// lab4.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <stdlib.h>

#include <iostream>

using namespace std;

int modal;

class Element

{ protected:

int a; // длина стороны

const static int modal =1000;

public:

Element(int a) // Конструктор для инициализации полей

{

this->a=a%modal;

}

Element() // Конструктор для инициализации полей

{

this->a=0;

}

virtual void print() // Метод для печати полей (виртуальный)

{

cout<<a; // Печатаем значения полей

}

Element operator+(Element& s)

{

int g=a+s.a;

Element q(g);

return q;

}

Element operator\*(Element& s)

{

int g=a\*s.a;

Element m(g);

return m;

}

virtual ~Element()

{

}

};

class Matrix

{protected:

Element\*\* a;

int stroki, stolbcu;

public:

Matrix(Element\*\* dv, int sk, int sb)

{

a = new Element\*[sk];

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

{

a[i] = new Element[sb];

}

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

{

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

{

if (dv != 0)

{

a[i][j] = dv[i][j];

}

else a[i][j] = 0;

}

}

stroki = sk;

stolbcu = sb;

}

Element poluch(int c, int b)

{

return a[c][b];

}

void vstav(int c, int b, Element d)

{

a[c][b] = d;

return;

}

Matrix operator |(const Matrix& m)

{

Matrix q(0, stroki, (stolbcu+m.stolbcu));

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

{

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

{

if (j < stolbcu)

{

q.a[i][j] = a[i][j];

}

else q.a[i][j] = m.a[i][j];

}

}

return q;

}

Matrix operator +(const Matrix& m)

{

Matrix q(0, stroki, stolbcu);

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

{

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

{

}

};

int main()

{ int a1,b1;

cout << "vvedite pervoe chislo ";

cin >>a1;

cout << "vvedite vtoroe chislo ";

cin>>b1;

Element a(a1);

Element b(b1);

Element \*pp;

pp=&a;

cout<<"a=";

pp->print();

cout << endl;

cout<<"a+b=";(a+b).print();

cout << endl;

cout<<"a\*b=";(a\*b).print();

cout<<endl;

system("pause");

return 0;

}