**12 Делегаты**

Задание 1. Объявить тип делегата, который ссылается на метод. Требования к сигнатуре метода следующие:

метод получает входным параметром переменную типа double;

метод возвращает значение типа double, которое есть результатом вычисления.

Реализовать вызов методов с помощью делегата, которые получают радиус R и вычисляют:

длину окружности по формуле D = 2 · π · R;

площадь круга по формуле S = π · R 2;

объем шара. Формула: V = 4/3 \* π · R 3.

Методы должны быть объявлены как статические (с использованием ключевого слова static).

Листинг программы:

double radius;

double length, area, volume;

Figures FG;

FG = new Figures();

CalcFigures CF;

radius = Convert.ToDouble(Console.ReadLine());

CF = FG.Get\_Length;

length = CF(radius);

Console.WriteLine(Math.Round(length,4));

CF = FG.Get\_Area; // присвоить делегату метод Get\_Area

area = CF(radius);

Console.WriteLine(Math.Round(area, 4));

CF = FG.Get\_Volume; // присвоить делегату метод Get\_Volume

volume = CF(radius); // вызов метода Get\_Volume

Console.WriteLine(Math.Round(volume,4));

class Figures

{

public double Get\_Length(double r)

{

double length;

length = 3.1415 \* 2 \* r;

return length;

}

public double Get\_Area(double r)

{

double area;

area = 3.1415 \* r \* r;

return area;

}

public double Get\_Volume(double r)

{

double volume;

volume = 4.0 / 3.0 \* 3.1415 \* r \* r \* r;

return volume;

}

}

delegate double CalcFigures(double r);

Таблица 12.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 790 | 490.074, 19112.886, 1987740. 144 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 12.1.

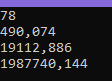


Рисунок 12.1 – Результат работы программы

Источник: собственная разработка

Задание 2. Используя Visual Studio, создать проект по шаблону Console Application.

Создайте четыре лямбда оператора для выполнения арифметических действий: (Add – сложение, Sub – вычитание, Mul – умножение, Div – деление). Каждый лямбда оператор должен принимать два аргумента и возвращать результат вычисления. Лямбда оператор деления должен делать проверку деления на ноль.

Написать программу, которая будет выполнять арифметические действия, указанные пользователем.

Листинг программы:

string a = " ";

while (a.Length > 0)

{

try

{

Console.WriteLine("Введите действие - \nCложение(+)\nВычитание(-)\nУмножение(\*)\nДеление(/)");

a = Console.ReadLine();

Console.WriteLine();

if (a.Length > 0)

{

Console.Write("Первое число: ");

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

Console.Write("Второе число: ");

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

Console.WriteLine();

switch (a)

{

case "+":

MyDelegat myDelegat = (c, b) => { return c + b; };

double del = myDelegat(x, y);

Console.WriteLine($"Ответ:{del}\n");

break;

case "-":

MyDelegat myDelegat1 = (c, b) => { return c - b; };

double del1 = myDelegat1(x, y);

Console.WriteLine($"Ответ:{del1}\n");

break;

case "\*":

MyDelegat myDelegat2 = (c, b) => { return c \* b; };

double del2 = myDelegat2(x, y);

Console.WriteLine($"Ответ:{del2}\n");

break;

case "/":

MyDelegat myDelegat3 = (c, b) => { return b == 0 ? throw new DivideByZeroException() : c / b; };

double del3 = myDelegat3(x, y);

Console.WriteLine($"Ответ:{del3}\n");

break;

default:

Console.WriteLine("Выберите из предложанного");

break;

}

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

public delegate double MyDelegat(int a, int b);

Таблица 12.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 4, 5, 5, 5 | 9, 25 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 12.2.

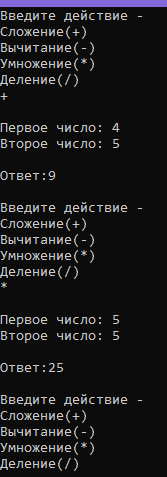


Рисунок 12.2 – Результат работы программы

Источник: собственная разработка

Задание 3. Создать 3 метода для работы со строкой. Создать делегат. В программе вызывать все три метода при помощи делегата.

Листинг программы:

try

{

Console.Write("Введите строку: ");

string str = Console.ReadLine();

Message msg = ChangeReg;

msg += Substr;

msg += Separate;

msg(str);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

void ChangeReg(string str)

{

Console.WriteLine(str.ToUpper());

}

void Substr(string str)

{

Console.WriteLine(str.Substring(2));

}

void Separate(string str)

{

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

foreach (string word in words)

Console.WriteLine(word);

}

delegate void Message(string str);

Таблица 12.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| привет | ПРИВЕТ, ивет, привет |

Источник: собственная разработка

Анализ результатов представлен на рисунке 12.3.

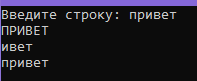


Рисунок 12.3 – Результат работы программы

Источник: собственная разработка

Задание 4. Используя Visual Studio, создайте проект по шаблону Console Application.

Создайте анонимный метод, который принимает в качестве аргумента массив делегатов и возвращает среднее арифметическое возвращаемых значений методов, сообщенных с делегатами в массиве. Методы, сообщенные с делегатами из массива, возвращают случайное значение типа int.

Листинг программы:

public delegate int Number();

public delegate double MediumCalc(Number[] arrayX);

class Program

{

static Random rand = new Random();

public static int Randomizer() //функция рандома

{

return rand.Next(1, 10);

}

static void Main(string[] args)

{

Number[] numArr = new Number[5]; //создание массива через анонимную функцию

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

{

numArr[i] = Randomizer; // генерация рандомных чисел в массиве

}

MediumCalc mediumCalc = delegate (Number[] arrayX) //выполнение анонимной функции

{

var arrayT = new int[5];

int sum = 0;

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

{

arrayT[i] = arrayX[i](); //присваимваем элементам нового массива arrayT, т.к. если проосто прибавлять arrayX[i](), то будут рандомиться новые числа в массиве arrayX, следовательно среднее арифметическое будет неправильным

sum += arrayT[i];

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

}

return ((double)(sum) / (arrayX.Length));

};

Console.WriteLine($"\nResult = {mediumCalc(numArr)}"); //вызов анонимной функции

Console.ReadLine();

}

}

Таблица 12.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Delegate array | 3.6 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 12.4.



Рисунок 12.4 – Результат работы программы

Источник: собственная разработка