1. **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.

Рассмотреть случай

- неявной реализации интерфейсов

- явной реализации интерфейса Iz

В программе должна выполняться:

- неявная неоднозначная реализация методов интерфейсов Iy и Iz,

- вызов функций с явным приведением к типу интерфейса,

- вызов метода для объекта посредством интерфейсной ссылки.

Листинг программы:

try

{

Console.WriteLine("Введите w: ");

int w = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Введите x");

int f = Convert.ToInt32(Console.ReadLine());

TestClass test = new TestClass(w);

Ix x = test;

Iy y = test;

Iz z = test;

x.IxF0(f);

x.IxF1();

y.F0(f);

y.F1();

z.F0(f);

z.F1();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

interface Ix

{

void IxF0(int w);

void IxF1();

}

interface Iy

{

void F0(int w);

void F1();

}

interface Iz

{

void F0(int w);

void F1();

}

public class TestClass : Ix, Iy, Iz

{

private int \_w;

public TestClass(int w)

{

\_w = w;

}

public void IxF0(int w)

{

Console.WriteLine(7 \* w);

}

public void IxF1()

{

Console.WriteLine(7 \* \_w);

}

public void F0(int w)

{

Console.WriteLine(w \* 8);

}

public void F1()

{

Console.WriteLine(\_w \* 8);

}

void Iz.F0(int w)

{

Console.WriteLine(6 + w);

}

void Iz.F1()

{

Console.WriteLine(6 +\_w);

}

}

Таблица 9.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 3,2 | 14,21,16,24,8,9 |

Анализ результатов:

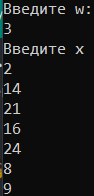


Рисунок 9.1 – Результат работы программы

Источник: собственная разработка

Задание 2. Описать класс для работы с одномерным массивом целых чисел (вектором). Обеспечить следующие возможности:

- задание произвольных целых границ индексов при создании объекта;

- обращение к отдельному элементу массива с контролем выхода за пределы массива;

- выполнение операций поэлементного сложения и вычитания массивов с одинаковыми границами индексов;

- вывод на экран элемента массива по заданному индексу и всего массива. Написать программу, демонстрирующую. все разработанные элементы класса.

Листинг программы:

try

{

Arr[] arr = new Arr[2];

for (int i = 0; i < arr.Length; i++)

{

Console.Write("Введите длину массива: ");

int len = Convert.ToInt32(Console.ReadLine());

Console.Write("Введите левую границу массива: ");

int leftBorder = Convert.ToInt32(Console.ReadLine());

Console.Write("Введите правую границу массива: ");

int rightBorder = Convert.ToInt32(Console.ReadLine());

arr[i] = new Arr(len, leftBorder, rightBorder);

arr[i].Fill();

}

for (int i = 0; i < arr.Length; i++)

{

Console.Write("Введите индекс элемента: ");

int index = Convert.ToInt32(Console.ReadLine());

arr[i].FindElem(index);

}

Console.WriteLine();

arr[0].SubSum(arr);

Console.WriteLine();

for (int i = 0; i < arr.Length; i++)

{

Console.WriteLine("Введите индекс элемента: ");

int index = Convert.ToInt32(Console.ReadLine());

arr[i].PrintElem(index);

Console.WriteLine();

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

public class Arr

{

int[] mass;

int leftBorder;

int rightBorder;

public Arr(int len, int left, int right)

{

mass = new int[len];

leftBorder = left;

rightBorder = right;

}

Random rnd = new Random();

public int[] Fill()

{

for (int i = 0; i < mass.Length; i++)

mass[i] = rnd.Next(leftBorder, rightBorder+1);

return mass;

}

public void FindElem(int index)

{

if (index >= 0 && index < mass.Length)

Console.WriteLine(mass[index]);

else

Console.WriteLine("Индекс выходит за пределы массива");

}

public void SubSum(Arr[] arr)

{

for (int i = 1; i < arr.Length; i++)

if (arr[0].leftBorder == arr[i].leftBorder && arr[0].rightBorder == arr[i].rightBorder)

{

int[] sum = arr[0].mass.Length > arr[i].mass.Length ? new int[arr[0].mass.Length] : new int[arr[i].mass.Length];

for (int j = 0; j < sum.Length; j++)

{

sum[j] = arr[0].mass[j] + arr[i].mass[j];

Console.Write(sum[j] + " ");

}

Console.WriteLine();

int[] sub = arr[0].mass.Length > arr[i].mass.Length ? new int[arr[0].mass.Length] : new int[arr[i].mass.Length];

for (int j = 0; j < sub.Length; j++)

{

sub[j] = arr[0].mass[j] - arr[i].mass[j];

Console.Write(sub[j] + " ");

}

}

}

public void PrintElem(int index)

{

Console.WriteLine(mass[index]);

for (int i = 0; i < mass.Length; i++)

Console.Write(mass[i] + " ");

}

}

Анализ результатов:

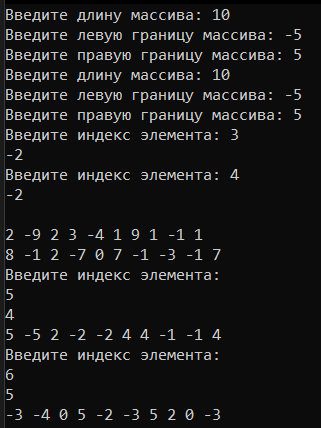


Рисунок 9.2 – Результат работы программы

Источник: собственная разработка