1. Un paradigma consiste en un método para llevar a cabo cómputos y la forma en la que deben estructurarse y organizarse las tareas que debe realizar un programa.  
   El paradigma orientado a objetos propone resolver problemas de la realidad a través de identificar objetos y relaciones de colaboración entre ellos.
2. Los pilares del paradigma de objetos son:   
   Abstracción: Decide que es importante y que no lo es, se enfoca en lo que es importante y lo que no lo ignora.   
   Encapsulamiento: Es la que denota la capacidad del objeto de responder a peticiones a través de métodos y propiedades sin la necesidad de exponer los medios utilizados.  
   Herencia: Los objetos pueden heredad métodos y datos de otros. Esto hace mas fácil definir nuevos objetos a partir de otros que ya teníamos y facilita la reescritura de programas.  
   Polimorfismo: Un mismo nombre de un método puede hacer referencia a comportamientos distintos.
3. Una clase es una clasificación en base a comportamientos y atributos comunes, a partir de esta se crea un vocabulario. Es una abstracción de un objeto.
4. Los objetos son clases intanciadas, se crean en tiempo de ejecución y poseen comportamiento (métodos) y estado (atributos). Es una entidad concreta basada en una clase.
5. Instanciar un objeto es reservar espacio de memoria de un tipo de clase para ese objeto y se hace mediante la palabra reservada New. La variable solo contiene una refencia a la ubicación de dicho objeto.
6. La abstracción es aquellas características especificas de un objeto, las que lo distinguen de los demás y que logran definir límites conceptuales respecto a quien esta haciendo dicha abstracción, es básicamente le principal concepto para la creación de una clase.
7. Los atributos son las variables y determinan las características de un objeto.
8. Los métodos son las funciones de los objetos que determinan los comportamientos o que pueden hacer.
9. Un método estático indica que es un “Método de clase”, puede usarse sin necesidad de instanciar el objeto. No puede acceder a los atributos de la clase (salvo que sean estáticos), no puede utilizar el operador this, puede llamar a otro método siempre y cuando sea estático.
10. Una clase estática es aquella que se puede usar sin necesidad de realizar una instancia de la misma, cuando se declara una clase estática todos sus atributos y métodos tienen que ser estático por lo tanto no hay ninguna variable de instancia como para instanciar un objeto de esa clase. Además se llama una única vez a un constructor estático y no puede contener constructores de instancia.
11. Si se puede tener miembros estáticos en clases no-estáticas como atributos, métodos y sobrecarga de operadores. Una clase estática solo tiene miembros y métodos estáticos.
12. No, no es necesario instanciar un objeto si el método al cual se quiere llamar es estático.
13. Si el campo es estático el valor es el mismo para todas las intancias del objeto, en cambio si el valor no es estático puede tener diferentes valores para cada instancia.
14. El operador “this” no puede ser utilizado por métodos estáticos. No, un método estático solo puede acceder a miembros estáticos. Si, se puede acceder a miembros estáticos desde métodos de instancia.
15. No ya que las variables estáticas están hechas para tener un tiempo de vida largo, y al crearla dentro de un método al finalizar este esa variable se estaría “muriendo”.}
16. System.Math y System.Console
17. StringBuilder y
18. Un namespace es una agrupación lógica de clases y otros elementos, toda clase esta dentro de un NameSpace. Proporcionan un marco de trabajo jerárquico sobre el cual se construye y organiza todo el código. Su función principal es la organización del código para reducir conflictos entre nombres.
19. Si se puede tener todos los NameSpace que quieras mientras tengan distinto nombre.
20. Las directivas son elementos que permiten a un programa identificar los NameSpace que se usarán en el mismo. Permite el uso de los miembros de un NameSpace sin tener que especificar un nombre completamente cualificado.   
    La directiva using permite la especificación de una llamada a un método sin el uso obligatorio de un nombre completamente cualificado.
21. La directiva Alias permite utilizar un nombre distinto para un NameSpace.
22. No ya que dentro del mismo NameSpace abría un error por conflicto entre los nombres. Si estuvieran en distintos NameSpace si sería posible.