Федеральное агентство по образованию РФ

Государственное образовательное учреждение

высшего профессионального образования

“Тверской государственный университет”

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Факультет прикладной математики и кибернетики

Направление: 010300.62 Прикладная математика и информатика

**Отчет по учебно-вычислительной практике**

 Выполнила:

 студентка 24 группы

Киут Л.Л.

Научный руководитель:

Дадеркин Д. О.

Задание на учебно-вычислительную практику

Вариант 18

**Задача 2.**

Создать класс «Матрица n, m3», где n, m – количество строк и столбцов в матрице. Каждый элемент, кроме элементов в крайних строках, должен быть связан с соседними (звено 4-связного списка).Найти сумму двух матриц.

Содержание

1. Словесное описание алгоритма……………………………………………………………....4

2.Описание классов: переменные и методы. Типы доступа. Объявление методов............. 4

3. Тестовый запуск……………………………………………………………………………….5

4. Код программы ………………………………………………………………….…………....6

5. Список использованной литературы………………………………………………………. 9

***1. Словесное описание алгоритма.***

Программа позволяет работать с матрицами, элементы которых связаны с соседними. В данной программе реализована операция сложения матриц.

***2. Описание классов: переменные и методы. Типы доступа. Объявление методов***

class MATRIX\\Класс матрицы

{

private:

 int n; \\ Число строк

 int m; \\Число столбцов

 chamber \*\*a;\\ Матрица

public:

 chamber &RETURN\_a(int i, int j) \\Метод возврата элемента

 MATRIX()\\конструктор

 ~MATRIX()\\деструктор

 MATRIX(MATRIX &M) \\конструктор копирования

 MATRIX(int x, int y) \\конструктор с параметрами

 int RETURN\_n()\\Метод возврата n

 int RETURN\_m()\\Метод возврата m

 void INITIATE()\\Метод инициализации

 void SET\_NULL()\\Метод заполнения матрицы

 void SET\_SMART()\\Метод заполнения матрицы

 void SET\_FULL(int x) \\Метод заполнения матрицы

 void SET\_LINES()\\Метод заполнения матрицы

 void TRANS()\\Метод транспонирования

 void PRINT()\\Метод печати

 MATRIX& operator=(const MATRIX& M) \\переопределение оператора присваивания

 const MATRIX operator+(MATRIX & left,MATRIX & right) \\ переопределение оператора сложения

***3. Тестовые запуски***



***4. Код программы***

#include <iostream>

#include <string>

using namespace std;

struct Node{

 int x;

 Node \*right;

 Node \*left;

 Node \*up;

 Node \*down;

};

class Matrix{

 int n;

 int m;

 Node \*first;

public:

 Matrix()

 {

 n = 1;

 m = 1;

 first = NULL;

 AddElem(0,0,0);

 }

 Matrix(int n1, int m1)

 {

 n = n1;

 m = m1;

 first = NULL;

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

 {

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

 {

 AddElem(0,i,j);

 }

 }

 }

 Matrix(const Matrix &M)

 {

 n = M.n;

 m = M.m;

 first = NULL;

 Node \*t = M.first;

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

 {

 Node \*el = t;

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

 {

 AddElem(el->x,i,j);

 el = el->right;

 }

 t = t->down;

 }

 }

 ~Matrix()

 {

 Node \*t;

 while(first != NULL)

 {

 t = first;

 first = first->down;

 while(t)

 {

 Node \*del = t;

 t = t->right;

 delete del;

 }

 }

 first = NULL;

 }

 void AddElem(int x, int i, int j)

 {

 if(i >= n || j>=m)

 {

 cout<<"Error in method AddElem (i >= n || j>=m)";

 return;

 }

 Node \*temp = new Node;

 temp->right = NULL;

 temp->down = NULL;

 temp->x = x;

 Node \*t = first;

 if(first != NULL)

 {

 Node \*t\_up = NULL;

 for(int i1=0; i1<i; i1++)

 {

 t\_up = t;

 t = t->down;

 }

 if(j==0)

 {

 t\_up->down = temp;

 temp->left = NULL;

 temp->up = t\_up;

 }

 else

 {

 for(int j1=0; j1<j-1; j1++)

 {

 t = t->right;

 if(i!=0)

 t\_up = t\_up->right;

 }

 if(i != 0)

 {

 t\_up = t\_up->right;

 t\_up->down = temp;

 }

 t->right = temp;

 temp->left = t;

 temp->up = t\_up;

 }

 }

 else

 {

 first = temp;

 first->left = NULL;

 first->up = NULL;

 }

 }

 void Set(int x, int i, int j)

 {

 if(i >= n || j>=m)

 {

 cout<<"Error in method Set (i >= n || j>=m)";

 return;

 }

 Node \*t = first;

 for(int i1=0; i1<i; i1++)

 {

 t = t->down;

 }

 for(int j1=0; j1<j; j1++)

 {

 t = t->right;

 }

 t->x = x;

 }

 Matrix& operator=(const Matrix &M)

 {

 if(n!=M.n || m!=M.m)

 {

 n = M.n;

 m = M.m;

 Node \*t;

 while(first != NULL)

 {

 t = first;

 first = first->down;

 while(t)

 {

 Node \*del = t;

 t = t->right;

 delete del;

 }

 }

 first = NULL;

 t = M.first;

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

 {

 Node \*el = t;

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

 {

 AddElem(el->x,i,j);

 el = el->right;

 }

 t = t->down;

 }

 }

 else{

 Node \*t = M.first;

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

 {

 Node \*el = t;

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

 {

 Set(el->x,i,j);

 el = el->right;

 }

 t = t->down;

 }

 }

 return \*this;

 }

 Matrix operator+(const Matrix &M)

 {

 if(n!=M.n || m!=M.m)

 {

 cout<<"Ошибка при сложении матриц. Размерности матриц не совпадают\n";

 exit(1);

 }

 Matrix Res(n,m);

 Node \*t1 = first;

 Node \*t2 = M.first;

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

 {

 Node \*el1 = t1;

 Node \*el2 = t2;

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

 {

 Res.Set(el1->x + el2->x, i, j);

 el1 = el1->right;

 el2 = el2->right;

 }

 t1 = t1->down;

 t2 = t2->down;

 }

 return Res;

 }

 //перегрузка вывода

 template <class C>

 friend basic\_ostream<C>& operator<<(basic\_ostream<C> &out,Matrix &A)

 {

 Node \*t = A.first;

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

 {

 Node \*el = t;

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

 {

 out<<el->x<<" ";

 el = el->right;

 }

 t = t->down;

 out<<endl;

 }

 return out;

 }

 //перегрузка ввода

 template <class C>

 friend basic\_istream<C>& operator>>(basic\_istream<C> &in,Matrix &A)

 {

 char ch;

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

 {

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

 {

 cout<<"M["<<i<<"]["<<j<<"] = ";

 in>>ch;

 if( !isdigit(ch) )

 {

 cout<<"Ошибка ввода. Нужно вводить цифры\n";

 exit(1);

 }

 else

 A.Set(ch - '0',i,j);

 }

 }

 return in;

 }

};

int main()

{

 setlocale(LC\_ALL, "rus");

 Matrix A(3,3);

 Matrix B(3,3);

 cout<<"Заполните первую матрицу.\n";

 cin>>A;

 cout<<"Вывод первой матрицы:\n"<<A<<endl;

 cout<<"Заполните вторую матрицу.\n";

 cin>>B;

 cout<<"Вывод второй матрицы:\n"<<B<<endl;

 cout<<"Результат сложения матриц:\n"<<A+B<<endl;

 system("pause");

 return 0;

}

Список использованной литературы

1. Камаев В. А. Технологии программирования : [учебник для вузов по направлению "Информатика и вычислительная техника"] / В. А. Камаев, В. В. Костерин. - М., 2006. - 453, [1] с. : ил., табл. - Рекомендовано МО.
2. Крылов Е. В. Техника разработки программ. В 2 кн.. Кн. 1 : [учебник для вузов по направлениям "Информатика и вычислительная техника" и "Техника и технологии"] / Е. В. Крылов, В. А. Острейковский, Н. Г. Типикин. - М., 2007. - 374, [1] с. : ил. - Рекомендовано УМО.
3. Хорев П. Б. Объектно-ориентированное программирование : [учебное пособие по направлению "Информатика и вычислительная техника"] / П. Б. Хорев. - М., 2011. - 446, [1] с. : ил. - Рекомендовано УМО.
4. Шеферд Д. Программирование на Microsoft Visual C++ .NET : мастер-класс [пер. с англ.] / Джордж Шеферд по материалам Дэвида Круглински. - М., 2007. - 892 с. : ил. + 1 CD-ROM.
5. Базы данных: модели, разработка, реализация/ Т.С. Карпова. - СПб.: Питер, 2001г.