Sea $f(n) = 10n^3 + 3$ y se sabe que para $c_1 = 12$ se cumple que $10n^3 + 3 \leq c_1 n^3 \ \forall n \geq 2$ y que para $c_2 = 15$ que $10n^3 + 3 \leq c_2 n^4 \ \forall n \geq 1$

$f(n)$ es $O(n^4)$ ✓

$f(n)$ es $O(n^3)$ ✓

La cota más representativa de $f(n)$ es $O(n^4)$ ✗

La cota más representativa de $f(n)$ es $O(n^3)$ ✓

$O(n) \subset O(n^2)$

$\Theta(f(n)) + \Theta(g(n)) = \Theta(\max(f(n), g(n)))$

si $f(n) \in \Theta(h(n))$ entonces $f(n) + b \in \Theta(h(n))$

Si $t_1(n)$ y $t_2(n)$ son los tiempos de dos implementaciones de un mismo algoritmo, se verifica que $\exists c \in \mathbb{R}^+ \wedge \exists n^o \in \mathbb{N} \mid t_1(n) \leq c\, t_2(n) \ \forall n \geq n_0$

Resolver la siguiente ecuación de recurrencia:

$$T(n) = \begin{cases} c_1 & \text{si } n \leq 2 \\ T(n-1) + T(n-2) + T(n \operatorname{div} 3) + c_2 & \text{si } n > 2 \end{cases}$$

$$T(n) = T(n-1) + T(n-2) + T(n \operatorname{div} 3) + c_2 =$$
$$= T(n-2) + T(n-3) + T((n-1)\operatorname{div} 3) + c_2 + T(n-3) + T(n-4) + T((n-2)\operatorname{div} 3) + c_2 +$$
$$+ T(n \operatorname{div} 3 - 2) + T(n \operatorname{div} 3 - 1) + T(n \operatorname{div} 3^2) + c_2 + c_2 \ldots$$

En la expresión anterior el nº de términos aumenta conforme avanzamos hacia la base, siendo difícil establecer una fórmula general. Para solventar este problema podemos acotar las ecuaciones $T_1(n)$ y $T_2(n)$

$$T_1(n) = \begin{cases} c_1 & \text{si } n \leq 2 \\ 3T(n-1) + c_2 & \text{si } n > 2 \end{cases}$$

$$T_1(n) = 3T(n-1) + c_2 = 3[3T(n-2) + c_2] + c_2 = 3^2 T(n-2) + 4c_2 =$$
$$= 3^2[3T(n-3) + c_2] + 3c_2 + c_2 = 3^3 T(n-3) + 3^2 c_2 + 3c_2 + c_2 \ldots =$$
$$= 3^i T(n-i) + \sum_{j=0}^{i-1} 3^j c_2$$

El caso base se alcanza cuando $n - i \leq 2$, por tanto $i \approx n$ por lo que:

$$= 3^n c_1 + \sum_{j=0}^{n-1} 3^j c_2 \in \Theta(3^n)$$

$$T_2(n) = \begin{cases} c_1 & \text{si } n \leq 2 \\ 3T(n \operatorname{div} 3) + c_2 & \text{si } n > 2 \end{cases}$$

$$T_2(n) = 3T_2(n \operatorname{div} 3) + c_2 =$$
$$= 3[3T_2(n \operatorname{div} 3^2) + c_2] + c_2 =$$
$$= 3^2 T_2(n \operatorname{div} 3^2) + 3c_2 + c_2 =$$
$$= 3^2[3T_2(n \operatorname{div} 3^3) + c_2] + 3c_2 + c_2 =$$
$$= 3^3 T_2(n \operatorname{div} 3^3) + 3^2 c_2 + c_2 + c_2 = \ldots = 3^i T(n \operatorname{div} 3^i) + \sum_{j=0}^{i-1} 3^j c_2$$

El caso base se alcanza para $i \approx \log_3 n$, por lo que sustituimos i por su valor en la expresión:

$$= 3^{\log_3 n} c_1 + \sum_{j=0}^{\log_3 n} 3^j c_2 \in \Theta(n)$$

$$T_2(n) = \Theta(3^{\log_3 n}) = \Theta(n^{\log_3 3}) = \Theta(n)$$

$T(n) \in \Omega(n)$

$T(n) \in \Theta(3^n)$

```
Funcion Cuestion (V[1..num]: vector de enteros, num: entero) retorna (s: entero)
    var k, p, q : entero fvar
    para k = 1 hasta num - 1 hacer
            q = AUX(V, K, num);
            p = v[k]
            v[k] = v[q]
            v[q] = p
    fpara
    retorna 0
ffuncion
```

```
Funcion AUX ( V[1..n] : vector de enteros; inicio, fin : enteros) retorna (t : entero)
        var i, m : entero fvar
        m = inicio
        para i = inicio + 1 hasta fin hacer :
             si (v[i] > v[m]) entonces m = i fsi
        fpara
        retorna pos
ffuncion
```

Talla: número de componentes de la sección a tratar, en este caso fin - inicio + 1 = num.
En principio existe mejor y peor caso, debido a la sentencia (V[i] > v[m]) ya que en función de su cumplimiento o no se realiza m = i.
El mejor caso se produce cuando v[i] <= v[m] :

$$T_{mc}(n) = 1 + \sum_{i=inicio+1}^{fin} 1 + 1 = 1 + 1 + (fin - inicio + 1 - 1) = 2 + fin - inicio =$$
$$= 1 + num \in \Theta(num)$$

En el caso de que (V[i] > V[m]), sería el peor caso. En ese supuesto el coste temporal de la función AUX es el siguiente:

$$T_{pc}(n) = 1 + \sum_{i=inicio+1}^{fin} 1 + 1 = 2 + (fin - inicio - 1 + 1) = 1 + fin - inicio + 1$$
$$= 1 + num \in \Theta(num)$$

Ambas complejidades coinciden

Funcion Cuestión

$$T(n) = \sum_{K=1}^{num-1} (3 + \overbrace{(num - K)}^{AUX(fin-inicio)}) = \sum_{K=1}^{num-1} 3 + \sum_{K=1}^{num-1} (num - K) = 3(num - 1) + \sum_{K=1}^{num-1} K =$$
$$= 3num - 3 + \frac{(1 + num - 1)}{2}(num - 1) = 3num + \frac{1}{2}num^2 - \frac{1}{2}num \in \Theta(num^2)$$

$Q = \{1 \leq inicio \leq fin + 1 \leq num + 1\}$

```
Funcion Cuestion ( v[1..num] : vector de enteros; inicio, fin : enteros) retorna (r: entero)
        var k, p: entero fvar
        p = 1
        si inicio >= fin retorna inicio * fin
        si no k = 1
              mientras <=100 hacer
                    p = p + v[inicio] * k
                    k = k + 1
              fmientras
              retorna p * Cuestion (v, inicio + 1, fin - 1)
    fsi
ffuncion
```

Talla: tamaño de la sección del vector a tratar, es decir fin - inicio + 1, lo renombraremos num.

No existe mejor y peor caso, ya que de cara al calculo de la eficiencia, el condicional puede inducir a cuestionamiento, pero a efectos prácticos, si inicio ≥ fin, si la talla del problema es menor o igual que 1, lo que es irrelevante al cálculo de la eficiencia.

Para tallas mayor que 1 (inicio < fin) se realiza una llamada recursiva en la que la talla del problema se reduce en dos unidades, pero previa llamada ocurre un bucle mientras cuya var de iteracion toma un valor inicial de 1. Siempre se realizan el mismo número de iteraciones, 100, por lo que su complejidad es constante. Dado que en cada iteración del bucle se realizan dos operaciones de coste unitario:

$$1 + \sum_{j=1}^{100} 2 = 1 + 2(100 - 1 + 1) = 1 + 200 \in \Theta(1)$$

La ecuacion de recurrencia resuelta:

$$T(n) = \begin{cases} C_1 & \text{si } n \leq 1 \\ T(n-2) + C_2 & \text{si } n > 1 \end{cases}$$

$$T(n) = T(n-2) + C_2 = T(n-4) + C_2 + C_2 = T(n-6) + C_2 + 2C_2 = T(n-2i) + iC_2$$

El caso base se produce cuando $n - 2i \leq 1$, $n/2 \leq i$

$$\approx T(0) + \frac{n}{2} C_2 = C_1 + \frac{n}{2} C_2 \in \Theta(n)$$