TSP - PROGRAMACION II

PROGRAMACIÓN DINÁMICA

PROGRAMACIÓN DINÁMICA

La programación dinámica es una técnica que consiste en:

- Dividir un problema en dos o más subproblemas o reducirlo a una instancia más fácil de calcular del problema.
- Resolver las instancias de cada subproblema de la misma manera (dividiendo en subproblemas o reduciendo a otra instancia) hasta llegar a un caso base.
- Guardar el resultado de cada instancia del problema la primera vez que se calcula, para que cada vez que se vuelva a necesitar el valor de esa instancia ya esté almacenado, y no sea necesario calcularlo nuevamente.

PROGRAMACIÓN DINÁMICA

La programación dinámica es un método para reducir el tiempo de ejecución de un algoritmo mediante la utilización de subproblemas superpuestos y subestructuras óptimas.

Una subestructura óptima: significa que se pueden usar soluciones óptimas de subproblemas para encontrar la solución óptima del problema en su conjunto.

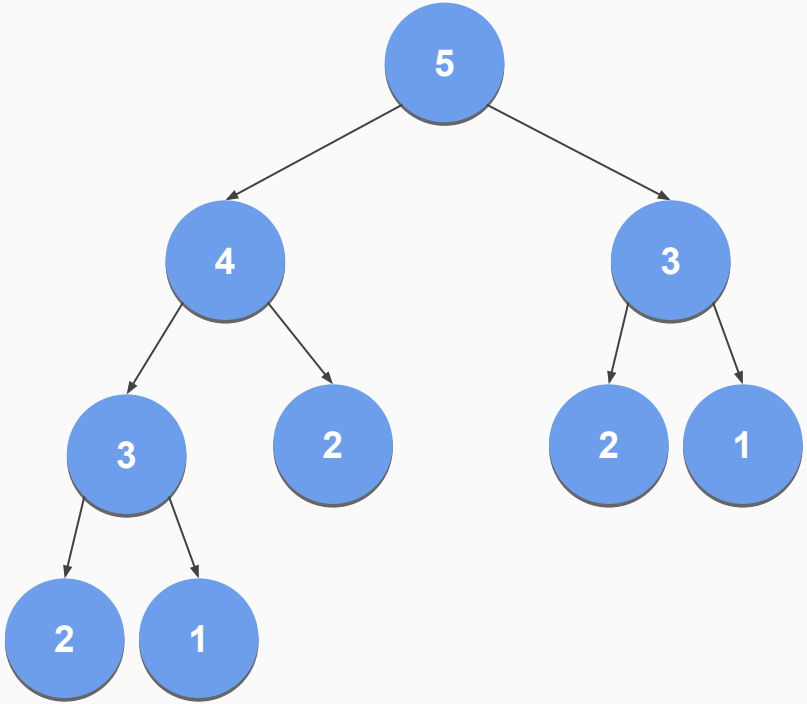
La programación dinámica consiste, esencialmente, en una recursión con una suerte de **memorización** o **cache**.

PROGRAMACIÓN DINÁMICA - Tipos de enfoque

Top-down: El problema se divide en subproblemas, y estos se resuelven recordando las soluciones por si fueran necesarias nuevamente. Es una combinación de **memoización** y **recursión**.

Bottom-up: Todos los problemas que puedan ser necesarios se resuelven de antemano y después se usan para resolver las soluciones a problemas mayores. Este enfoque es ligeramente mejor en consumo de espacio y llamadas a funciones, pero a veces resulta poco intuitivo encontrar todos los subproblemas necesarios para resolver un problema dado.

PROGRAMACIÓN DINÁMICA - Fibonacci



PROGRAMACIÓN DINÁMICA - Fibonacci - Top-Down

Top-down: Es una combinación de **memoización** y **recursión**.

```
FUNC Fibonacci (↓n: NATURAL, ↕tabla: ARRAY [0..n] DE NATURALES): NATURAL
 VARIABLES
    i: NATURAL
 INICIO
     SI n <= 1 ENTONCES
         devolver n
    FIN SI
     SI tabla[n-1] = -1 ENTONCES
        tabla[n-1] := Fibonacci(n-1, tabla)
    FIN SI
     SI tabla[n-2] = -1 ENTONCES
        tabla[n-2] := Fibonacci(n-2, tabla)
    FIN SI
        tabla[n] := tabla[n-1] + tabla[n-2]
    devolver tabla[n]
FIN
```

tabla (n)	valor
0	
1	
2	
3	
4	
5	

PROGRAMACIÓN DINÁMICA - Fibonacci - Bottom-up

Bottom-up: Todos los problemas se resuelven de antemano y después se usan para resolver las soluciones a problemas mayores.

```
FUNC Fibonacci (↓n: NATURAL): NATURAL
VARIABLES
    tabla: ARRAY [0..n] DE NATURALES
    i: NATURAL
INICIO
    SI n = 0 ENTONCES
        DEVOLVER 0
    SI NO, SI n = 1 ENTONCES
        DEVOLVER 1
    SI NO
        tabla[0] := 0
        tabla[1] := 1
        PARA i = 2 HASTA n HACER
            tabla[i] := tabla[i-1] + tabla[i-2]
        FIN PARA
        DEVOLVER tabla[n]
    FIN SI
FIN
```

tabla (n)	valor
0	
1	
2	
3	
4	
5	