 **Closure y funciones anidadas**  
Diseña una función que genere otra función para incrementar un número en una cantidad específica. La función generada debe recordar el valor del incremento proporcionado al momento de su creación.

 **Funciones Lambda**  
Escribe una función que reciba una lista de números y una lambda como parámetro. La lambda debe evaluar si un número cumple con una condición (por ejemplo, si es par). Filtra los números que cumplan dicha condición.

 **Funciones de Orden Superior**  
Crea una función de orden superior que reciba otra función como argumento. La función proporcionada debe transformar un número entero en un valor personalizado (ejemplo: convertirlo en su representación binaria).

 **Funciones de Extensión**  
Añade una función de extensión a la clase String para que determine si una cadena es un palíndromo (se lee igual de izquierda a derecha y viceversa).

 **Sobrecarga de Operadores**  
Implementa una clase que represente un vector en 2D y redefine los operadores + y - para que sea posible sumar y restar vectores directamente.

 **Funciones Inline**  
Diseña una función de orden superior que reciba una lambda para procesar elementos de una lista. Usa inline para optimizar la ejecución de la función.

 **Funciones Reified**  
Implementa una función genérica que permita verificar si un objeto pertenece a un tipo específico en tiempo de ejecución usando reified.

 **Funciones Infix**  
Crea una clase que represente una fracción. Implementa una función infix que permita sumar dos fracciones directamente.

 **Lambda con Receiver**  
Define una lambda con receptor que permita configurar un objeto de configuración (por ejemplo, ajustar propiedades como nombre y tamaño).

 **Funciones con DSL**  
Diseña un mini lenguaje específico de dominio (DSL) para construir una receta. Debe permitir definir pasos como añadir ingredientes y establecer tiempos de cocción.