**9 Интерфейсы**

Задание №1. Интерфейсы Ix, Iy, Iz, содержат объявления методов с одной и той же сигнатурой, эти интерфейсы наследуются в классе TestClass, содержащий член w типа параметр и реализуются так, как задано в варианте. В каждом методе задать вывод результата. Задаётся параметр – int, IxF0, IxF1 возвращают w + 5, F0 и F1 возвращают: неявная реализация – w3, явная реализация Iz – 7w - 2.

Листинг программы:

TestClass testClass = new TestClass();

TestClass testClass1 = new TestClass(2);

Console.WriteLine($"IxF0 - "); testClass.IxF0(2);

Console.WriteLine($"IxF1 - "); testClass1.IxF1();

Console.WriteLine("неявная неоднозначная реализация: ");

testClass.F0(2);

testClass1.F1();

Console.WriteLine("Явное приведение к типу интерфейса: ");

(testClass as Iy).F0(3);

(testClass1 as Iz).F1();

Console.WriteLine("Ix --->");

Ix ix = testClass1;

ix.IxF0(2);

ix.IxF1();

Console.WriteLine("Iy --->");

Iy iy = testClass1;

iy.F0(2);

iy.F1();

Console.WriteLine("Iz --->");

Iz iz = testClass1;

iz.F0(2);

iz.F1();

interface Ix

{

void IxF0(int w);

void IxF1();

}

interface Iy

{

void F0(int w);

void F1();

}

interface Iz

{

void F0(int w);

void F1();

}

class TestClass : Ix, Iy, Iz

{

int w;

public TestClass()

{

w = 63;

}

public TestClass(int W)

{

w = W;

}

public void IxF0(int w)

{

Console.WriteLine(w + 5);

}

public void IxF1()

{

Console.WriteLine(w + 5);

}

public void F0(int w)

{

Console.WriteLine(Math.Pow(w, 3));

}

public void F1()

{

Console.WriteLine(Math.Pow(w, 3));

}

void Iz.F0(int w)

{

Console.WriteLine(7 \* w - 2);

}

void Iz.F1()

{

Console.WriteLine(7 \* w - 2);

}

}

Таблица 9.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 2, 2, 3, 2, 2 | 7, 7, 8, 8, 27, 12 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 9.1.

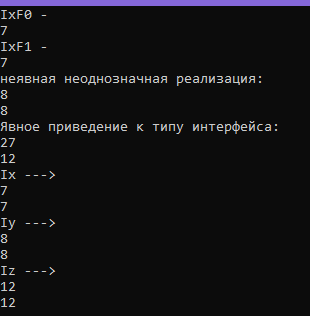


Рисунок 9.1 – Результат работы программы

Источник: собственная разработка

Задание №2. Описать класс, обеспечивающий представление матрицы произвольного размерам с возможностью изменения числа строк и столбцов, вывода на экран подматрицы любого размера и всей матрицы, доступ по индексам к элементу матрицы.

Написать программу, демонстрирующую все разработанные элементы класса.

Листинг программы:

namespace new\_V4\_p4

{

class Program

{

static void Main(string[] args)

{

int N, M, choose = 1;

Console.WriteLine("Enter matrix dimensions (N x M): ");

N = int.Parse(Console.ReadLine());

M = int.Parse(Console.ReadLine());

Matrix matrix = new Matrix(N, M);//Создали матрицу

Console.WriteLine("fill matrix: ");

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

{

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

{

matrix[i, j] = Double.Parse(Console.ReadLine());

}

}

while (choose != 5)

{

Console.WriteLine("Choose operation:\n 1 - Print matrix\n 2 - Print submatrix\n 3 - Change matrix \n 4 - Get value\n 5 - Exit\nEnter number: ");

string choice = Console.ReadLine();

choose = Convert.ToInt32(choice);

switch (choose)

{

case 1:

{

Console.WriteLine("=====Matrix=====");

matrix.Print();

Console.WriteLine();

break;

}

case 2:

{

Console.WriteLine("=====Submatrix=====");

Console.WriteLine("Enter coordinates: ");

matrix.Print1();

Console.WriteLine();

break;

}

case 3:

{

Console.WriteLine("Enter dimensions (N x M): ");

N = int.Parse(Console.ReadLine());

M = int.Parse(Console.ReadLine());

matrix = new Matrix(N, M);

Console.WriteLine("Fill matrix: ");

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

{

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

{

matrix[i, j] = Double.Parse(Console.ReadLine());

}

}

break;

}

case 4:

{

Console.WriteLine("Enter coordinates: ");

N = int.Parse(Console.ReadLine());

M = int.Parse(Console.ReadLine());

Console.WriteLine($"Value: {matrix[N, M]}");

break;

}

}

}

Console.ReadLine();

}

}

}

class Matrix

{

private double[,] matrix;

private int N, M;

public Matrix()

{

matrix = new Double[0, 0];

this.N = 0;

this.M = 0;

}

public Matrix(int N, int M)

{

matrix = new Double[N, M];

this.N = N;

this.M = M;

}

public int iN

{

get

{

return N;

}

set

{

this.N = value;

}

}

public int iM

{

get

{

return M;

}

set

{

this.M = value;

}

}

public double this[int x, int y]

{

set

{

if (N >= x && M >= y && x >= 0 && y >= 0)

{

matrix[x, y] = value;

}

else

{

Console.WriteLine("Invalid coordinate!");

matrix[x, y] = 0;

}

}

get

{

if (N >= x && M >= y && x >= 0 && y >= 0)

{

return matrix[x-1, y-1];

}

else

{

Console.WriteLine("Invalid coordinate!");

return 0;

}

}

}

public void Print()

{

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

{

Console.WriteLine();

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

{

Console.Write(" {0} ", matrix[i, j]);

}

}

Console.WriteLine();

}

public void Print1()

{

int r = int.Parse(Console.ReadLine());

int s = int.Parse(Console.ReadLine());

Console.WriteLine("Submatrix: ");

if (r <= N && s <= N)

{

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

{

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

{

Console.Write($" {matrix[i, j]} ");

}

Console.WriteLine();

}

}

}

}

Таблица 9.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 2, 2, 4, 7, 9, 1 | Matrix, submatrix, 7 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 9.2.

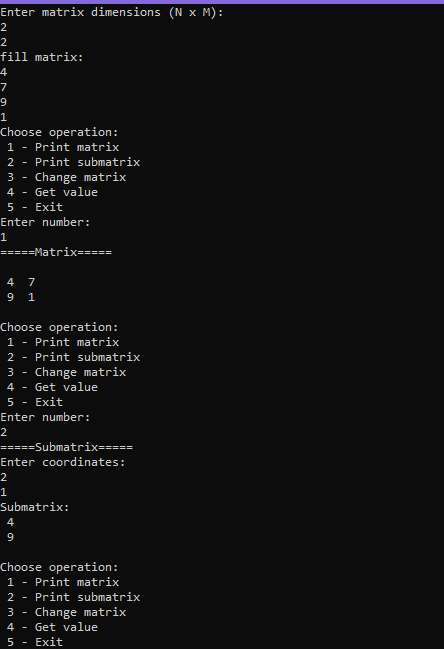


Рисунок 9.2 – Результат работы программы

Источник: собственная разработка