# Teoria .NET

1. **¿Qué es Microsoft Visual Studio?**

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) creado por Microsoft. Se utiliza para desarrollar una variedad de aplicaciones, desde aplicaciones de escritorio y aplicaciones web hasta aplicaciones móviles y servicios en la nube.

1. **¿Qué es el Framework .Net?**

NET es un framework de desarrollo de software desarrollado por Microsoft. Proporciona una plataforma para crear y ejecutar una variedad de aplicaciones, incluyendo aplicaciones de escritorio, aplicaciones web, aplicaciones móviles y servicios en la nube. .NET incluye un entorno de ejecución común y una biblioteca de clases que los desarrolladores pueden utilizar para crear aplicaciones en diversos lenguajes de programación, como C#, F# y Visual Basic. También ofrece herramientas para el desarrollo, la compilación y la administración de aplicaciones, lo que facilita la creación de software eficiente y escalable en diferentes plataformas.

1. **¿Qué es el CLR?**

El CLR (Common Language Runtime) es el entorno de ejecución en el framework .NET de Microsoft. Administra la ejecución de programas escritos en diferentes lenguajes, realizando compilación JIT, administración de memoria, seguridad y más.

1. **¿Qué es la BCL?**

La BCL (Base Class Library) es una colección de clases reutilizables y componentes predefinidos en el framework .NET de Microsoft. Proporciona una amplia gama de funcionalidades y abstracciones que los desarrolladores pueden utilizar al crear aplicaciones.

La BCL abarca aspectos como manipulación de cadenas, colecciones, E/S, seguridad, tiempo, matemáticas y más. Estas clases facilitan el desarrollo de aplicaciones al ofrecer soluciones comunes y abstracciones consistentes.

1. **Indique y explique el orden que se sigue en el proceso de compilación y ejecución en .NET**

* **Compilación:** El código fuente se convierte en Código Intermedio de Lenguaje Común (CIL).
* **Generación de ensamblados:** Los archivos CIL se agrupan en ensamblados (DLL o EXE).
* **Compilación JIT:** El CIL se convierte en código máquina nativo mientras se ejecuta.
* **Ejecución:** El código nativo se ejecuta en el entorno CLR.
* **Gestión de memoria y recursos:** El CLR administra la memoria y la recolección de basura.

**\*CIL y JIT:** Common Intermediate Language: Es un lenguaje intermedio en el entorno .NET. Se genera durante la compilación y es independiente del lenguaje fuente. Permite la portabilidad y la compilación Just-In-Time (JIT) en diferentes plataformas.

1. **¿Cuál es la signatura de Main?**

static void Main(string[] args)

1. **¿Qué es un espacio de nombre (namespace)?**

En C#, un espacio de nombres (namespace) es una forma de organizar y agrupar clases y tipos relacionados en un programa para evitar conflictos de nombres y mantener el código ordenado.

1. **¿Cómo se incluye una librería en C#?**

* Agrega una referencia a la librería en tu proyecto.
* Usa el espacio de nombres de la librería con using.
* Utiliza las clases y tipos de la librería en tu código.
* Compila y ejecuta tu proyecto para ver los resultados.

Ejemplo: using System;

1. **¿Qué indica la palabra clave params?**

La palabra clave params en C# se utiliza para definir un parámetro de método que puede aceptar un número variable de argumentos del mismo tipo. Permite que un método reciba una lista separada por comas de valores de un tipo específico sin tener que pasar un array explícito como argumento.

# Objetos

1. **¿Qué es una clase?**

En C#, una clase es un plano para crear objetos. Define atributos (propiedades) y comportamientos (métodos) que los objetos tendrán. Las clases encapsulan la estructura y el comportamiento en la programación orientada a objetos.

1. **¿Qué es un objeto?**

Un objeto es una instancia específica de una clase en la programación orientada a objetos (POO). En otras palabras, un objeto es una entidad concreta que se crea a partir de una clase y que tiene atributos y comportamientos definidos por esa clase.

1. **¿Porque se caracterizan los objetos?**

Los objetos se caracterizan por dos aspectos fundamentales en la programación orientada a objetos:

* **Atributos (Propiedades):** Los objetos tienen características o atributos que los describen y diferencian entre sí. Estos atributos son representados por variables y almacenan valores específicos para cada objeto. Por ejemplo, en una clase "Coche", los atributos podrían ser "marca", "modelo" y "color".
* **Comportamientos (Métodos):** Los objetos pueden realizar acciones o responder a eventos. Estas acciones están definidas por los métodos de la clase a la que pertenecen. Los métodos describen cómo un objeto puede interactuar y afectar su entorno. Siguiendo el ejemplo del "Coche", los métodos podrían ser "arrancar", "detener" y "acelerar".

1. **¿Cómo se llama la táctica de obtener la forma mínima y esencial de un objeto?**

La táctica se llama "abstracción". Consiste en simplificar un objeto al identificar y definir sus aspectos esenciales, omitiendo detalles irrelevantes para un diseño más claro y eficiente en la programación orientada a objetos.