*NOTAS 2.

Estructuras de Datos

* Análisis con algoritmos Recursivos *

* Ejemplo 1:

```
// Función suma, regresa la suma 1+2+...+n,
// donde n>= 1

int funcSuma (int n) {
        if (n==1) {
            return 1;
        }
        return n + funcSuma (n-1);
}
```

Sea la función $T(n)$ el número de operaciones elementales realizadas por la llamada de la función funcSuma(n)

→ Identificamos dos propiedades de $T(n)$

1) Dado que funcSum(1) se calcula utilizando un número fijo de operaciones k_1, $T(1) = k_1$

2) Si $n > 1$ la función realizará un número fijo de operaciones k_2, y además, hará una llamada recursiva a funcSuma(n-1).

Esta llamada recursiva realizará operaciones $T(n-1)$.

Por lo tanto tendremos $T(n) = K_2 + T(n-1)$

- Si solo buscamos una estimación asintótica de la complejidad del tiempo, no es necesario especificar los valores reales de las constantes k_1 y k_2. En su lugar, dejamos a $K_1 = K_2 = 1$. Para encontrar la complejidad de tiempo de funcSoma, podemos resolver la relación de recorrencia

$$T(n) = \begin{cases} 1 \text{ si } n = 1 & \cdots (*) \\ 1 + T(n-1), \text{ si } n > 1 & \cdots (**) \end{cases}$$

Aplicando de manera repetitiva tenemos que: podemos calcular a $T(n)$ para cualquier número positivo $\underline{n}$

1ª iteración para $n=1$ tenemos el caso (*) de la recurrencia.

Por lo tanto $T(n) = T(1) = 1$

→ Probemos para $n>1$, tenemos:

$$T(n) = 1 + T(n-1) \dashrightarrow \text{iteración } 1$$

$$\Rightarrow \quad 1 + (1 + T(n-2)) \dashrightarrow \text{iteración } 2$$
$$= 2 + T(n-2)$$

$$\Rightarrow \quad 2 + (1 + T(n-3)) \dashrightarrow \text{iteración } 3$$
$$= 3 + T(n-3)$$

$$\vdots$$

así seguimos para $k = n-1$, iteraciones.

∴ Sea $k = n-1$ iteraciones

∴ sustituyo el valor de k en la segunda relación (**)

$$\hookrightarrow \quad k + T(n-k)$$
$$= (n-1) + T(n-(n-1))$$
$$= (n-1) + T(\cancel{n-n}+1)$$
$$= n-1 + \underbrace{T(1)}$$

↳ por la relac. de recurrencia (*) equivale a 1

$$= n \cancel{-1+1}$$
$$= n$$

∴ T. ejec. = n ∴ $O(n)$ es lineal

¡Ojo! NOTACION: i.e.
Significa: "es decir"

Ejemplo 2.-

Sea el siguiente algoritmo:

```
1   void Prueba(int n) {
2       if (n>0) {
3           for (int i=0; i<n; i++) {
4               System.out.println(n); --- 1 op.el.
5           }
6           Prueba(n-1);
7       }
8   }
```

¿Como calculamos su tiempo de ejecución?

1°.- analizamos línea por línea, i.e. en la línea 2 tenemos solo una operación ya que si $n=0$ entonces regresamos false (podemos denotar a "false" como 1)

2°.- En nuestro ciclo tenemos en la línea 3 las operaciones $1+(n+1)+n$, adentro del ciclo for solo tendremos una constante.
$\therefore$ el tiempo para nuestro ciclo es
$$1+(n+1)+n$$

3°.- Por último en la línea 6 tenemos a $T(n-1)$ (nuestra llamada recursiva)

¿Por qué "n-1"? porque serán n-1 veces en la que llamaremos a nuestro algoritmo.

Ahora, con todos estos datos obtenemos que:

$$T(n) = T(n-1) + (2n+2)$$

línea 6

regla de la suma porque no hay ciclos anidados

↳ línea 3 $(1+n+1+n)$

↳ esto a su vez ¿qué tiempo tendrá?
= Sería lineal y lo denotamos por "n"

Por lo que podemos deducir nuestra relación de recurrencia como:

$$T(n) = \begin{cases} 1 & \text{si } n=0 \text{, es false} \\ T(n-1)+n & \text{, si } n>0 \text{, n constante.} \end{cases}$$

P. D para cualquier número positivo n, tenemos:

Sea $T(n) = \underline{T(n-1)} + n$ (*) --→ iteración 1

sustituimos para $\underline{T(n-1)} = \underline{T(n-2)+n-1}$

porque entramos de nuevo a la llamada recursiva

⇒ sustituimos •→ lo subrayado en morado en (*)

$$= \underbrace{[T(n-2)+(n-1)]}_{T(n-1)} + n$$

$$= \underline{T(n-2)} + (n-1) + n$$

volvemos a llamar la función, por lo que $T(n-2)$ es igual a:

$$\underline{T(n-2)} = \underline{T(n-3) + n-2}$$

→ sustituimos

$$= \underbrace{T(n-3) + (n-2)}_{T(n-2)} + (n-1) + n \quad -- \; **$$

→ Ahora supongamos que se cumple para cualquier $K > 0$, por lo tanto podemos ver a ** como:

$$= T(n-K) + (n-(K-1)) + (n-(K-2)) + \dots$$
$$\dots (n-1) + n$$

Asumiendo que $n - K = 0$ tenemos que
$$n = K$$

Por lo tanto:

$$T(n) = T(\cancel{n-n})^{0} + (\cancel{n-n}+1) + (\cancel{n-n}+2) + \dots$$
$$\dots (n-1) + n$$

$$T(n) = \underbrace{T(0) + 1 + 2 + 3 + \dots + (n-1) + n}$$

$$= 1 + \frac{n(n+1)}{2}$$

Por lo tanto $1 + \frac{n(n+1)}{2} = 1 + \frac{n^2}{2} + \frac{n}{2}$

T. ejec. $= \frac{n^2}{2} + \frac{n}{2} + 1$

$\therefore$ la complejidad es $O(n^2)$