "\".\n";

cin.clear();

++attempts;

}

} else if (!cin.eof()) cin.clear();

}

return attempts < 4 ? 0 : 1;

}

int AnalyzeFile(fstream& f, unsigned int& size\_container, unsigned int& k\_container, unsigned int& m\_container) {

int result = 1; float buf; unsigned int n = 0; unsigned int l = 0, w = 1; char c;

f >> buf;

if (f.eof()) result &= ~1;

if (result & 1) {

if (buf < 0.f) {

buf = -buf;

result |= (1 << 30);

}

if (buf - float(n = size\_container = unsigned int(buf)) >= 1e-30f) result |= (1 << 30);

}

f >> buf;

if (f.eof()) result &= ~1;

if (result & 1) {

if (buf < 0.f) {

buf = -buf;

result |= (1 << 29);

}

if (buf - float(k\_container = unsigned int(buf)) >= 1e-30f) result |= (1 << 26);

}

f >> buf;

if (f.eof()) result &= ~1;

if (result & 1) {

if (buf < 0.f) {

buf = -buf;

result |= (1 << 28);

}

if (buf - float(m\_container = unsigned int(buf)) >= 1e-30f) result |= (1 << 25);

}

f >> buf;

if (f.fail()) result &= ~1;

if (result & 1) {

f << resetiosflags(ios::skipws);

w = 1;

while (!f.eof() && l < size\_container) {

for (;;) {

do f >> c;

while (!f.eof() && (c == ' ' || c == '\t'));

if (f.eof() || c == '\n') {

NEWLINE: if (w < size\_container) size\_container = w;

do f >> c;

while (!f.eof() && (c == ' ' || c == '\t' || c == '\n'));

if (!f.eof()) f.seekg(-1, ios::cur);

++l;

break;

}

f.seekg(-1, ios::cur);

f >> buf;

if (f.fail() || (++w) > size\_container) {

SKIP\_LINE: c = 0;

f.clear();

do f >> c;

while (!f.eof() && c != '\n');

goto NEWLINE;

}

}

w = 0;

}

if (l < size\_container) size\_container = l;

if (size\_container < n && n != 0xFFFFFFFF) result |= (1 << 27);

f.clear(); f.seekg(0);

}

return result;

}

float\*\* ReadArray(fstream& f, unsigned int size) {

float\*\* array = size ? new float\* [size] : NULL; char c {};

if (array) {

if (array[0] = new float[size \* size]) {

for (int i = size; i;) {

--i;

array[i] = array[0] + size \* i;

}

} else {

delete[] array;

array = NULL;

}

}

if (array) {

f << setiosflags(ios::skipws);

f >> \*\*array >> \*\*array >> \*\*array;

for (int y = 0; y < size; ++y) {

f << setiosflags(ios::skipws);

for (int x = 0; x < size; ++x) f >> array[y][x];

f << resetiosflags(ios::skipws);

do f >> c;

while (!f.eof() && c != '\n');

}

f.clear(); f.seekg(0);

}

return array;

}

void PrintArrayLine(fstream& f, float\* line, unsigned int size) {

while (--size) f << setw(7) << \*(line++) << ' ';

f << setw(7) << \*line << '\n';

}

void PrintArray(fstream& f, float\*\* array, unsigned int size) {

for (int i = 0; i < size; ++i) PrintArrayLine(f, array[i], size);

}

void MainOutput(fstream& f, float\*\* array, unsigned int size, unsigned int k, unsigned int m) {

f << "Введённая матрица (" << size << 'x' << size << "):\n";

PrintArray(f, array, size);

f << "\nВведённые параметры заштрихованной области: k=" << k << ", m=" << m << ".\n\n";

}

float MAX(float a, float b) {

return a > b + 1e-30f ? a : b;

}

float MAX(float a, float b, float c, float d) {

return MAX(MAX(a, b), MAX(c, d));

}

int AnalyzeArray(float\*\* array, unsigned int size, unsigned int k, unsigned int m, float\* result\_container) {

float result; unsigned int h, mid = size / 2 + 1;

if (size && k && m) {

if (k > mid) k = mid;

if (m > mid) m = mid;

result = array[0][0];

--size; --m;

while (m) {

h = 0;

do {

result = MAX(result, MAX(array[h][m], array[h][size - m], array[size - h][m], array[size - h][size - m]));

++m; ++h;

} while (m < mid);

m -= h + 1;

}

while (m < k) {

h = 0;

do {

result = MAX(result, MAX(array[m][h], array[size - m][h], array[m][size - h], array[size - m][size - h]));

++m; ++h;

} while (m < mid);

m -= h - 1;

}

} else array = NULL;

return array ? ((\*result\_container = result), 0) : 1;

}

void PrintResult(fstream& f, float\* result) {

if (result) f << "Максимальное значение в заштрихованной области: " << \*result << ".\n";

else f << "Максимального значения нет. Заштрихованная область представляет из себя пустое множество или не является определённой.\n";

}

void EndArray(float\*\* array, unsigned int size) {

if (array) {

delete[] (\*array); delete[] array;

}

}

# 13 Результат работы программы

Результат работы программы (далее — Результат) для контрольного примера №1 (раздел 4). Вывод в консоль:

Лабораторная работа №5

Автор: Н.А. Чехонадских

Версия: 1.0.2 Даты: начало 24.10.2024, конец 06.11.2024

Ограничения: -10^10 < a(i,j) < 10^10 с точностью до 30 знаков после запятой; N,k,m < 357.

Задание: Найти наибольший элемент заштрихованной области, зависящей от параметров m и k, квадратной матрицы A размера NxN.

m

+-+-+-+-+-+-+-+

|X|X| | | |X|X|

+-+-+-+-+-+-+-+

k|X|X|X| |X|X|X|

+-+-+-+-+-+-+-+

| |X|X|X|X|X| |

+-+-+-+-+-+-+-+

| | |X|X|X| | |

+-+-+-+-+-+-+-+

| |X|X|X|X|X| |

+-+-+-+-+-+-+-+

|X|X|X| |X|X|X|

+-+-+-+-+-+-+-+

|X|X| | | |X|X|

+-+-+-+-+-+-+-+

Введите путь к файлу, который требуется открыть: 1.txt

Считанная матрица меньше того размера, который был заявлен.

Вывод в файл:

Введённая матрица (4x4):

1 2 3 4

6 7 8 9

11 12 13 14

16 17 40 19

Введённые параметры заштрихованной области: k=2, m=1.

Максимальное значение в заштрихованной области: 19.

Результат для контрольного примера №2 (раздел 4). Вывод в консоль (в сокращении):

...

Введите путь к файлу, который требуется открыть: 2.txt

Недопустимый формат файла.

Вывод в файл не проводился.

Результат для контрольного примера №4 (раздел 4). Вывод в консоль (в сокращении):

...

Введите путь к файлу, который требуется открыть: 4.txt

Считанная матрица меньше того размера, который был заявлен.

Значение m должно быть положительным.

Значение m должно быть целым.

Вывод в файл:

Введённая матрица (1x1):

7.12

Введённые параметры заштрихованной области: k=2, m=2.

Максимальное значение в заштрихованной области: 7.12.

# 14 Вывод

Были изучены аспекты использования ссылок, указателей, двумерных массивов и функций в языке программирования C++, а также способы их задания и объявления, и была пройдена практика по использованию функций и двумерных массивов при обработке матрицы данных.