1. Принципы ООП - наследование, полиморфизм, инкапсуляция. Наследование позволяет создавать новые классы на основе уже существующих, полиморфизм - использование одного и того же имени метода для разных классов, инкапсуляция - скрытие данных и методов внутри класса и предоставление доступа к ним только через определенные методы.

2. Класс .NET, от которого наследуются все классы - System.Object.

3. Открытые методы System.Object - Equals(), GetHashCode(), GetType(), ToString().

4. Закрытые методы System.Object - Finalize(), MemberwiseClone().

5. Пример определения класса:

public class MyClass {

private int myField;

public int MyProperty {

get { return myField; }

set { myField = value; }

}

public void MyMethod() {

//some code here

}

}

6. Ключевые слова при определении класса - public, private, protected, internal, static, abstract, sealed.

7. Объект - это экземпляр класса, созданный в памяти компьютера. Класс - это шаблон для создания объектов, содержащий описание его свойств и методов.

8. Конструктор - это специальный метод класса, который вызывается при создании нового объекта этого класса. Конструктор может принимать параметры или не принимать их вовсе.

9. Свойства конструктора - имя, параметры, тело метода.

10. Деструктор - это метод класса, который вызывается при уничтожении объекта этого класса. Деструктор не принимает параметров и не возвращает значения.

11. This - это ключевое слово, которое ссылается на текущий объект класса.

12. Результат выполнения программы: 5 5, 7 7. Переменная b ссылается на тот же объект, что и переменная a, поэтому изменение свойства Num у переменной a также отразится на переменной b.

13. Спецификаторы доступа для класса и методов класса в C# - public, private, protected, internal, protected internal.

14. Модификатор protected internal позволяет обращаться к членам класса из других классов в той же сборке или из производных классов в других сборках.

15. Ref и out параметры функции используются для передачи значений по ссылке. Ref используется для передачи уже инициализированной переменной, out - для передачи переменной, которая будет инициализирована внутри функции.

16. Пример необязательных и именованных параметров метода:

public void MyMethod(int a = 0, int b = 0) {

//some code here

}

public void MyMethod(int a, int b = 0, int c = 0) {

//some code here

}

17. Примеры полей класса - статические (static), константные (const), только для чтения (readonly).

18. Пример определения свойств класса:

public class MyClass {

private int myField;

public int MyProperty {

get { return myField; }

set { myField = value; }

}

}

Свойства связаны с инкапсуляцией, так как позволяют скрыть данные от прямого доступа и предоставить к ним доступ только через методы.

19. Явное имя параметра, передаваемого в метод set свойства класса - value.

20. Автоматические свойства - это свойства, которые не имеют явно определенного поля в классе, а хранятся внутри компилятора. Они определяются с помощью ключевого слова "prop" и могут иметь только геттеры и/или сеттеры.

21. Индексаторы класса - это методы, которые позволяют получать или устанавливать значения элементов массива или коллекции по индексу. Ограничения на индексатор - он должен иметь хотя бы один параметр, тип которого должен быть целочисленным или строковым.

22. Перегруженный метод - это метод с тем же именем, что и другой метод в классе, но с разными параметрами.

23. Частичный класс - это класс, который может быть разделен на несколько файлов. Преимущества частичных классов - удобство разделения больших классов на несколько файлов, возможность распределения работы между несколькими программистами.

24. Анонимный тип - это тип данных, который создается во время выполнения программы и не имеет явного имени. Он используется для хранения временных данных или для передачи данных между методами.

25. Статические классы используются для хранения методов и свойств, которые не требуют создания объекта этого класса.

26. Статическое поле принадлежит всем объектам класса, а экземплярное - только одному объекту это го класса.

27. Статический конструктор вызывается только один раз при первом обращении к статическому члену класса. Он используется для инициализации статических полей или выполнения других действий, которые требуются только один раз.

28. При поверхностном копировании копируются только ссылки на объекты, а не сами объекты. При глубоком копировании копируются все объекты и их содержимое.

29. Равенство объектов проверяет, равны ли ссылки на объекты, тождество объектов проверяет, равны ли значения объектов.

30. Частичные классы и методы используются для разделения кода класса на несколько файлов или для разделения кода метода на несколько частей.