**13 События**

Задание №1. Создать 3 метода для подсчета значений трех разных функций. Создать делегат. В программе вызывать все три метода при помощи делегата.

Листинг программы:

Operation operation = ExpOne;

double result = operation(2);

Console.WriteLine(result);

operation = ExpTwo;

result = operation(2);

Console.WriteLine(result);

operation = ExpThree;

result = operation(2);

Console.WriteLine(result);

Display(operation);

double ExpOne(int x) => Math.Pow(x, 2);

double ExpTwo(int x) => Math.Pow(x, 3);

double ExpThree(int x) => Math.Pow(x, 4);

void Display(Operation expResult) => Console.WriteLine($"Result: {expResult(2)}");

delegate double Operation(int x);

Таблица 13.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 4, 5, 3 | 16, 125, 81 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.1.

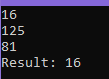


Рисунок 13.1 – Результат работы программы

Источник: собственная разработка

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

Листинг программы:

void Display(Operation expResult) => Console.WriteLine($"Result: {expResult(2)}");

Таблица 13.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Delegate | expResult |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.2.

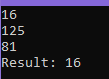


Рисунок 13.2 – Результат работы программы

Источник: собственная разработка

Задание №3. Для класса MyInfo содержащего поле name (Ваше имя) создать событие выводящее оповещение в случае изменения значения поля name.

Листинг программы:

class MyInfo

{

public delegate void MyEventHandler(string message);

public event MyEventHandler ChangeEvent = delegate { };

string name = "Aleksey";

public string SetValue

{

get

{

return name;

}

set

{

name = value;

ChangeEvent($"Name has been changed: {name}");

}

}

}

static void Main(string[] args)

{

MyInfo obj = new MyInfo();

obj.ChangeEvent += x => Console.WriteLine(x);

obj.SetValue = "Mike";

obj.SetValue = "Jake";

obj.SetValue = "Finn";

}

Таблица 13.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Aleksey | ChangeEvent |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.3.

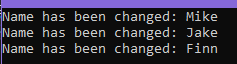


Рисунок 13.3 – Результат работы программы

Источник: собственная разработка

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:

try

{

Console.Write("Enter name: ");

string name = Console.ReadLine();

MyInfo inf = new MyInfo();

A objA = new A();

B objB = new B();

inf.Detect += inf.Display;

inf.Detect += objA.ObserverA;

inf.Detect += objB.ObserverB;

inf.NameChange(name);

Console.WriteLine("Deleted observer result: ");

inf.Detect -= objB.ObserverB;

inf.NameChange(name);

}

catch (Exception e)

{

Console.WriteLine($"An error occured! {e.Message}");

}

delegate void Name(string message);

class MyInfo

{

public string name = "Aleksey";

public event Name? Detect;

public void NameChange(string name)

{

this.name = name;

Detect?.Invoke($"Name has been changed! Current name: {name}");

}

public void Display(string message) => Console.WriteLine(message);

}

class A

{

public void ObserverA(string name) => Console.WriteLine($"{name}: observe A");

}

class B

{

public void ObserverB(string name) => Console.WriteLine($"{name}: observe B");

}

Таблица 13.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Jake | ObserverA, ObserverB |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.4.

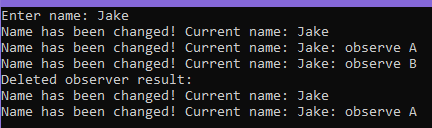


Рисунок 13.4 – Результат работы программы

Источник: собственная разработка