TEMA 4: DISEÑO DE ALGORITMOS RECURSIVOS

1. Principio de inducción
2. Diseño recursivo
3. Coste temporal de un algoritmo recursivo
4. Coste espacial de un algoritmo recursivo
5. Inmersión de programas
6. Inmersión de especificaciones

3. Coste temporal de un algoritmo recursivo

- El problema en el cálculo del coste de una función recursiva es que no conocemos el coste de la llamada recursiva.

```
factorial(n: natural) dev f: natural      // Coste
si n = 0 entonces                         // 1
  f <- 1                                  // 1
sino
  f <- n * factorial(n - 1)               // 1 + ???
fsi
```

- Si denominamos T(n) al coste de la función factorial, el coste de la llamada recursiva será T(n – 1).

3. Coste temporal de un algoritmo recursivo

- Coste de la función factorial:

```
factorial(n: natural) dev f: natural      // Coste
si n = 0 entonces                         // 1
  f <- 1                                  // 1
sino
  f <- n * factorial(n - 1)               // 1 + T(n-1)
fsi
```

$$T(n) = \begin{cases} 2 & \text{si } n = 0 \\ T(n-1) + 2 & \text{si } n > 0 \end{cases}$$

3. Coste temporal de un algoritmo recursivo

Def: Llamaremos **recurrencia** a cualquier expresión que relacione el valor de una función T para un parámetro n, con uno o más valores de la función para parámetros anteriores a n.

Def: Una función T(n) es **solución de una recurrencia** si al sustituir el valor de la función en la recurrencia se obtiene una expresión cierta para todos los valores de n especificados.

Solución de la recurrencia del factorial:

T(n) = 2n + 2