Ordena estos lenguajes **de más a menos abstracto:**
DSL(especifico de dominio),ensamblado,C, C#

Ensamblador, C, C# y DSL.

Indica si son paradigma declarativos o imperativos:
Orientada a objetos, funcional, estructurado basado en procedimientos, lógico

Orientada a objetos imperativo, funcional delarativo, estructurado basado en procedimiento imperativo y lógico declarativo.

Ventajas de la POO, es decir, programación orientada a objetos.

Reutilización de código, Modularidad, Flexibilidad y escalabilidad y Modelado del mundo real

Problemas de la herencia múltiple

Los principales problemas de la herencia múltiple son la posible coincidencia de nombre o herencias repetidas

Definir metaprogramación (o generación dinámica de código)

La metaprogramación es una técnica de programación que permite escribir programas que manipulan, generan o transforman otros programas o incluso a sí mismos. Es la capacidad de un programa para analizar, generar y manipular su propio código fuente o el código fuente de otros programas durante el tiempo de compilación o ejecución.

Te dan código con un problema de Lock, y te ofrecen cuatro opciones para bloquear: Lock(¿????) El problema era más o menos : "Tienes una clase comun en la que hay un FileLog, y escribes en ese mismo fichero, a partir de diferentes hilos. Evidentemente el FileLog es un recurso compartido y puede llegar a dar problemas. ¿Cómo arreglas este codigo? Indica para las cuatro opciones que te damos, el porqué sí o porqué no son buenas o malas ideas. Deja claro cual eliges".

```
a. This -> No nos sirve

b. Objeto anonimo bloqueador, unico para cada clase (es decir, teniamos la clase
comun en la que hay un fichero compartido, y otra clase para cada Worker, me
parece. Pues este objeto se crea nuevo en cada una de estas) -> No nos sirve

c. Console.out -> Sirve como método de sincronización aunque en este contexto no
tiene importancia, es un recurso que comparten todos los hilos y apunta al mismo
```

```
objeto

d. FileLogger (que es el mismo para todos porque es donde escribe el fichero) ->
Esta sería la mejor solución posible
```

Con estos dos métodos, indica si devuelve true o false:
Ejercicio para comprobar el tipado dinámico

```
CheckDynamic(a):

CheckDynamic(b):

CheckDynamic(c):

CheckDynamic(d):
```

Indica a qué tipos inferencia **el compilador:**

Te daba dos métodos y tenias que interpretar qué pensará el compilador para esas variables y siguiendo el codigo que daba

```
Var Q1:

Var Q2:

Var Q3:
```

Invocar dos veces a una funcion lambda, aplicando el uso de *true* y *fals*e. Similar a los de seminario. Muy importante no saltarse ningún paso, mejor escribir mucho.

Escribir código en c# o pseudocodigo del método reducir hecho en los laboratorios. Después escribir una invoación para una lista de numero {1,2,3,4} obtener su producto. (Aquí creo que aunque no te lo especificaba, tenias que ponerle la famosa "semilla". Porque si empiezas con el acumulador a cero, e intentas hacer 0 por 1, siempre vas a arrastar el cero. Si hubieran pedido suma, y no el producto, no habría este problema)

Una pregunta sobre si es referencialmente transparente. Decir qué es, y justificar ese caso. Nota: aparecia una variable aux que no se usaba para nada en el retorno, pero su valor se calcula en cada llamada con un random

Otro ejercicio sobre qué imprime por consola: si lanza la excepción o muesra el mensaje del código (parecido a los que están en los examenes del 2020, basicamente es para

saber cómo funcionan las excepciones)

Pregunta si se capturará o no la excepción dentro de ese código con los hilos.

Dos metodos: metodo1() y metodo2(). Si hay tres bucles for anidados en cada, pero en metodo1() se hace ParallelFor al for más externo, y en método2 al más interno. ¿Cuál es la mejor opción?