1. Назовите принципы ООП. Поясните каждый из них.

- инкапсуляция (пакетирование) - механизм, связывающий вместе данные и код, обрабатывающий эти данные, и сохраняющий их от внешнего воздействия и ошибочного использования  
- наследование - процесс, благодаря которому один объект может наследовать (приобретать) свойства от другого объекта  
- полиморфизм - способность вызывать метод потомка через экземпляр предка  
- абстракция данных - разделение и независимое рассмотрение интерфейса и реализации

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

System.Object.

3. Охарактеризуйте открытые методы System.Object.

|  |  |
| --- | --- |
| bool Equals() | Сравнивает две ссылки на объекты в период выполнения, чтобы определить, указывают ли они в точности один и тот же объект. Если две переменные ссылаются на один и тот же объект, возвращается *true.*В случае размерных типов (см. о них следующий раздел) этот метод возвращает *true,*если типы переменных идентичны и их значения равны. |
| *int GetHashCodeO* | Возвращает заданный для объекта хзш-код. Хэш-функции используются в реализации класса, когда хэш-код объекта нужно поместить в хэш-таблицу для повышения производительности. |
| Type GetType() | Используется с методами отражения (см. о них главу 16) для получения информации о типе данного объекта. |
| string ToString | Используется по умолчанию для получения имени объекта. Его можно переопределить в производных классах, чтобы они возвращали понятное пользователю текстовое представление объекта. |

4. Охарактеризуйте закрытые методы System.Object.

|  |  |
| --- | --- |
| Метод | Описание |
| void Finalize() | Вызывается в период выполнения для освобождение ресурсов перед сбором мусора. Этот метод можно вызывать, а можно и не делать этого. Поэтому не помещайте в него подлежащий исполнению код. Это правило выливается в нечто под названием *детерминированное завершение*(deterministic finalization), о котором подробнее см. главу 5. |
| Object MemberwiseClone | Представляет *ограниченную копию*(shallow copy) объекта. Под этим я понимаю копию объекта, содержащую ссылки на другие объекты, но не копии этих объектов. Если ваши классы должны поддерживать *полную копию*(deep copy), которая действительно включает копии объектов, на которые она ссылается, то вам нужно реализовать интерфейс *ICloneable*и самому вручную производить клонирование или копирование. |

5. Приведите пример определения класса.

[атрибуты][спецификаторы] class имя класса [:предок]

{

Тело класса

}

6. Какие ключевые слова можно использовать при определении класса?

[new](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/new-modifier) и [override](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/override). Модификатор overrideрасширяет метод virtual базового класса, а модификатор newскрывает доступный метод базового класса.

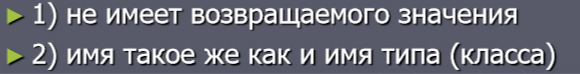
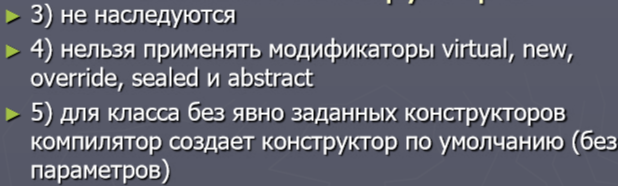
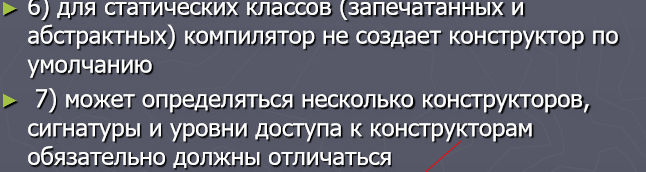
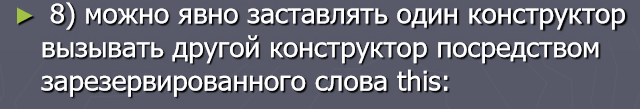
7. В чем отличие между объектом и классом?

**Класс –**тип данных, определяемый программистом, **объект –** переменная класса (Отличие состоит в том, что тип данных либо уже определен в языке, либо формально составляется из уже существующих (но без внутреннего программирования). Значит, класс можно определить как тип данных, определяемый программистом. Тогда объект – это переменная класса.)

Объект – это просто что-то конкретное, а класс – это некое абстрактное понятие, которое просто содержит описательную часть.

8. Что такое конструктор? Когда вызывается конструктор?

Конструктор – специальный метод, который позволяет корректно инициализировать экземпляр типа;  
он вызовется сразу после инициализации статических полей объекта, когда процесс выполнения достигает места, где объекты объявляются

9. Перечислите свойства конструктора?   
  
  
  


10. Что такое деструктор (destructor) ?

Противоположное конструктору, служат как уборщики мусора, не используются в шарпах, так как есть встроенный уборщик.

11. Что такое this?

This – ссылка, которая обеспечивает доступ к текущему экземпляру класса. Любой не статический метод автоматически содержит в качестве 1-ого скрытого параметра this.

12. Что будет выведено в результате?

5 5

7 7

13. Какие спецификаторы доступа для класса и методов класса существуют в C#?

1. Public – Доступ неограничен (все методы во всех сборка)

2. Private – по умолчанию (доступ только в текущем типа и вложенных в него)

3. Protected – доступ только в определяющем типе и во вложенных типах и производных от него;

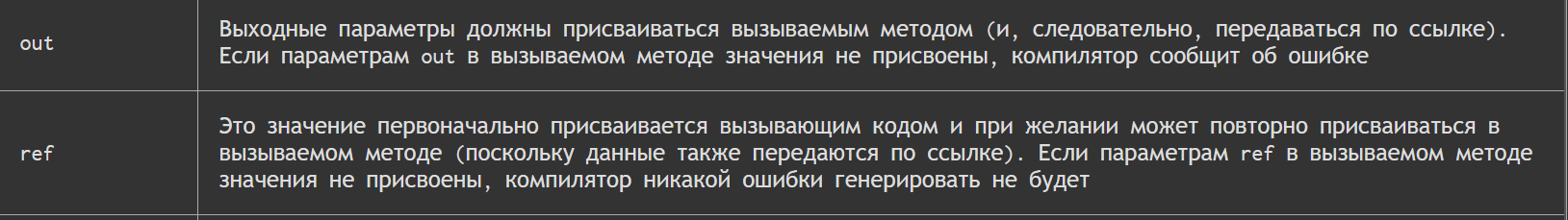
4. Internal – доступ только из текущей сборки.

14. Опишите модификатор protected internal.

совмещает функционал двух модификаторов. Классы и члены класса с таким модификатором доступны из текущей сборки и из производных классов.

15. Зачем и как используются ref и out параметры функции?

out - тот же ref, но с условием, что параметру в методе должно быть присвоено значение.

  
16. Приведите пример необязательных и именованных параметров метода.

Для необязательных параметров нам необходимо объявить значение по умолчанию, но можно опустить некоторые члены.  
Но мы можем нарушить подобный порядок, используя именованные параметры

17. Приведите пример полей класса – статические, константные, только для чтения.

**class** Foo

{

    SomeType x; *// статический*

Const g;

Readonly l;

}

18. Приведите пример определения свойств класса. Как свойства связаны с инкапсуляцией?

[атрибуты][спецификаторы] тип имя\_свойства

{

[get code]

[set code ]

}

Связаны тем, что могут иметь модификаторы доступа.

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

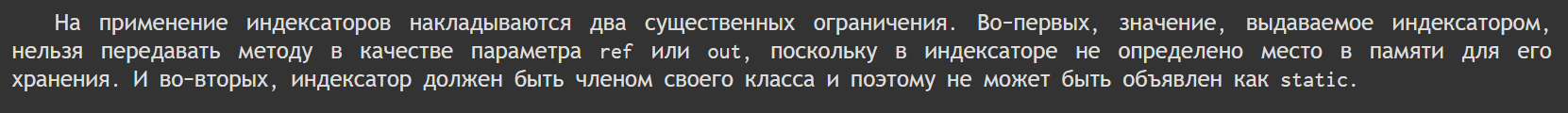
20. Что такое автоматические свойства?

Тип имя {get; set}

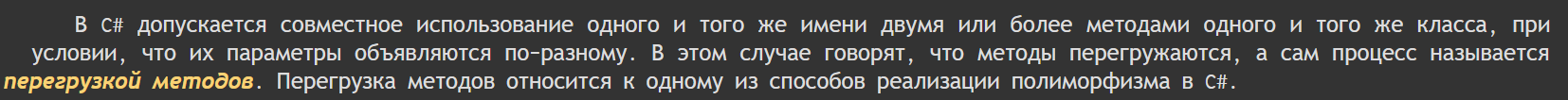
Компилятор автоматически реализует методы и генерирует поля для свойств;

21. Что такое индексаторы класса? Какие ограничения существуют для индексатора?

**Индексаторы** позволяют индексировать объекты и использовать их как массивы.

Ограничение в методах доступа;  


22. Что такое перегруженный метод?

Несколько методов вызвано;  


23. Что такое partial класс и какие его преимущества?

Ключевое слово **partial** указывает на то, что другие части класса, структуры или интерфейса могут быть определены в пространстве имен.

24. Что такое анонимный тип в C#?

Анонимные типы позволяют создать объект с некоторым набором свойств без определения класса

25. Для чего делают статические классы?

роль пространства имен для глобальных переменных

26. В чем отличие статического поля от экземплярного?

Статический метод - вызывается на уровне типа (вызов: Имя\_типа точка Имя\_метода)  
Экземплярный метод - вызывается на уровне объявленного экземпляра типа (вызов: Имя\_переменной точка Имя\_метода).

27. Поясните работу статических конструкторов.

Иногда требуется определить такой член класса, который будет использоваться независимо от всех остальных объектов этого класса

28. Какая разница между поверхностным (shallow) и глубоким (deep) копированием?

При поверхностном копировании копируются значения полей класса, включая значения любых указателей или ссылок.

При глубоком копировании мы копируем значения полей не только на первом "уровне", но и заходим глубже, копируя все значения.

29. В чем разница между равенством и тождеством объектов?

Равенство при определенных условиях; тождество при любых.

30. Что такое частичные классы и частичные методы?

Частичные классы (**partial class**) представляют возможность разделить функционал одного класса на несколько файлов

31. Что будет выведено на консоль результате выполнения следующего кода:

Тип переменной age;

32. Что будет выведено на консоль результате выполнения следующего кода:

2 1

33. Пусть задан следующий класс. Какой из конструкторов задан неверно?

3 так как не должен иметь возвращаемого значения

34. Пусть задан следующий класс. Сколько аргументов может быть задано при вызове конструктора данного класса?

2

35. Почему не удается создать объект класса A?

2

36. Что будет выведено в консоль при выполнении данной программы?

37. Какая строка приведенного далее класса вызовет ошибку компиляции?

public Points ведь нету параметров!!!