1. Для чего используют статические классы?

В основном используются как контейнеры для методов. Экземпляры статического класса создавать нельзя. Не могут содержать конструкторы и являются запечатанным классом – следовательно, от него нельзя наследовать.

2. Что может содержать статический класс?

Ответ в первом.

3. Что такое производный и базовый классы?

Базовый класс – это который уже существует, а производный – тот, который может наследовать его свойства и методы.

4. Как используют ключевое слово base?

Используется для доступа к членам базового класса. Может быть использован в конструкторе или методе экземпляра.

5. В чем заключена основная задача наследования?

Один класс может наследовать функциональность другого.

6. Пусть базовый класс содержит метод basefunc(), а производный класс не имеет метода с таким именем. Может ли объект производного класса иметь доступ к методу basefunc()? Если да, то при каких условиях?

Может, с использованием ключевого слова base.

7. Напишите объявление конструктора без аргументов для производного класса B, который будет вызывать конструктор без аргументов базового класса A.

public Constructor() : base()

8. Что такое полиморфизм? Приведите пример.

Простыми словами, полиморфизм – это различная реализация однотипных действий. Хороший пример с рисованием фигуры – есть метод Draw() и фигуры квадрат, круг и треугольник. Конечно, они будут рисоваться по-разному – вот и один метод и разная реализация.

9. Определите назначение виртуальных функций.

Виртуальный метод может быть переопределён в классе-наследнике, может иметь своё тело в отличие от абстрактного, который должен быть переопределён в классе-наследнике и не может иметь своего тела.

10. Кому доступны переменные с модификатором protected?

Любому классу-наследнику.

11. Наследуются ли переменные с модификатором private?

Наследуются, но не являются доступными.

12. As, is – что это, как применяется? В чем между ними отличие?

Ключевое слово is проверяет принадлежит ли объект к проверяемому типу, возвращает true или false. Ключевое слово as – служит для приведения объекта к нужному типу.

13. Поддерживает ли C# множественное наследование?

Поддерживает множественное наследование от класса и нескольких интерфейсов, но не поддерживает наследование сразу от нескольких классов.

14. Можно ли запретить наследование от класса?

Да, с помощью модификатора доступа sealed.

15. Можно ли разрешить наследование класса, но запретить перекрытие метода?

Да, просто методу прописать sealed.

16. Что такое абстрактный класс?

Это класс, объект которого не может быть создан. Чисто как красивый чертежик без какой-либо реализации, все методы абстрактного метода должны быть переопределены в классе-наследнике.

17. В каком случае вы обязаны объявить класс абстрактным?

Если класс уже наследует от абстрактного класса или имеет хотя бы один абстрактный метод.

18. В чем разница между абстрактными и виртуальными классами? Между

виртуальными и абстрактными методами?

Ответ в девятом.

19. Какие компоненты класса могут быть виртуальными?

Методы, свойства, индексаторы, события.

20. Что такое интерфейс?

Интерфейс определяет набор свойств, методов и событий, но в отличие от классов не содержат их реализацию.

21. Что может содержать интерфейс?

Методы, свойства, индексаторы, события и статические поля и константы начиная с восьмой версии.

22. Как работать с объектом через унаследованный интерфейс?

Написать реализация методов и свойств и работать как с обычным классом.

23. Приведите пример явной реализации интерфейса.

24. Почему нельзя указать модификатор видимости для методов интерфейса?

А в чём тогда будет суть интерфейса?

25. Можно ли наследовать от нескольких интерфейсов?

Можно

26. Назовите отличия между интерфейсом и абстрактным классом.

Разница в реализации методов и модификаторах доступа, которые там и там прописаны. В интерфейсе у методов нет реализации, у абстрактного класса – может быть. В интерфейсе все методы и свойства имеют модификатор доступа public по умолчанию, в абстрактном классе это можно настроить.

27. Для чего используются стандартные интерфейсы ICloneable, IComparable, IComparer, lEnumerable?

– позволяет предоставить настраиваемую реализацию для создания копии объекта

– имеет один метод, передаваемым параметром которого является объект, с которым сравнивается текущий

– предоставляет метод сравнивающий два объекта

– возвращает ссылку на интерфейс, который имеет метод, поддерживающий перебор элементов

28. Во втором