Для раскрытия интерфейса требуется определить в классе Matrix

все методы, которые указаны в интерфейсе. Приведем код класса

Matrix с указанием переопределенных операций.

class Matrix : IMathObject

{

// количество строк и столбцов матрицы

protected int m, n;

// массив элементов матрицы

protected double[,] a;

// конструктор - осуществляет выделение

// памяти под хранение матрицы

public Matrix(int m1, int n1)

{

n = n1;

m = m1;

a = new double[m, n];

}

// определение операции сложения двух матриц,

// раскрывающей метод интерфейса IMathObject

public IMathObject Summa(IMathObject ob)

{

// приведение типа аргумента к классу Matrix

Matrix ob1 = ob as Matrix;

if (m != ob1.m || n != ob1.n)

throw new Exception("Сложение таких матриц невозможно");

Matrix res = new Matrix(m, n);

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

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

res[i, j] = a[i, j] + ob1[i, j];

return res;

}

// определение операции вычитания двух матриц,

// раскрывающей метод интерфейса IMathObject

public IMathObject Substract(IMathObject ob)

{

Matrix ob1 = ob as Matrix;

if (m != ob1.m || n != ob1.n)

throw new Exception("Вычитание таких

матриц невозможно");

Matrix res = new Matrix(m, n);

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

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

res[i, j] = a[i, j] - ob1[i, j];

return res;

}

// определение операции умножения двух матриц,

// раскрывающей метод интерфейса IMathObject

public IMathObject Multiply(IMathObject ob)

{

Matrix ob1 = ob as Matrix;

if (n != ob1.m)

throw new Exception("Такие матрицы перемножить нельзя");

Matrix res = new Matrix(m, ob1.n);

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

for (int j = 0; j < ob1.n; j++)

{

res[i, j] = 0;

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

res[i, j] = res[i, j] + a[i, k] \* ob1[k, j];

}

return res;

}

// определение операции умножения матрицы на число,

// раскрывающей метод интерфейса IMathObject

public IMathObject Multiply(double chislo)

{

Matrix res = new Matrix(m, n);

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

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

res[i, j] = a[i, j] \* chislo;

return res;

}

// индексатор для получения элементов матрицы по индексам

double this[int i, int j]

{

get { return a[i, j]; }

set { a[i, j] = value; }

}

// метод ввода матрицы

public void Input()

{

Console.WriteLine("Введите элементы матрицы");

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

{

string str = Console.ReadLine();

string [] s = str.Split(' ');

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

a[i, j] = double.Parse(s[j]);

}

}

// метод получения строкового представления матрицы.

// Элементы в матрице располагаются через символ

// табуляции, при переходе на новую строку матрицы

// добавляется символ ‘\n’

public override string ToString()

{

string str = "";

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

{

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

str = str + a[i, j] + "\t";

str = str + a[i, n - 1] + "\n";

}

return str;

}

}