### 1. Для чего используют статические классы?

Статические классы используются для группировки методов и свойств, которые не требуют создания экземпляра класса. Они могут содержать только статические члены и обычно применяются для утилит и вспомогательных функций.

### 2. Что может содержать статический класс?

Статический класс может содержать только статические методы, статические свойства, статические поля и статические события. Он не может содержать экземплярные члены.

### 3. Что такое производный и базовый классы?

Базовый класс — это класс, от которого наследуются другие классы. Производный класс — это класс, который наследует члены базового класса и может добавлять свои собственные члены.

### 4. Как используют ключевое слово `base`?

Ключевое слово `base` используется для обращения к членам базового класса, например, для вызова конструктора базового класса или доступа к его методам и свойствам.

### 5. В чем заключена основная задача наследования?

Основная задача наследования — это повторное использование кода и создание иерархии классов, что позволяет организовать код более структурированно и поддерживать его.

### 6. Может ли объект производного класса иметь доступ к методу `basefunc()` из базового класса, если производный класс не имеет метода с таким именем?

Да, объект производного класса может иметь доступ к методу `basefunc()`, если этот метод является публичным или защищенным (protected) и находится в базовом классе.

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

```csharp

class A

{

public A() { }

}

class B : A

{

public B() : base() { }

}

```

### 8. Что такое полиморфизм? Приведите пример.

Полиморфизм — это способность объектов разных классов обрабатываться одинаково через общий интерфейс. Например, методы `Draw()` могут быть реализованы в классах `Circle` и `Square`, но вызывать их можно через ссылку на базовый класс `Shape`.

### 9. Определите назначение виртуальных функций.

Виртуальные функции позволяют производным классам переопределять поведение методов базового класса, что обеспечивает возможность динамического связывания и полиморфизма.

### 10. Кому доступны переменные с модификатором `protected`?

Переменные с модификатором `protected` доступны в самом классе, а также в производных классах.

### 11. Наследуются ли переменные с модификатором `private`?

Нет, переменные с модификатором `private` не наследуются в производных классах.

### 12. `as`, `is` – что это, как применяется? В чем между ними отличие?

- `is` проверяет, является ли объект экземпляром указанного типа.

- `as` пытается привести объект к указанному типу и возвращает `null`, если приведение невозможно.

Пример:

```csharp

if (obj is MyClass) { /\* ... \*/ }

MyClass myObj = obj as MyClass;

```

### 13. Поддерживает ли C# множественное наследование?

Нет, C# не поддерживает множественное наследование для классов, но поддерживает его для интерфейсов.

### 14. Можно ли запретить наследование от класса?

Да, для этого используется модификатор `sealed`.

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

Да, для этого используется модификатор `sealed` в определении виртуального метода.

### 16. Что такое абстрактный класс?

Абстрактный класс — это класс, который не может быть инстанцирован и может содержать абстрактные методы, которые должны быть реализованы в производных классах.

### 17. В каком случае вы обязаны объявить класс абстрактным?

Если класс содержит хотя бы один абстрактный метод, он должен быть объявлен как абстрактный.

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

- \*\*Абстрактные классы\*\*: не могут быть инстанцированы и могут содержать абстрактные и обычные методы.

- \*\*Виртуальные методы\*\*: могут иметь реализацию и могут быть переопределены в производных классах.

### 19. Какие компоненты класса могут быть виртуальными?

Методы и свойства могут быть виртуальными.

### 20. Что такое интерфейс?

Интерфейс — это контракт, который определяет набор методов и свойств, которые класс должен реализовать.

### 21. Что может содержать интерфейс?

Интерфейс может содержать объявления методов, свойств, событий и индексаторов, но не может содержать реализации.

### 22. Как работать с объектом через унаследованный интерфейс?

Объект можно привести к интерфейсу, чтобы вызывать его методы. Например:

```csharp

IExample obj = new Example();

obj.Method();

```

### 23. Приведите пример явной реализации интерфейса.

```csharp

public interface IExample

{

void Method();

}

public class Example : IExample

{

void IExample.Method()

{

Console.WriteLine("Explicit Interface Implementation");

}

}

```

### 24. Почему нельзя указать модификатор видимости для методов интерфейса?

Методы интерфейса по умолчанию являются публичными и не могут иметь других модификаторов видимости.

### 25. Можно ли наследовать от нескольких интерфейсов?

Да, класс может реализовывать несколько интерфейсов.

### 26. Назовите отличия между интерфейсом и абстрактным классом.

- Интерфейс не может содержать реализацию, а абстрактный класс может.

- Класс может наследовать только от одного абстрактного класса, но может реализовать несколько интерфейсов.

### 27. Для чего используются стандартные интерфейсы `ICloneable`, `IComparable`, `IComparer`, `IEnumerable`?

- `ICloneable`: для создания копий объектов.

- `IComparable`: для определения порядка объектов.

- `IComparer`: для сравнения двух объектов.

- `IEnumerable`: для итерации по коллекции.

### 28. В какой строке приведенного ниже фрагмента листинга не содержится ошибки?

```csharp

public virtual void g() { } //2

```

### 29. Что будет выведено на консоль в результате выполнения следующего фрагмента?

```plaintext

3 4

```

### 30. Что будет выведено на консоль в результате выполнения следующего фрагмента кода?

```plaintext

A

B

```

### 31. Чем может быть `M4`, если дано следующее определение?

1) M4 - только интерфейс.

### 32. Выберите верное присваивание для объектов, определенных в листинге.

5) `c = b;`

### 33. Что будет выведено на консоль в результате выполнения следующего фрагмента, если раскомментировать строчку 1?

1) `B`

### 34. В какой строке приведенного ниже фрагмента листинга содержится ошибка?

3) `public string m() { return "C"; }//3` (переопределение метода не корректно).

### 35. Почему приведенный ниже фрагмент листинга содержит ошибку?

Нельзя создать экземпляр абстрактного класса.

### 36. В какой строке может быть ошибка компиляции?

4) `one = two;` (нельзя присвоить экземпляр разных производных классов).

### 37. Что будет выведено на консоль в результате выполнения следующего фрагмента листинга?

```

new G

```

Если у вас есть дополнительные вопросы или нужна более подробная информация по какой-либо теме, дайте знать!