TRABAJO PRÁCTICO 2: RECURSIÓN

1. Para las siguientes funciones recursivas, dibuje la pila de ejecución y calcule cuántas llamadas recursivas se realizan en total:

```
func factorial(n int) int {
   if n == 0 {
      return 1
   }
   return n * factorial(n-1)
}
```

   a. Dibuje la pila de ejecución para factorial(5).

```
func fibonacci(n int) int {
   if n <= 1 {
      return n
   }
   return fibonacci(n-1) + fibonacci(n-2)
}
```

   b. Dibuje la pila de ejecución para fibonacci(4). ¿Cuántas veces se llama a fibonacci(2)?

2. Implemente en GO las siguientes funciones en forma recursiva e iterativa. Calcule para cada caso el orden de complejidad. Justifique.
   a. **Cuenta regresiva**: recibe un número entero e imprime todos los números comprendidos entre el mismo y 0.
   b. **Suma de enteros:** permite sumar todos los números enteros comprendidos entre un parámetro de inicio y uno de fin.
   c. **Vocales y consonantes:** devuelve la cantidad de vocales y de consonantes que contiene una cadena.

d. **Mayor elemento:** dado un arreglo de enteros, devolver el mayor elemento.
e. **Invertir:** dado un arreglo de enteros, invertirlo.

3. Dado un arreglo ordenado de números enteros y un número a buscar, describa paso a paso cómo se ejecuta el algoritmo de búsqueda binaria recursivo.

Arreglo = [3, 8, 15, 19, 25, 31, 42, 57, 63, 79] Buscar = 31

Pasos a seguir: Escribe los valores de inicio, fin y medio en cada llamada recursiva. Indica qué parte del arreglo se descarta en cada paso. Explica qué sucede si el número está presente o si no se encuentra en el arreglo.

4. Dado un arreglo desordenado de números enteros, describa paso a paso cómo el algoritmo MergeSort lo divide y lo combina.

Arreglo = [38, 27, 43, 3, 9, 82, 10]

Pasos a seguir: Muestra cómo el arreglo se divide en subarreglos en cada nivel de recursión. Indica en qué momento se empiezan a combinar los subarreglos. Explica cómo se realiza la fusión ordenada de los elementos.