События

	Вопросы
1.	Может ли событие содержать ссылки на несколько методов?
2.	Выберите верные утверждения: 1) событие - тип 2) событие можно объявить в блоке исполняемого кода 3) событие всегда является членом класса, либо членом структуры 4) событие хранит ссылки на методы, как и многоадресный делегат 5) событие не может быть статическим
3.	Модификаторами события могут быть(выберете верные варианты) 1) static 2) new 3) abstract 4) virtual 5) readonly
4.	Какие модификаторы видимости могут быть применены для событий?
5.	Как подписаться на событие (методы, которые должны запуститься) 1. <КлассИлиОбъект>.<ИмяСобытия> ()+= <КлассЧейМетодДолженЗапуститься>.<МетодПодходящийПоСигнатуре>() 2. <ИмяСобытия> += <КлассЧейМетодДолженЗапуститься>.<МетодПодходящийПоСигнатуре>() 3. <ИмяСобытия> += <КлассЧейМетодДолженЗапуститься>.<МетодПодходящийПоСигнатуре> 4. <КлассИлиОбъект>.<ИмяСобытия> += <КлассЧейМетодДолженЗапуститься>.<МетодПодходящийПоСигнатуре> 5. <КлассИлиОбъект>.<МетодПодходящийПоСигнатуре>+= <ИмяСобытия>

6.	Что выведет программа?
	
	delegate void MyEvent1(int n);
	class MyEvent
	{
	public event MyEvent1 Event1;
	
	public void OnEvent1(int n)
	{
	if (Event1 != null)
	Event1(n);
	}
	}
	class Program
	{
	static void Main()
	{
	MyEvent ev =new MyEvent();
	ev.Event1 += (n) =>
	Console.Write(n++ * 8);
	ev.OnEvent1(3);
	ev.OnEvent1(2);
	}
	}

7.	Встроенный делегат, обрабатывающий событие: 1)public delegate void EventHandler(object sender) 2)public delegate void EventHandler(EventArgs e) 3)public delegate void EventHandler(object sender, EventArgs e) 4)public delegate void EventHandler() 5)C# не имеет встроенных делегатов для этого

8.	Выберите верные утверждения: 1)Для событий может использоваться модификатор области видимости internal 2)События могут реализовываться при помощи модификатора abstract 3)События НЕ могут быть членом класса static 4)События НЕ могут быть помечены модификатором sealed

9.	Что будет выведено на экран после выполнения данного кода? (***
	- ошибка компиляции)
	namespace ImSorry
	{
	class Program
	{
	delegate void del(int x);
	private static event del growth = p_1;
	
	private static void p_1(int y)
	{
	growth += (int z) => { Console.Write((int)z*z); };
	}
	
	static void Main(string[] args)
	{
	for (int i = 0; i <= 5; i++)
	{
	if (i % 2 == 1)
	{
	growth(i);
	}
	}
	}
	}
	}

10.	- Что выведет следующая программа?
	
	using System;
	
	namespace Task-1
	{
	class MyEventArgs : EventArgs
	{
	public MyEventArgs() : base() { }
	}
	
	class Program
	{
	static event EventHandler OnEvent;
	
	static void Main(string[] args)
	{
	Program pr = new Program();
	OnEvent += MyMethod;
	OnEvent(pr, EventArgs.Empty);
	}
	
	static void MyMethod(object sender, MyEventArgs args)
	{
	Console.WriteLine("This is my method!");
	}
	}
	}

	1) This is my method! 2) Пустую строку 3) Ничего - ошибка компиляции
11.	Выберите верные утверждения: 1) Методы экземпляра и статические методы могут быть использованы в качестве обработчиков событий. 2) Когда в качестве обработчика используется метод экземпляра, то события адресуются конкретным экземплярам объектов. 3) События не могут быть вызваны в теле метода. 4) Как и делегаты, события поддерживают групповую адресацию.

12.	Как должны выглядеть .NET Framework совместимые обработчики событий, установленные корпорацией Microsoft:
	a) protected event обработчик (int a) { //.. }
	b) public event обработчик (var a) { //.. }
	c) event обработчик (object отправитель, EventArgs e) { //.. }
	d) event обработчик (ref a, ref b) { //.. }

13.	Что будет выведено в результате выполнения данного кода (*** - ошибка компиляции)
	using System;
	
	namespace Application
	{
	public class Mass
	{
	public delegate void Del(ref double x);
	public event Del Inventor2;
	public static void Mas2(ref double x)
	{
	x = 100;
	
	}
	public static void Mas3(ref double x)
	{
	double sqrt = 0;
	sqrt = Math.Sqrt(x);
	Console.WriteLine("Sqrt = " + sqrt);
	}
	}
	class MainClass
	{
	public static void Print(double x)
	{
	Console.WriteLine(x);
	}
	public static void Main(string[] args)
	{
	Random gen = new Random();
	double x = 0;
	Mass a = new Mass();
	a.Inventor2 += Mass.Mas2;
	a.Inventor2 += Mass.Mas3;
	a.Inventor2(ref x);
	Print(x);
	}
	}
	}

14.	Выберите неправильные ответы: 1. Событие может быть декларировано в перечислении 2. Событие может быть деклалировано с модификатором abstract 3. Событие может быть декларировано с модификатором sealed

	4. Событие может быть декларировано в классе 5. Событие может быть декларировано с модификатором readonly

15.	В результате выполнения следующей программы:
	delegate void RunHandler(ref byte d);
	
	class Program
	{
	static event RunHandler onRun;
	
	static void Main()
	{
	onRun += new RunHandler(one);
	onRun += new RunHandler(two);
	byte z = 7;
	while (z++ < 9)
	{
	onRun(ref z);
	Console.Write(z);
	}
	}
	
	static void one(ref byte x)
	{
	Console.Write(x += 2);
	}
	
	static void two(ref byte y)
	{
	Console.Write(y += 1);
	}
	}
	на экран будет выведено:

16.	Какие из данных ключевых слов могут быть применены к событию? 1) sealed 2) extern 3) abstract 4) sealed 5) volatile 6) virtual

17.	Что выведет данный код:
	class FirstHandler{
	public void Message(){
	Console.WriteLine("I'm here!");
	}
	}
	
	class SecondHandler{
	public void Message(){
	Console.WriteLine("And there!");
	}
	}
	
	class Program{
	static void Main(){
	delegate void WhereAreYou();
	event WhereAreYou callMe;
	FirstHandler first = new FirstHandler();
	SecondHandler second = new SecondHandler();
	callMe += first.Message;
	callMe += second.Message;
	callMe();

	} }

18.	Что выведет предоставленный кусок кода?
	Если возникнет ошибка компиляции, напишите: ***
	Если ошибок и исключений нет, но на экран не выведется ничего, напишите: —-
	Если возникнет ошибка исполнения или исключение, напишите: +++
	
	public delegate void Delegate(string s);
	public static event Delegate Event;
	public static void ConsoleWorker(string message)
	{
	Console.WriteLine(message);
	}
	static void Main(string[] args)
	{
	Delegate del = message => Console.WriteLine(message);
	Event("Hello");
	}

19.	Выберете верные утверждения 1)События могут быть объявлены в классе или в пространстве имен 2)События не запускаются извне классов, в которых они объявлены 3) Событие может быть локальной переменной метода 4) Ключевое слово event используется для объявления события 5) Для подписки метода на событие используется оператор +=
20.	Может ли событие объявлено с модификатором abstract/override?

21.	Что выведет предоставленный кусок кода?
	Если возникнет ошибка компиляции, напишите: ***
	Если ошибок и исключений нет, но на экран не выведется ничего, напишите: ---
	Если возникнет ошибка исполнения или исключение, напишите: +++
	class Counter
	{
	public delegate string Delegate();
	public event Delegate onCount;
	public void Count()
	{
	for (int i = 0; i < 100; i++)
	{
	if (i == 71)
	{
	onCount();
	}
	}
	}
	}
	class A
	{
	public void Message()
	{
	Console.WriteLine("Пора действовать, ведь уже 71!");
	}
	}
	class B
	{
	public void Message()
	{
	Console.WriteLine("Точно, уже 71!");
	}

	}
	class Program
	{
	static void Main(string[] args)
	{
	Counter Counter = new Counter();
	A Handler1 = new A();
	B Handler2 = new B();
	
	Counter.onCount += Handler1.Message;
	Counter.onCount += Handler2.Message;
	
	Counter.Count();
	}
	}

22.	Выберите верные утверждения: 1) Если к событию применить ключевое слово virtual, то это позволит производным классам переопределять поведение события при помощи ключевого слова override 2) Нельзя к событию применить ключевое слово static 3) События поддерживают только операторы += и -= 4) Аксесорные формы событий могут быть абстрактными
23.	event допускает модификаторы: 1)sealed 2)virtual 3)abstract 4)static 5)public

24.	Что выведется на экран, после выполнения программы:
	*** - Ошибка компиляции
	** - Необработанное исключение
	* - Ничего не выведется
	namespace ConsoleApp9
	{
	class A
	{
	public event EventHandler OnSomething;
	public void Test()
	{
	for (int i = 0; i < 10; i++)
	{
	if (i == 1)
	{
	OnSomething();
	}
	}
	}
	}
	public delegate void EventHandler();
	class B
	{
	
	public void Event()
	{
	Console.WriteLine("1");
	}
	}
	class Program
	{
	static void Main(string[] args)
	{

	A a = new A();
	B b = new B();
	a.OnSomething += b.Event;
	
	}
	}
	}

25.	Что из перечисленного может быть назначено обработчиком события? 1. Автоматически реализуемые свойства 2. Делегат (объект) 3. Любая функция 4. Статический метод с сигнатурой делегата, от которого он образован
26.	- Какие параметры может иметь функция, совместимая с делегатом для событий EventHandler? Выберите все возможные ответы 1) (object sender, EventArgs args) 2) (int sender, EventArgs args) 3) (int sender, myEventArgs args), где MyEventArgs - производный класс от EventArgs
27.	Верно ли данное утверждение: События представляют собой специальный вид многоадресного делегата, который можно вызвать только из класса или структуры, в которых он объявлен. 1) Да 2) Нет
28.	Для чего нужны аксессоры событий? a) не нужны b) для инициализации обработчиков событий c) для управления списком обработчиков d) для слежения за обработчиками событий, при групповой адресации

29.	Что будет выведено в результате выполнения данного кода (*** - ошибка компиляции)
	using System;
	namespace Application
	{
	delegate void Del();
	public class A
	{
	public event Del ev;
	public void Creator()
	{
	ev();
	}
	}
	class MainClass
	{
	public static void Pr()
	{
	Console.WriteLine("Hi");
	}
	public static void Main(string[] args)
	{
	A a = new A();
	a.ev += Pr;
	a.Creator();
	}
	}
	}

30.	-
31.	Что делает данная программа? public int counter = 0;

	public Form1()
	{
	InitializeComponent();
	this.MouseEnter = Form1_MouseEnter;
	}
	
	private void Form1_MouseEnter(object sender, EventArgs e)
	{
	this.ourLabel.Text = $"Mouse entered {counter++} times";
	}
	Примечание:
	Если возникнет ошибка компиляции, введите: ***
	Если возникнет ошибка исполнения или исключение, введите: +++

32.	Выберете аксессоры событий: 1) get 2) add 3) remove 4) set 5) addRange 6) update

	Ответы
1.	Да
2.	34
3.	1234
4.	public, protected, private, internal
5.	4
6.	2416
7.	3
8.	1,2
9.	92525
10.	3
11.	124
12.	c) event обработчик (object отправитель, EventArgs e) { //.. }
13.	***
14.	1, 5
15.	101111
16.	1) 3) 4) 6)
17.	I'm here! And there!
18.	+++

19.	245
20.	Да
21.	***
22.	13
23.	12345
24.	* (Ничего не выведется)
25.	24
26.	12
27.	1
28.	c) для управления списком обработчиков
29.	***
30.	-
31.	***
32.	2) 3)