**13 «СОБЫТИЯ».**

Задание 1. Задача 1. Создать 3 метода, подсчитывающих периметр, площадь

квадрата, и выводящий сторону треугольника. Создать делегат. В программе вызывать все три метода при помощи делегата.

Листинг программы:

using System;

public delegate void MessageDelegate();

class Program

{

static void Message1()

{

Console.WriteLine("Это сообщение 1");

}

static void Message2()

{

try

{

Console.WriteLine("Это сообщение 2");

throw new Exception("Произошла исключительная ситуация в методе 2");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

static void Message3()

{

Console.WriteLine("Это сообщение 3");

}

static void Main(string[] args)

{

MessageDelegate messageDelegate1 = Message1;

MessageDelegate messageDelegate2 = Message2;

MessageDelegate messageDelegate3 = Message3;

messageDelegate1();

messageDelegate2();

messageDelegate3();

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Это сообщение 1  Это сообщение 2  Произошла исключительная ситуация в методе 2  Это сообщение 3 |

Анализ результатов:

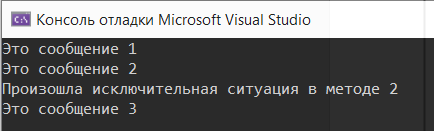


Рисунок 1.1 – Результат работы программы

Задание 2. Задача №2. На основе своей программы создать метод, в котором делегат будет параметром.

Листинг программы:

using System;

public delegate void MessageDelegate();

class Program

{

static void Message1()

{

Console.WriteLine("Это сообщение 1");

}

static void Message2()

{

try

{

Console.WriteLine("Это сообщение 2");

throw new Exception("Произошла исключительная ситуация в методе 2");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

static void Message3()

{

Console.WriteLine("Это сообщение 3");

}

static void ExecuteMessage(MessageDelegate messageDelegate)

{

messageDelegate();

}

static void Main(string[] args)

{

MessageDelegate messageDelegate1 = Message1;

MessageDelegate messageDelegate2 = Message2;

MessageDelegate messageDelegate3 = Message3;

ExecuteMessage(messageDelegate1);

ExecuteMessage(messageDelegate2);

ExecuteMessage(messageDelegate3);

}

}

Таблица 1.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Это сообщение 1  Это сообщение 2  Произошла исключительная ситуация в методе 2  Это сообщение 3 |

Анализ результатов:

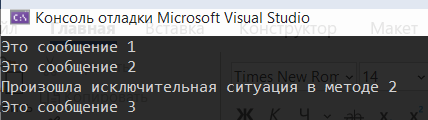


Рисунок 1.2 – Результат работы программы

Задание 3. Для класса MyInfo содержащего поле name (Ваше имя) создать событие выводящее оповещение в случае изменения значения поля name.

Листинг программы:

using System;

public class MyInfo

{

public delegate void NameChangedDelegate(string oldName, string newName);

public event NameChangedDelegate NameChanged;

private string \_name;

public string Name

{

get { return \_name; }

set

{

if (\_name != value)

{

string oldName = \_name;

\_name = value;

NameChanged?.Invoke(oldName, \_name);

}

}

}

}

class Program

{

static void Main(string[] args)

{

MyInfo myInfo = new MyInfo();

myInfo.NameChanged += (oldName, newName) =>

{

Console.WriteLine($"Имя изменено с {oldName} на {newName}");

};

myInfo.Name = "Ваше имя";

}

}

Таблица 1.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Имя изменено с на Ваше имя |

Анализ результатов:

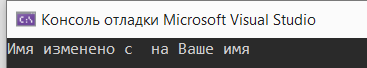


Рисунок 1.3 – Результат работы программы

Задание 4. Создать класс, в нем делегат и соответствующее ему событие. Создать два класса-наблюдателя, в методах которых будет описана их реакция на событие. Добавить 2 обработчика к событию из первого класса и один из второго. Вывести результат на экран. Удалить один обработчик события и вывести результат на экран.

Листинг программы:

using System;

public class EventClass

{

public delegate void MyDelegate();

public event MyDelegate MyEvent;

public void RaiseEvent()

{

MyEvent?.Invoke();

}

}

public class Observer1

{

public void OnEvent()

{

Console.WriteLine("Observer1: Событие произошло");

}

}

public class Observer2

{

public void OnEvent()

{

Console.WriteLine("Observer2: Событие произошло");

}

}

class Program

{

static void Main(string[] args)

{

EventClass eventClass = new EventClass();

Observer1 observer1 = new Observer1();

Observer2 observer2 = new Observer2();

eventClass.MyEvent += observer1.OnEvent;

eventClass.MyEvent += observer1.OnEvent;

eventClass.MyEvent += observer2.OnEvent;

Console.WriteLine("Вызов события с тремя обработчиками:");

eventClass.RaiseEvent();

eventClass.MyEvent -= observer1.OnEvent;

Console.WriteLine("\nВызов события с двумя обработчиками:");

eventClass.RaiseEvent();

}

}

Таблица 1.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Вызов события с тремя обработчиками:  Observer1: Событие произошло  Observer1: Событие произошло  Observer2: Событие произошло  Вызов события с двумя обработчиками:  Observer1: Событие произошло  Observer2: Событие произошло |

Анализ результатов:

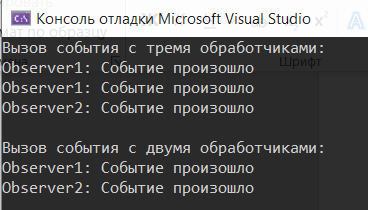


Рисунок 1.4 – Результат работы программы