UNIDAD TEMÁTICA 2: Diseño de Algoritmos

PRÁCTICOS DOMICILIARIOS INDIVIDUALES - 6

Escenario

Se desea implementar diferentes alternativas de resolver el cálculo de números de Fibonacci mediante técnicas de Programación Dinámica.

Los números de Fibonacci se definen recursivamente:

- $F_0 = 0$
- $F_1 = 1$
- $F_i = F_{i-1} + F_{i-2}$ para todo $i > 1$.

Ejercicio 1: enfoque top-down

Top-down: primero descomponemos en sub-problemas y luego se calculan y almacenan los resultados

SEUDOCÓDIGO

Memo ← {} //diccionario, mapa

fib(n)

COM

SI **n** no está en *memo*

SI n <= 2

memo[n] ← 1

SINO

memo[n] ← fib(n − 1) + fib(n − 2)

devolver *memo*[n]

FIN

Desarrolla una función en JAVA que implemente este algoritmo y pruébala para diferentes números. Agrégale un contador de "invocaciones" y uno de accesos a los valores ya almacenados, e indica, como resultado final, la cantidad de veces que se invoca al método recursivo completo y la cantidad de veces que se utilizan valores previamente almacenados.

- ¿qué comentarios te merece esta implementación?
- ¿qué costos de memoria tiene asociados?
- ¿qué limitaciones puede tener?

Ejercicio 2: enfoque bottom-up

Primero calculamos los subproblemas…..

SEUDOCÓDIGO

fib(n)

COM

```
SI n = 0
        devolver 0
    SINO
        fibAnterior ← 0,
        fibActual ← 1
        REPETIR (n − 1) veces // cortamos para n = 1
         fibNuevo ←  fibActual + fibAnterior
            fibAnterior ← fibActual
            fibActual ← fibNuevo
        FIN REPETIR
    devolver fibActual
```

FIN

Desarrolla una función en JAVA que implemente este algoritmo y pruébala para diferentes números.

- ¿qué comentarios te merece esta implementación?
- ¿qué costos de memoria tiene asociados?
- ¿qué limitaciones puede tener?
- ¿cómo se compara con la anterior?