**1.Назовите принципы ООП. Поясните их.**

Выделяют три самых главных из них: наследование, полиморфизм и инкапсуляция.

Инкапсуляция - это объединения данных и инструкций по их обработке в единую сущность - класс.

Наследование - это возможность одного класса использовать методы другого без повторения их фактической реализации.

Полиморфизм – это различная реализация однотипных действий.

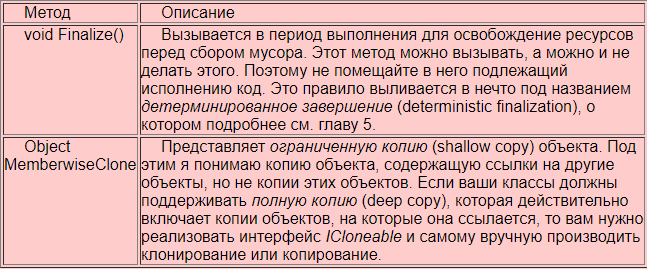
**2. Назовите класс .NET, от которого наследуются все классы.**

System.Object

**3. Охарактеризуйте открытые методы System.Object**

Класс System.Object, его методы Equals(Определяет, равен ли заданный объект текущему объекту.), ToString(Возвращает строковое представление текущего объекта.), GetHashCode(Служит хэш-функцией по умолчанию.), GetType(Возвращает объект [Type](https://msdn.microsoft.com/ru-ru/library/system.type(v=vs.110).aspx) для текущего экземпляра.)

**4.Закрытые методы S.O**



**7.Чем отличается объект от класса?**

Класс-это описание объекта, а объект представляет экземпляр этого класса.

**8.Что такое конструктор? Когда он вызывается?**

Конструктор - это специальный метод, с тем же именем, что и имя класса и без возвращаемого типа; Конструктор вызывается при создании класса и никогда не может быть вызван напрямую в коде.

**9.Перечислите свойства конструктора.**

Не имеет возвращаемого значения, имя такое же как и у класса , не наследуется, для класса без явного заданного конструктора компилятор создаёт по умолчанию.

**10.Что такое деструктор?**

метод, который будет вызываться непосредственно перед окончательным уничтожением объекта системой "сборки мусора". Такой метод называется деструктором и может использоваться в ряде особых случаев, чтобы гарантировать четкое окончание срока действия объекта.

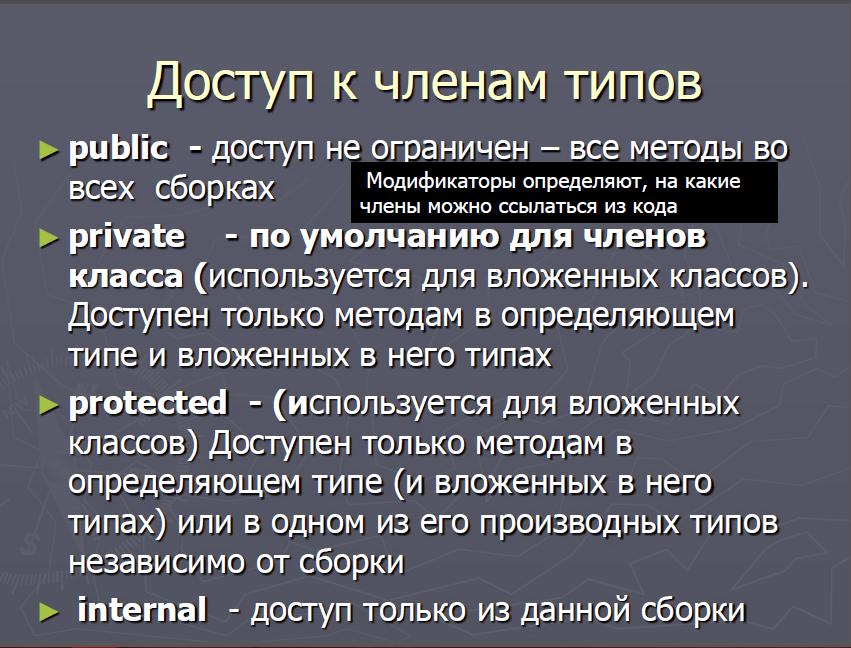
**11. Что такое this?**

This обеспечивает доступ к текущему экземпляру класса

12, 5 5

7 7

**13.Спецификаторы доступа**



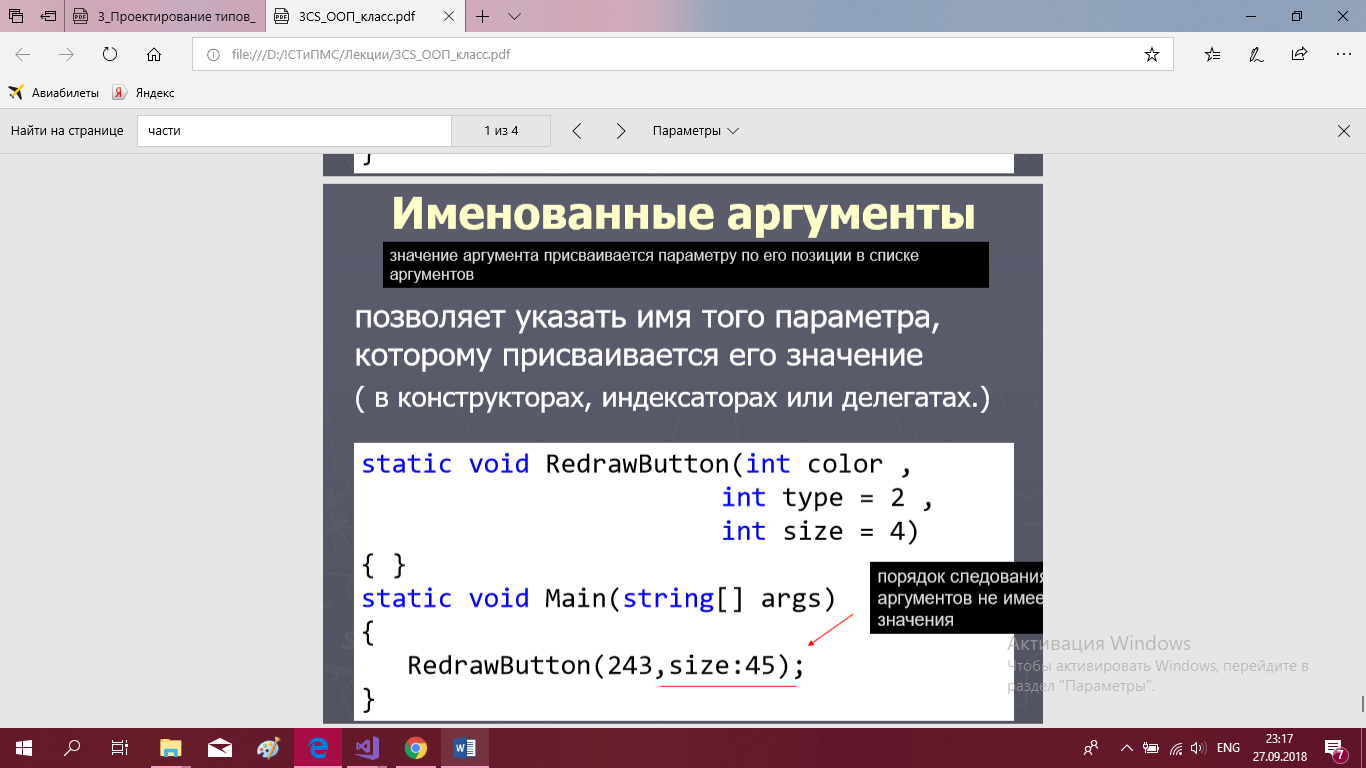
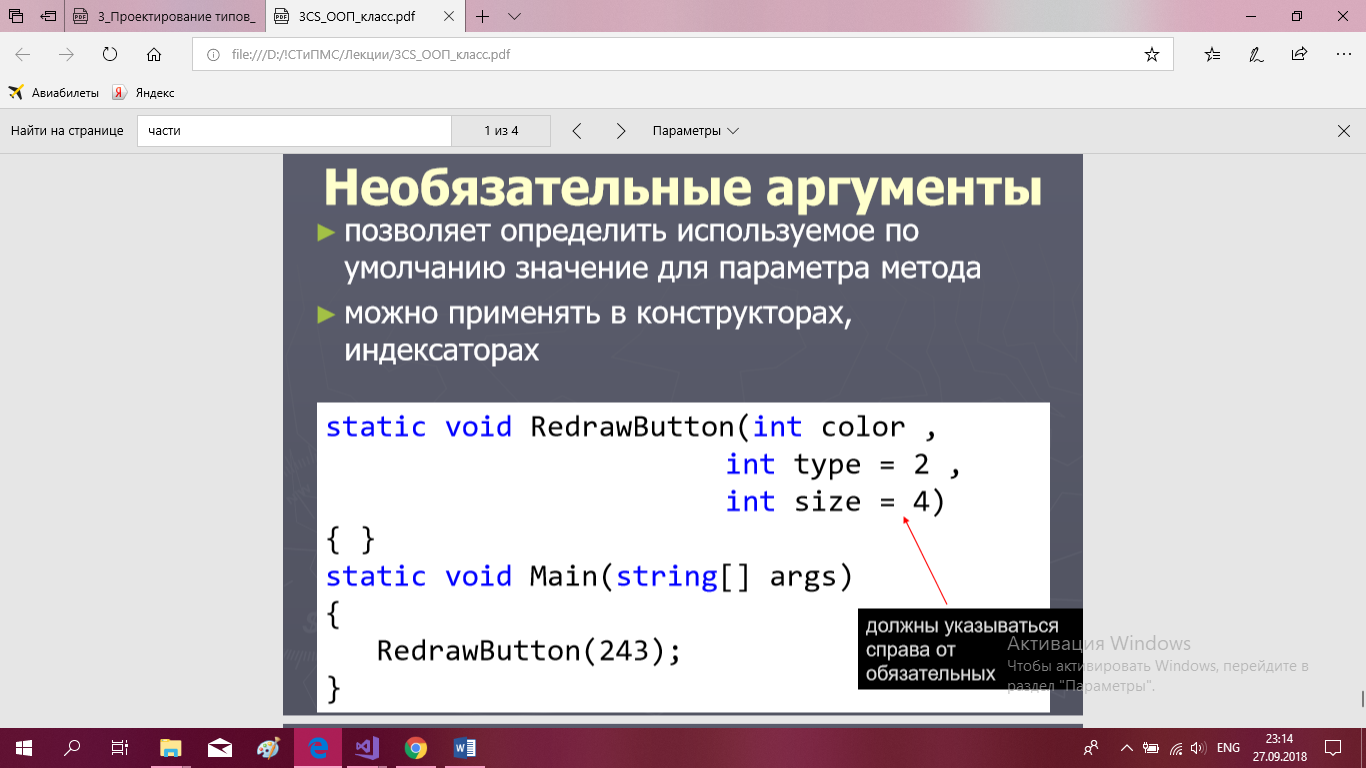
14.Protected internal

Доступ в любом месте программы и из классов-наследников

15.ref Out

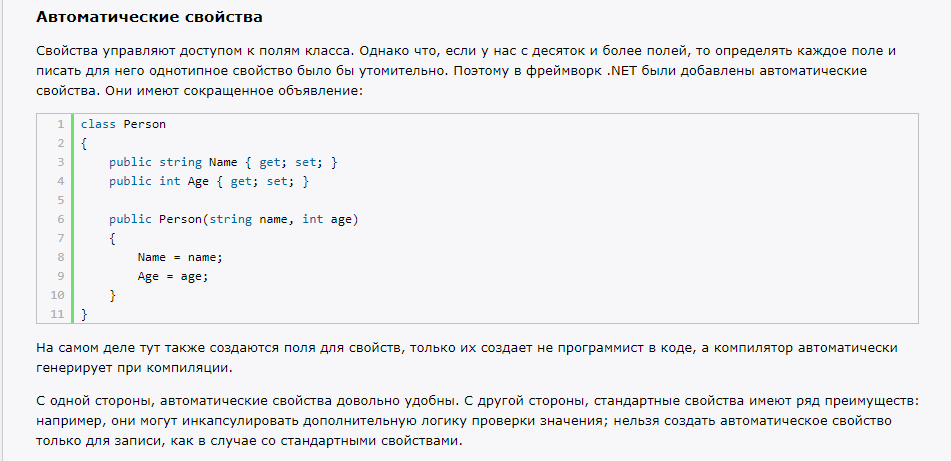
Разница в том, что out позволяет вам передать ссылку на переменную неинициализированную, а ref — только на инициализированную.

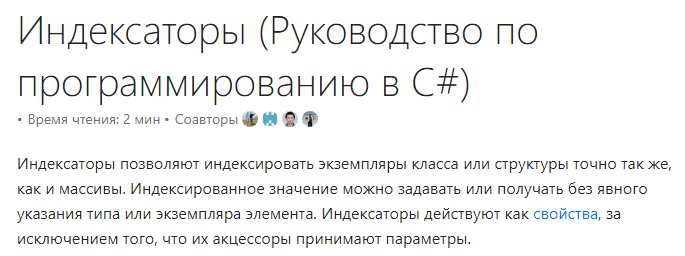
16.

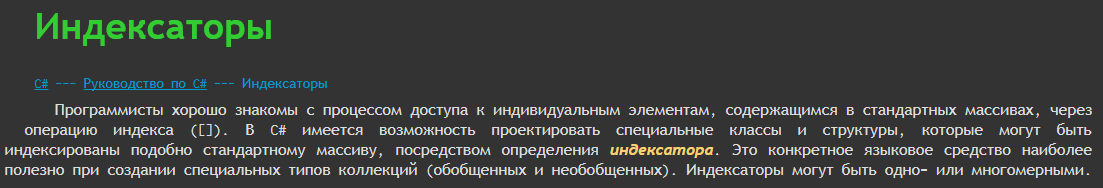


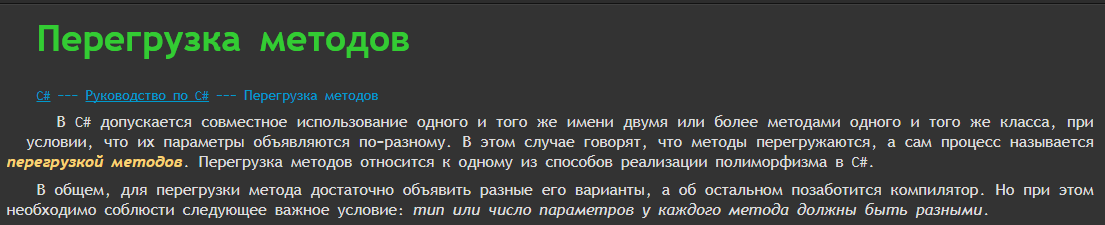
**19 Назовите явное имя параметра, передаваемого в метод set свойства класса?**

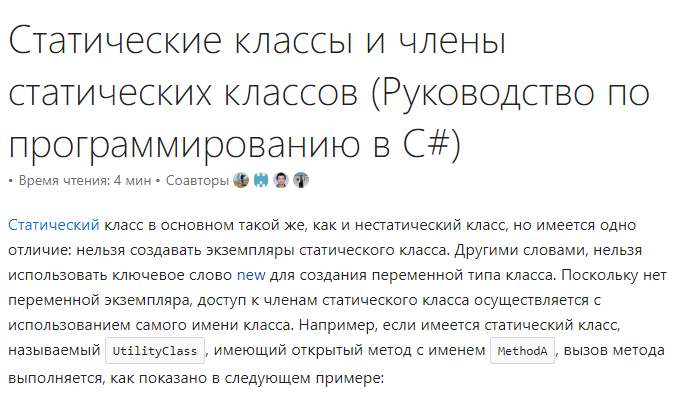
value. Тип этого параметра определяется типом свойства.

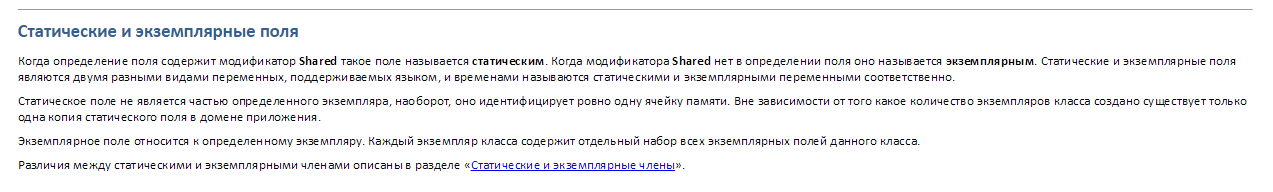
20.

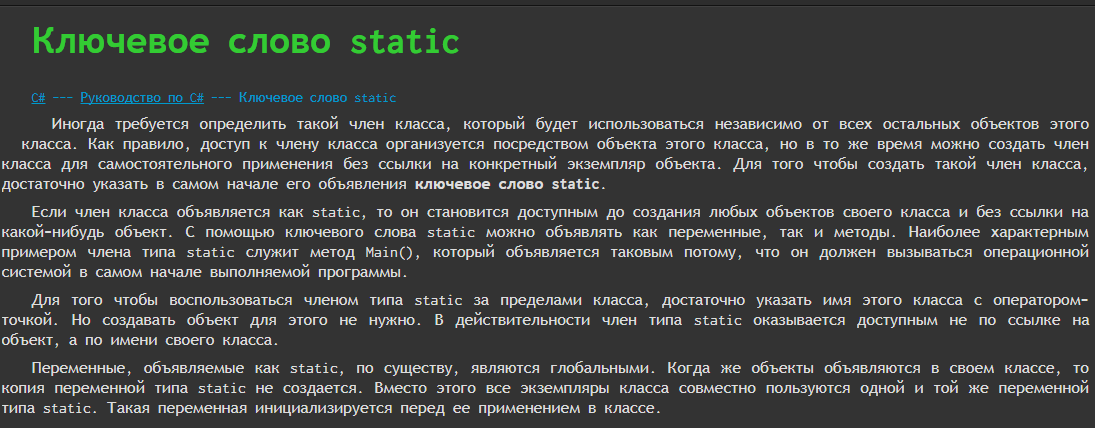
21. 



22. 

25. 

26. 

27. 

28.

Конструктор - это специальный метод, с тем же именем, что и имя класса и без возвращаемого типа; Конструктор вызывается при создании класса и никогда не может быть вызван напрямую в коде.

Если член класса объявляется как static, то он становится доступным до создания любых объектов своего класса и без ссылки на какой-нибудь объект. С помощью ключевого слова static можно объявлять как переменные, так и методы.

Еще одной разновидностью члена класса является свойство. В частности, свойство состоит из имени и аксессоров get и set. Аксессоры служат для получения и установки значения переменной. Главное преимущество свойства заключается в том, что его имя может быть использовано в выражениях и операторах присваивания аналогично имени обычной переменной, но в действительности при обращении к свойству по имени автоматически вызываются его аксессоры get и set.

Разница в том, что out позволяет вам передать ссылку на переменную неинициализированную, а ref — только на инициализированную.

Частичные классы (partial class) представляют возможность разделить функционал одного класса на несколько частей.

Метод — это блок кода, содержащий ряд инструкций. Программа инициирует выполнение инструкций, вызывая метод и указывая все аргументы, необходимые для этого метода.

28. При поверхностном копировании копируются значения полей класса, включая значения любых указателей или ссылок. При этом скопированные значения этих указателей и ссылок указывают на одни и те же объекты, что и в оригинальном объекте, что зачастую ведет к ошибкам. Отсюда и название такого метода копирования: мы копируем только указатели/ссылки, вместо того, чтобы делать копии этих внутренних объектов и ссылаться на них, собственно не углубляемся во внутреннюю структуру объекта. При глубоком копировании мы копируем значения полей не только на первом "уровне", но и заходим глубже, копируя все значения.

30. Классы могут быть частичными. То есть мы можем иметь несколько файлов с определением одного и того же класса, и при компиляции все эти определения будут скомпилированы в одно. Частичные классы могут содержать частичные методы. Таким методы также опреляются с ключевым словом **partial**. Причем определение частичного метода без тела метода находится в одном частичном классе, а реализация этого же метода - в другом частичном классе.

31.System.Int32

32.a=2;b=1

33.//2 имена членов не могут совпадать с именами типов, в которых они содержаться

35.Недоступен из-за его уровня защиты

36.A static

A

37. //3 Присваивание значений доступному только для чтения статическому полю допускается только в статическом конструкторе и в инициализаторе переменных.

23. В языке C# возможно разбиение определения класса, структуры или интерфейса между двумя или больше исходными файлами. Каждый исходный файл содержит свою часть определения класса и все такие части собираются во время компиляции.  
Есть несколько ситуаций, когда удачно разбить определение класса на несколько файлов:  
1. При работе над большим проектом, разбиение класса на несколько файлов позволяет нескольким программистам работать над ним одновременно.  
2. При работе с автоматически генерируемыми исходниками, код может быть добавлен в класс без необходимости пересоздавать исходник. Visual Studio использует этот подход при создании компонентов Windows Forms, веб-сервисов и т.д. Возможно создать код, который использует эти классы без необходимости редактировать файлы, которые создает Visual Studio.  
  
Для такого разбития класса используется модификатор **partial**.

24. Анонимные типы позволяют создать объект с некоторым набором свойств без определения класса. Анонимный тип определяется с помощью ключевого слова **var** и инициализатора объектов