1-2)

3) В языке C# класс, который наследуется, называется базовым, а класс, который наследует, — производным. Следовательно, производный класс представляет собой специализированный вариант базового класса. Он наследует все переменные, методы, свойства и индексаторы, определяемые в базовом классе, добавляя к ним свои собственные элементы.

4) Можно использовать ключевое слово base для вызова базовой реализации из любого метода. Это связывает вызов метода непосредственно с базовой реализацией, а это означает, что даже если новые дочерние классы переопределяют виртуальный метод, базовая реализация все равно будет вызвана, поэтому это нужно использовать с осторожностью.

5) Наследование (inheritance) является одним из ключевых моментов ООП. Благодаря наследованию один класс может унаследовать функциональность другого класса.

6)Может, причём при любых условиях

7)----

8) Полиморфизм - это способность объекта использовать методы производного класса, который не существует на момент создания базового.

9) При наследовании класса. Данный метод м.б. переопределен в производных классах с помощью ключевого слова override.

10) Любому классу-наследнику

11) Да, но они не являются доступными

12) Во-первых, можно использовать ключевое слово **as**. С помощью него программа пытается преобразовать выражение к определенному типу, при этом не выбрасывает исключение. В случае неудачного преобразования выражение будет содержать значение null. Второй способ заключается в проверке допустимости преобразования с помощью ключевого слова **is**:

|  |  |
| --- | --- |
| 1 | значение is тип |

Если значение слева от оператора представляет тип, указаный справа от оператора, то оператор is возвращает true, иначе возвращается false.

13) C# не поддерживает множественное наследование классов, но вам разрешено наследовать/реализовывать любое количество интерфейсов.

14) С помощью ключевого слова "sealed" можно запретить создавать наследников от класса.

15) *Да. Указываем класс как public, а метод как sealed.*

16) Абстрактный класс в объектно-ориентированном программировании — базовый класс, который не предполагает создания экземпляров. Абстрактные классы реализуют на практике один из принципов ООП — полиморфизм. Абстрактный класс может содержать (и не содержать) абстрактные методы и свойства.

17)Когда он описывает методы, которые применимы к классам, которые от него наследуются. Создавать экземпляры абстрактного класса нельзя. Назначение абстрактного класса заключается в предоставлении общего определения для базового класса, которое могут совместно использовать несколько производных классов. Например, в библиотеке классов может быть определен абстрактный класс, используемый в качестве параметра для многих из ее функций, поэтому программисты, использующие эту библиотеку, должны задать свою реализацию этого класса, создав производный класс.

18)Можно и нельзя создать объект. **абстрактный метод не может иметь тело (функциональность), может быть объявлена только в теле абстрактного класса, требует от производных классов её реализации (переопределения).** **Виртуальны метод может иметь функциональность, может быть объявлен в классе/структуре/абстрактном классе, не требует от производных классов его переопределения (но при этом может быть переопределён).**

**19) Виртуальный** **элемент** в базовом **классе** обозначается ключевым словом **virtual**, во всех унаследованных **классах** ключевым словом override. В C# **виртуальным** **элементом** **может** **быть** метод, событие, индексатор или свойство.

20)