

Relacion-T1-y-T2-ADA (2:2).pdf



Valeaal



Análisis y diseño de algoritmos



2º Grado en Ingeniería del Software



Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

The collage consists of four vertical panels. From left to right: 1. A close-up of golden wheat ears with the text 'MALTA DE CEBADA' overlaid. 2. A waterfall cascading down rocks, with the text 'AGUA' overlaid vertically. 3. A bunch of green hops hanging from a vine, with the text 'LÚPULO' overlaid vertically. 4. A close-up of a frothy, yellow-orange head of beer, representing yeast.

Tu CERVEZA
se hace a partir de ingredientes naturales.
Descubre más en cerveceros.org

 DISFRUTA DE LO NATURAL

Cerveceros de España recomienda el consumo responsable +18

EL PODCAST DE MUOLAH

temporada 1

No sé en qué momento nos pareció buena idea lanzar nuestro podcast para estudiantes en verano.

escúchate o algo, así te diré a mi jefe que ha sido un éxito

3. Supongamos la función $T(n) = 2T(n-1) + 3^n(n+5)$, con $n > 0$. Se pide calcular el orden de complejidad resolviendo la ecuación de forma exacta para la condición inicial $T(0) = 0$.

$$T(n) - 2T(n-1) + 3^n(n+5) = 0$$

$$T(n) - 2T(n-1) + 3^n \cdot 3 + 3^n \cdot 5 = 0$$

$$T(n) - 2T(n-1) + 3^{n+1} \cdot 5 = 0$$

$$\frac{1}{x-2} \downarrow \quad x \rightarrow 3 \text{ grado polinomio, } 1 \rightarrow (3n, \text{ grado }) + 1$$

$$(x-2)' \cdot (x-3)^2 = 0$$

$$(x-2) \cdot (x^2-6x+9) = 0$$

$$x^3 - 6x^2 + 9x - 2x^2 + 12x - 18 = 0$$

$$x^3 - 8x^2 + 21x - 18 = 0 \rightarrow \text{Hallar los } x \text{ para tener las raíces y la multiplicidad}$$

Globos a partir del anexo

$$T(1) = 2T(0) + 3^1 \cdot 6 = 18$$

$$T(2) = 2T(1) + 3^2 \cdot 11 = 99$$

Resolver con las
valores de n calculados

$$\begin{array}{|ccc|} \hline & 1 & -8 & 21 & -18 \\ 3 & \downarrow & 3 & 15 & 18 \\ & 1 & -5 & 6 & 0 \\ \hline \end{array}$$

cc 2' grado

$$r_1 = 2 \rightarrow m=1 \longrightarrow z^n \cdot \Delta \longrightarrow z^n(\Delta) + 3^n(Bn+C)$$

$$r_2 = 3 \rightarrow m=2 \longrightarrow 3^n \cdot (Bn+C)$$

$$\left. \begin{array}{l} n=0 \rightarrow \Delta + C = 0 \\ n=1 \rightarrow 2\Delta + 3B + 3C = 18 \\ n=2 \rightarrow 4\Delta + 18B + 9C = 99 \end{array} \right\} \begin{array}{l} \Delta = -9 \\ B = 3 \\ C = 9 \end{array}$$

4. Resolver las siguientes ecuaciones y dar su orden de complejidad:

- $T(n) = 3T(n-1) + 4T(n-2)$ si $n > 1$; $T(0) = 0$; $T(1) = 1$.
- $T(n) = 4T(n/2) + n^2$ si $n > 4$, n potencia de 2; $T(1) = 1$; $T(2) = 8$.
- $T(n) = 2T(n/2) + n \log n$ si $n > 1$, n potencia de 2.
- $T(n) = 3T(n/2) + 5n + 3$ si $n > 1$, n potencia de 2.
- $T(n) = 2T(n/2) + \log n$ si $n > 1$, n potencia de 2.
- $T(n) = T(n-1) + 2T(n-2) - 2T(n-3)$ si $n > 2$; $T(n) = 9n^2 - 15n + 106$ si $n = 0, 1, 2$.
- $T(n) = 2T(n/4) + n^{1/2}$ si $n > 4$, n potencia de 4.
- $T(n) = 4T(n/3) + n^2$ si $n > 3$, n potencia de 3.

a) $T(n) - 3T(n-1) - 4T(n-2) \rightarrow$ Usando resto n, el grado menor es 2, se resuelven los coeficientes

$$\begin{cases} r_1 = 4 & m=1 \\ r_2 = -1 & m=1 \end{cases} \rightarrow 4^n \Delta + (-1)^n B \rightarrow \begin{cases} T(0) = 0 \text{ y} \\ T(1) = 1 \end{cases}$$

Resolvemos usando $\begin{cases} n=0 \rightarrow \Delta + B = 0 \\ n=1 \rightarrow 4\Delta - B = 1 \end{cases} \begin{array}{l} \Delta = 1/5 \\ B = -1/5 \end{array}$

$T(n) = \frac{4^n}{5} + (-1)^{n-1} \in O(4^n)$

b) $T(n) = 4T(n/2) + n^2 \rightarrow n$ potencia de 2 $\rightarrow n = 2^k \rightarrow k = \log_2 n$

$$T(2^k) = 4T(2^k/2) + 2^{2k}$$

Cambiamos de variable $C(k) = T(2^k) \rightarrow C(k) = 4C(k-1) + 4^k \rightarrow$ Ahora se pueden hallar el polinomio característico

Δ a partir del anexo:

$$T(1) = T(2^0) \rightarrow k=0 \rightarrow C(0) = 1$$

$$T(2) = T(2^1) \rightarrow k=1 \rightarrow C(1) = 8$$

$$C(k) - 4C(k-1) - 4^k$$

$$\frac{(k-1)'}{(k-1)^2} \rightarrow x^2 - 8x + 16 \quad \begin{cases} r = 4 & m=2 \\ r = 1 & m=1 \end{cases} \rightarrow 4^k(\Delta k + B) \quad \begin{cases} k=0 \rightarrow B = 1 \\ k=1 \rightarrow 4(\Delta + B) = 8 \end{cases}$$

$$C(k) = 4^k(k+1) \rightarrow k4^k + 4^k \rightarrow k(2^k)^k + (2^k)^k$$

$$T(2^k) \rightarrow k(2^k)^k + (2^k)^k$$

$$T(n) \rightarrow \log_2 n \cdot n^k + n^k \in O(n \log_2 n)$$

$$C) T(n) = ZT(n/2) + n \log_2 n, \quad n \text{ potencia de } 2 \rightarrow n = 2^k \rightarrow k = \log_2 n$$

$$T(2^k) = ZT(2^{k-1}) + 2^k \log_2 2^k$$

$$\text{Cambio de variable } T(2^k) = C(k)$$

$$k = \log_2 2^k$$

$$C(k) = ZC(k-1) + 2^k \log_2 2^k$$

$$C(k) - ZC(k-1) = k2^k$$

$$\underbrace{(k-2) \cdot (k-2)^2}_{\substack{\text{C}(k-1) \\ \text{Z}C(k-1)}}$$

$$(k-2)^3 \rightarrow r=2, m=3$$

$$\rightarrow 2^k (\Delta + Bk + Ck^2) \rightarrow$$

Como solo queremos calcular la complejidad

no es necesario calcular Δ, B, C (los constantes no importan) solo los valores a

desconocer el cambio de variable $k = \log_2 n$

$$2^k (\Delta + B \log_2 n + C \log_2^2 n)$$

$$\Delta n + Bn \log_2 n + Cn \log_2^2 n$$

$$\in O(n \log_2^2 n)$$

$$d) T(n) = 3T(n/2) + Sn + 3, \quad n \text{ potencia de } 2 \rightarrow n = 2^k \rightarrow k = \log_2 n$$

$$T(2^k) = 3T(2^{k-1}) + S2^k + 3$$

$$\text{Cambio de variable } T(2^k) = C(k)$$

$$k = \log_2 2^k$$

$$C(k) = 3T(k-1) + S2^k + 3$$

$$C(k) - 3T(k-1) - S2^k - 3 = 0$$

$$\underbrace{(k-3) \cdot (k-2) \cdot (k-1)}_{\substack{\text{C}(k-1) \\ \text{Z}T(k-1) \\ \text{S}2^k}} = 0$$

$$\begin{cases} r_1 = 3 & m=1 \\ r_2 = 2 & m=1 \\ r_3 = 1 & m=1 \end{cases}$$

$$\begin{aligned} & 3\Delta + 2^k B + C \rightarrow \text{Deshacemos el cambio de variable} \\ & 3^{\log_2 n} + 2^{\log_2 n} + C \\ & n^{\log_2 3} + n + C \in O(n^{\log_2 3}) \end{aligned}$$

$$e) T(n) = ZT(n/2) + \log_2 n \rightarrow n \text{ es potencia de } 2 \rightarrow n = 2^k \rightarrow k = \log_2 n$$

$$T(2^k) = ZT(2^{k-1}) + \log_2 2^k$$

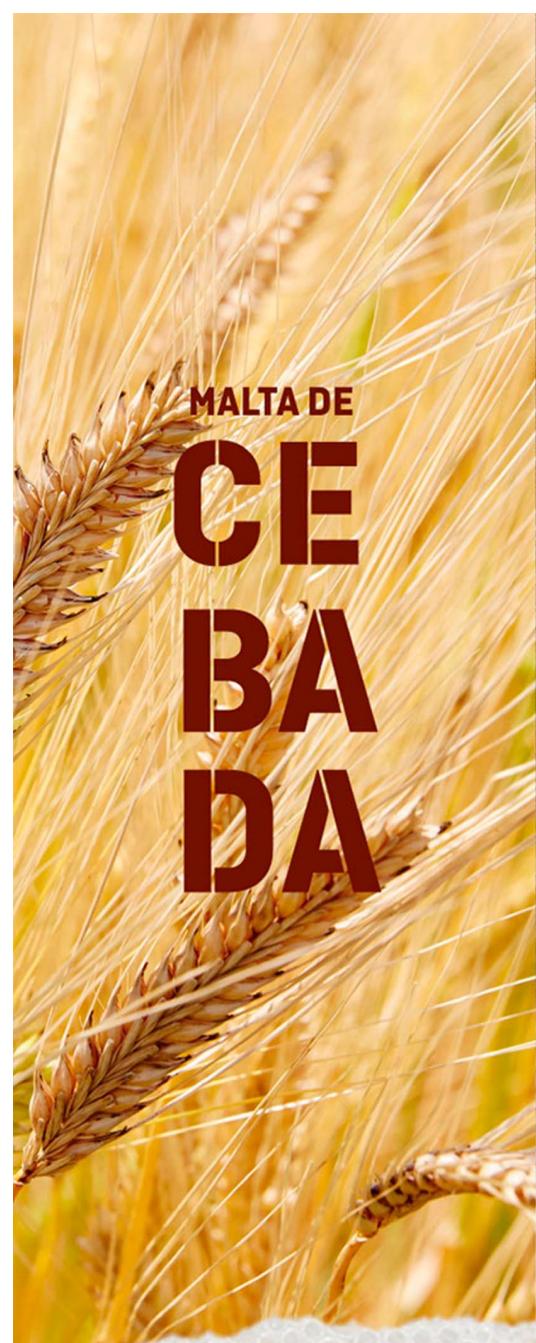
$$\text{Cambio de variable } T(2^k) = C(k) \rightarrow k = \log_2 2^k$$

$$C(k) = ZT(k-1) + \log_2 2^k$$

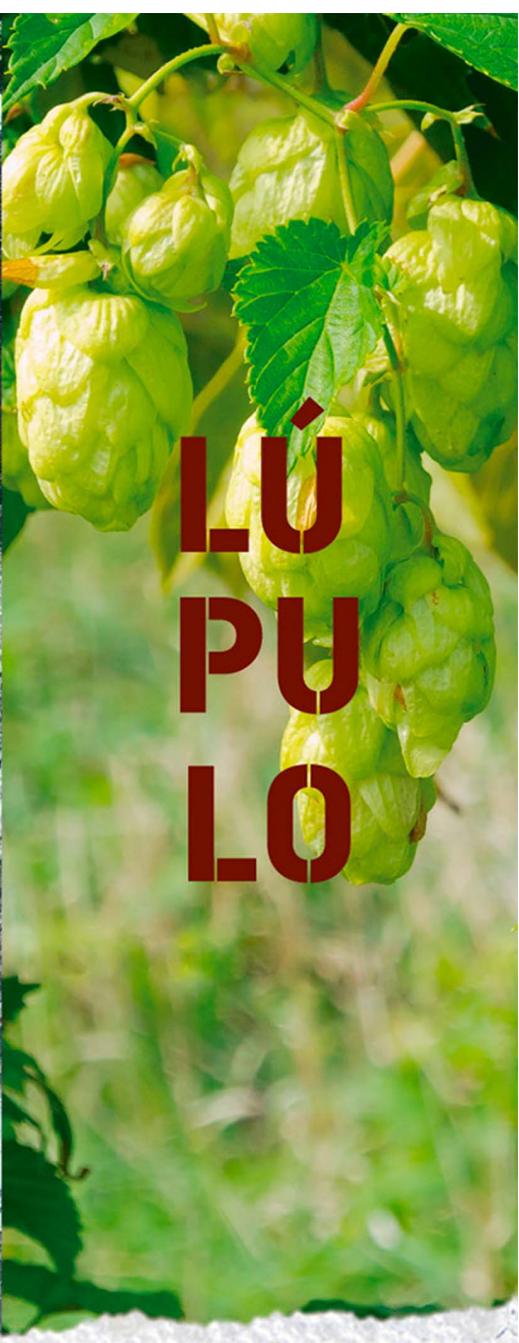
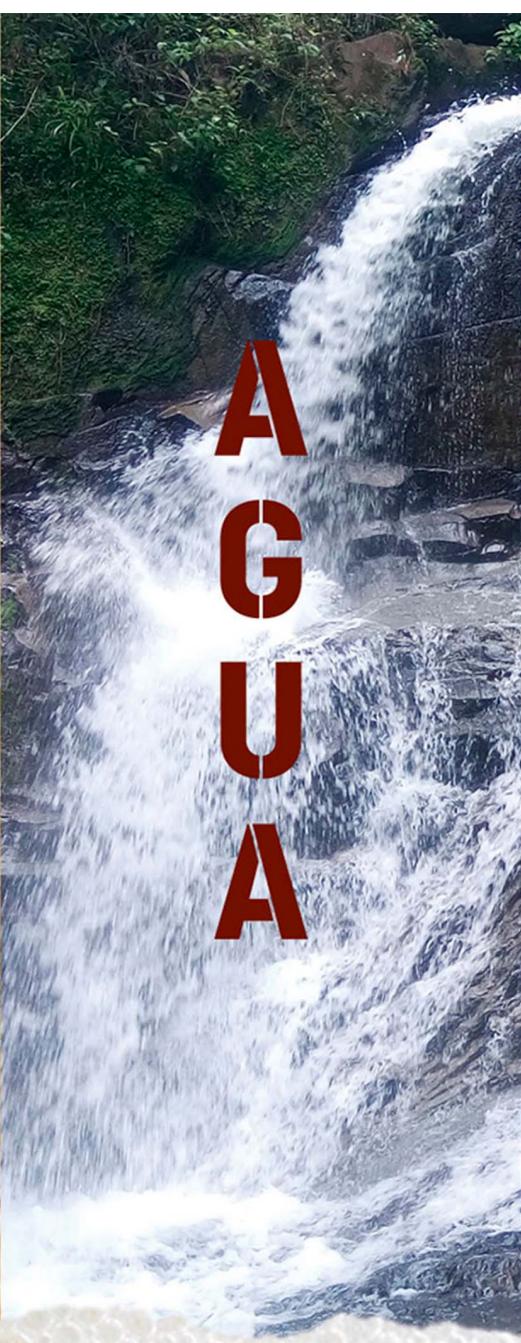
$$C(k) - ZT(k-1) - \log_2 2^k = 0$$

$$\underbrace{(k-2) \cdot (k-1)^2}_{\substack{\text{C}(k-1) \\ \text{Z}T(k-1) \\ \log_2 2^k}} \begin{cases} r_1 = 2 & m=1 \\ r_2 = 1 & m=2 \end{cases}$$

$$\begin{aligned} & 2^k (\Delta) + 1(Bk + C) \rightarrow \text{Deshacemos el cambio de variable} \\ & 2^{\log_2 n} \Delta + B \log_2 n + C \\ & n^{\log_2 2} \Delta + B \log_2 n + C \\ & \Delta n + Bn \log_2 n + C \in O(n) \end{aligned}$$



MALTA DE
**CÉ
BÀ
DA**



Tu **CERVEZA**
se hace a partir de ingredientes naturales.

Descubre más en cerveceros.org



**DISFRUTA
DE LO NATURAL.**

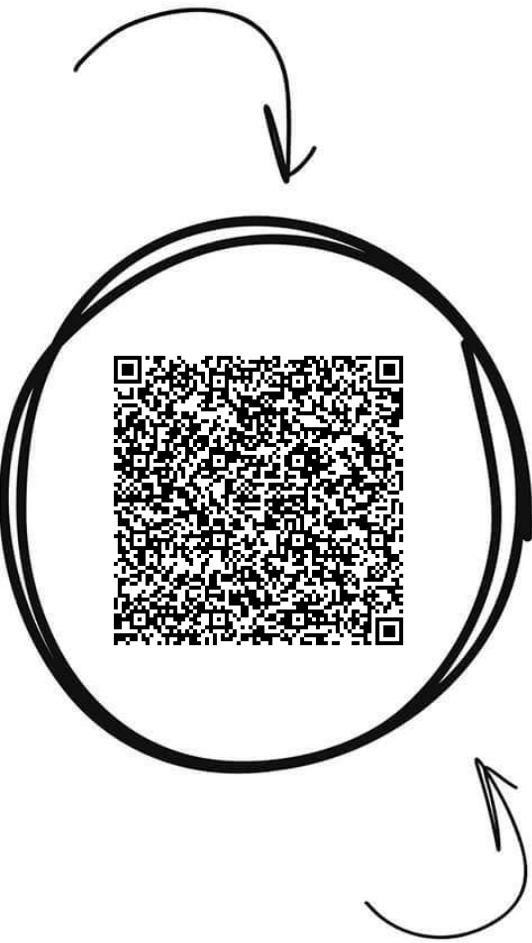
Cerveceros de España recomienda el consumo responsable



Análisis y diseño de algoritmos



Comparte estos flyers en tu clase y consigue más dinero y recompensas



- 1 Imprime esta hoja
- 2 Recorta por la mitad
- 3 Coloca en un lugar visible para que tus compis puedan escanar y acceder a apuntes
- 4 Llévate dinero por cada descarga de los documentos descargados a través de tu QR

Banco de apuntes de la



$$g) \begin{cases} T(n) = T(n-1) + 2T(n-2) - 2T(n-3) & \text{si } n > 2 \\ T(n) = 9n^2 - 15n + 106 & \text{si } n = 0, 1, 2 \end{cases} \rightarrow \begin{cases} T(n) - T(n-1) - 2T(n-2) + 2T(n-3) = 0 & \text{si } n > 2 \\ T(n) = 9n^2 - 15n + 106 & \text{si } n = 0, 1, 2 \end{cases}$$

$$\underbrace{T(n) - T(n-1) - 2T(n-2) + 2T(n-3)}_{\text{Usando la técnica de operaciones elementales}} = 0$$

$$n^3 - 1n^2 - 2n + 2 = 0$$

$$\begin{array}{c|cccc} 1 & 1 & -1 & -2 & 2 \\ & \downarrow & 1 & 0 & -2 \\ \hline 1 & 0 & -2 & 0 \end{array}$$

$$\left\{ \begin{array}{l} r=1 \\ R_1 = 1 \\ R_2 = -1 \\ R_3 = -1 \end{array} \quad \begin{array}{l} m=1 \\ M_1 = 1 \\ M_2 = 1 \\ M_3 = 1 \end{array} \right\}$$

$$1^n \Delta + (\sqrt{-1})^n B + (-\sqrt{-1})^n C \rightarrow \text{usando la otra ecuación}$$

Hallar Δ, B y C

$$\begin{aligned} n=0 &\rightarrow T(0) = 106 \rightarrow \Delta + B + C = 106 \\ n=1 &\rightarrow T(1) = 100 \rightarrow \Delta + \sqrt{-1}B - \sqrt{-1}C = 100 \\ n=2 &\rightarrow T(2) = 112 \rightarrow \Delta + 2B + 2C = 112 \end{aligned} \quad \left\{ \begin{array}{l} \text{Metemos en la calculadora} \\ \text{y resolvemos} \end{array} \right.$$

$$\Delta = 100$$

$$B = 3 \rightarrow 100 + 3(\sqrt{-1}^1 + (-\sqrt{-1})^1)$$

$$C = 3$$

$$\in O((\sqrt{-1})^n)$$

$$g) T(n) = 2T(n/4) + n^{3/2} \quad \text{si } n > 4, \quad n \text{ es potencia de } 4 \rightarrow n = 4^k \rightarrow k = \log_4 n$$

$$T(4^k) - 2T(4^{k-1}) - 4^{k/2}$$

$$T(4^k) = C(k) \rightarrow \underbrace{C(k) - 2C(k-1)}_{k = \log_4 4^k} - 4^{k/2}$$

$$1 \cdot 4^k - 2 \cdot 4^{k-1} - 4^{k/2}$$

$$(k-2) \cdot (k-2)^{1/2} - 2^k$$

$$\left\{ \begin{array}{l} r=2 \\ m=2 \end{array} \rightarrow 4^k (\Delta + Bk) \right\}$$

Como solo queremos calcular la complejidad
no es necesario calcular Δ y B (los constantes
no importan) fijarnos en
desacelerar el cambio de
variable

$$\rightarrow 2^{k/2} (\Delta + B \log_4 n)$$

$$2^{k/2} = 4^{k/4}$$

$$n^{3/2} (\Delta + B \log_4 n)$$

$$n^{3/2} \Delta + B n^{3/2} \log_4 n \in O(n^{3/2} \log_4 n)$$

$$h) T(n) = 4T(n/3) + n^2, \quad n \text{ potencia de } 3 \rightarrow n = 3^k \rightarrow k = \log_3 n$$

$$T(3^k) = 4T(3^{k-1}) + 3^k$$

Cambio de variable $T(3^k) = C(k)$

$$C(k) = 4C(k-1) + 3^k$$

$$C(k) - 4C(k-1) - 3^k$$

$$\frac{(k-1)!}{(k-4)!} \cdot \frac{(k-4)!}{(k-9)!} \left\{ \begin{array}{l} r_1 = 4 \quad m=1 \\ r_2 = 9 \quad m=1 \end{array} \right. \rightarrow 4^k \Delta + 9^k B \rightarrow$$

Deshacerse del cambio

$$\Delta 4^k \log_3 4 + B 9^k \log_3 9$$

$$\Delta n^{\log_3 4} + B n^{\log_3 9} \in O(n^2)$$

#DíasNaranjas

Haz frente al otoño con nuestros vitamínicos descuentos.
Del 2 al 15 de octubre



Hasta un
-40%
de descuento

5. Para cada una de las siguientes recurrencias determine su orden de complejidad usando el Teorema Maestro. Una vez realizado esto, encuentre las ecuaciones exactas para cada una de ellas (no es necesario calcular los valores de los coeficientes que dependen de las condiciones iniciales):

- $T(n) = 4T(n/2) + n^2$
 - $T(n) = 2T(n/2) + n \log n$
 - $T(n) = 3T(n/2) + 5n + 3$
 - $T(n) = 2T(n/2) + \log n$
 - $T(n) = 5T(n/2) + (n \log n)^2$
 - $T(n) = 2T(\sqrt{n}) + \log n$
- (No tiene caso base)
(la recursión reduce)

$$a) T(n) = \frac{4}{2}T\left(\frac{n}{2}\right) + \frac{n^2}{2}$$

$$n^{\log_2 4} = n^2 = n^2 \rightarrow \text{Estando en el caso medio}$$

$$f(n) = n^2 \in \Theta(n^2) = \Theta(n \cdot \log n)$$

Completa la
condición

$$T(n) \in \Theta(n \cdot \log n)$$

Ingeniería gráfica equilibradas
e igualdad aplicar caso 3

$$\lim_{n \rightarrow \infty} \frac{n \log n}{n^2} = \lim_{n \rightarrow \infty} \frac{\log n}{n} = \frac{-1}{n} \rightarrow \lim_{n \rightarrow \infty} \frac{1}{n^{2-1}} = \frac{1}{n} = \frac{1}{\infty} = 0 \rightarrow n \log n \in \Omega(n^2)$$

No Podemos DEMOSTRARLO

$$b) T(n) = \frac{2}{2}T\left(\frac{n}{2}\right) + n \log n \xrightarrow{\text{Por inducción}} f(n) = n \log n \in \Theta(n \cdot \log n) \rightarrow T(n) \in \Theta(n \cdot \log n)$$

$n^{\log_2 2} = n$
Caso 2, se
diferencia en $\log n$

$$c) T(n) = \frac{3}{2}T\left(\frac{n}{2}\right) + \frac{5n+3}{2} \xrightarrow{\substack{\lim_{n \rightarrow \infty} \frac{n^{\log_2 3}}{5n+3} \\ n^{\log_2 3} > 5n+3}} \frac{(\log_2 3 \cdot n)^{(\log_2 3)-1}}{5} \rightarrow \infty \quad \begin{array}{l} n^{\log_2 3} \text{ es claramente} \\ \text{peor que } f(n), \text{ caso 1} \end{array} \rightarrow T(n) \in O(n^{\log_2 3})$$

$$d) T(n) = \frac{2}{2}T\left(\frac{n}{2}\right) + \frac{\log n}{2} \xrightarrow{\substack{\lim_{n \rightarrow \infty} \frac{n}{\log n} \\ n^{\log_2 2} = n}} \infty \quad \begin{array}{l} n \text{ es claramente peor} \\ \text{que la función, caso 1} \end{array} \rightarrow T(n) \in O(n)$$

$$e) T(n) = \frac{5}{2}T\left(\frac{n}{2}\right) + (\log n)^2 \xrightarrow{\substack{\lim_{n \rightarrow \infty} \frac{n^{\log_2 5}}{(\log n)^2} \\ n^{\log_2 5} > (\log n)^2}} \frac{(\log_2 5 \cdot n)^{(\log_2 5)-1}}{2} = \infty \quad \begin{array}{l} n^{\log_2 5} \text{ es claramente mayor} \\ \text{que } f(n), \text{ caso 1} \end{array} \rightarrow T(n) \in O(n^{\log_2 5})$$

$$\begin{aligned} f) T(n) &= 2T(\sqrt{n}) + \log n \\ &\xrightarrow{n = z^k} T(z^k) = 2T(z^{k/2}) + \log z^k \\ &\text{Cambio de variable } T(2^k) = C(k) \\ &C(k) = \frac{2}{2}T\left(\frac{k}{2}\right) + \log z^k \xrightarrow{\substack{\lim_{k \rightarrow \infty} \frac{K}{\log z^k} \\ K < \log z^k}} \frac{K}{\log z} = \frac{1}{\log z} \end{aligned}$$

El límite no da un número,
por lo que se difieren en
una constante $\log n$ y salimos

en el caso 2. De hecho, $K = 0$ $\rightarrow C(k) \in O(1/k \log k) \rightarrow$ Deshacemos el cambio
 $T(k) \in O(\log n \cdot \log(\log n))$

6. Dado un valor numérico x y un valor natural $n \geq 0$ potencia de 2, queremos escribir un algoritmo para calcular x^n . Se pide:

- Diseñar un algoritmo para resolver el problema mediante un enfoque iterativo, e indicar su complejidad en términos del número de multiplicaciones ejecutadas.
- Diseñar un algoritmo recursivo, de manera que sea más eficiente que el enfoque anterior.

a) $\text{int res} = 1$
 $\text{for (int } i=0; i < n; i++) \text{ res} = \text{res} \cdot x$

Número de multiplicaciones: $n-1$



7. Calcular mediante expansión de recurrencias el tiempo de ejecución y determinar la complejidad en los casos mejor y peor del siguiente algoritmo:

```
public static int recursiva (int n){
    if ( n <= 1 )
        return 1;
    else
        return recursiva (n-1)+recursiva(n-1);
};
```

Tamaño de entrada n .
 Contarás todas las operaciones
 (5 elementales + 2 invocaciones).
 No hay caso mejor o peor

$$T(n) = 7 + 2T(n-1) \rightarrow \text{Resolver}$$

Contarás todas las operaciones
 (en este caso peor caso)

$$T(n) - 2T(n-1) - 7 = 0$$

$$(n-2)(n-1) \begin{cases} c_1 = 2 & m-1 \\ c_2 = 1 & m-1 \end{cases} \rightarrow 1^n A + 2^n B$$

$$T(n) \in O(2^n)$$

8. Dado un array A de n números enteros distintos que se encuentra ordenado, y un número entero x , diseñar un algoritmo que determine (devuelva un booleano) si existen dos elementos de A cuya suma sea exactamente x .

- a) ¿Puedes proporcionar la complejidad exacta de tu algoritmo?
 b) Existe una solución sencilla que es de orden lineal. ¿La has encontrado?

bodean suma (int []a, int x)

```
int i=0
while(!encontrado && i<a.length-1){
    int j = i+1
    while(!encontrado && j<a.length){
        encontrado = (a[i]+a[j]==x)
        j++
    }
    i++
}
return encontrado
```



EXÁMEN BLOQUE 1 2019

Ejercicio 1: Análisis de complejidad (3 puntos)

Puntos:

- a) Resolver la siguiente ecuación de recurrencia y dar el orden de complejidad:

$$T(n) = 7T(n/2) - 6T(n/3), \text{ con } T(n) = (5/2)n^2 + (5/2)n + 6 \text{ para } n \leq 2$$

- b) Teniendo en cuenta el teorema Maestro decir, para cada caso, una ecuación de recurrencia $T(n) = aT(n/b) + f(n)$ válida, cuya solución sea:

- a. $T(n) \in \Theta(n^2)$
- b. $T(n) \in \Theta(n \log n)$
- c. $T(n) \in \Theta(n^2 \log n)$

Ejemplo: $T(n) \in \Theta(n)$, entonces una solución es: $a=2, b=2, d=0.5: T(n) = 2T(n/2) + \sqrt{n}$

- c) 1. Escribir una función que se sitúe entre n y $\log n$.
 2. Escribir una función que supera a $\log n$ pero esté muy cerca de ella.

a) Primero haremos el caso general.

Con la segunda ecuación: $\begin{cases} T(0) = 6 \\ T(1) = 11 \end{cases}$

$$T(n) = 7T(n/2) - 6T(n/3)$$

$$T(n) - 7T(n/2) + 6T(n/3) = 0$$

$$6n^3 - 7n^2 + 1/n = 0 \quad \begin{cases} r_1 = 0 & m=1 \\ r_2 = 1 & m=1 \\ r_3 = 1/6 & m=1 \end{cases} \rightarrow \Delta n + \frac{B}{8n}$$

$$\begin{cases} n=0 \rightarrow \Delta+B = 6 \\ n=1 \rightarrow \Delta + \frac{B}{6} = 11 \end{cases}$$

$$\begin{aligned} \Delta &= 12 \\ B &= -6 \end{aligned}$$

$$\text{Sol: } 12n^3 + \frac{-6}{6n}$$

b)

$n^{63/4}$ es claramente peor.

I) $T(n) = \underbrace{4T(n/2)}_{n^{63/4}} + 3 \rightarrow$ por b que estamos en el caso 1 $\rightarrow T(n) \in O(n^2)$

II) $T(n) = \underbrace{2T(n/2)}_{n^{63/4}} + n \rightarrow$ La complejidad es la misma, así que estamos en el caso 2, se diferencian en logn. $\rightarrow T(n) \in O(n \log n)$

III) $T(n) = \underbrace{4T(n/2)}_{n^{63/4}} + n^2 \rightarrow$ La complejidad es la misma, así que estamos en el caso 2, se diferencian en logn. $\rightarrow T(n) \in O(n^2 \log n)$

c) $\log n + \frac{1}{2}$? No se

Ejercicio 2: Especificación (1 punto)

Puntos:

Especificar un algoritmo que tiene como entrada un vector de valores enteros positivos y produce como salida la cantidad de números impares que hay en el vector. Ejemplo: Entrada v[] = [2, 1, 5, 4, 23, 6, 37] Salida: 4.

Piccard: $(v.\text{length} > 0) \wedge (\forall i \in (0, v.\text{length}), i > 0)$

fun imp: $(\# v) \text{ int}, \uparrow n$

Postcond: $n = \sum_{i=0}^{\# v} \{v[i] \text{ es impar}\}$

5€ DE BIENVENIDA

Ejercicio 4: Divide y Vencerás (6 puntos)

Puntos:

Sea un array ordenado estrictamente creciente, de números naturales, con $A[0] = k$. Se pide obtener el menor elemento superior a k que no está en el array. Si están todos, el elemento que falta es el número siguiente.

Ejemplos:

Input: $A[] = [21, 22, 24, 30, 36, 37]$

Output: El número que falta es 23

Input: $A[] = [21, 22, 23, 24, 25, 26]$

Output: El número que falta es 27

- Dar la especificación Pre-Post para este algoritmo (1 punto)
- Desarrollar un algoritmo de fuerza bruta (1 punto)
- Calcular su complejidad de forma razonada (0,3 puntos)
- Desarrollar un algoritmo por la técnica de Divide y Vencerás (3 puntos)
- Calcular su complejidad mediante el teorema Maestro (0,7 puntos)

Nota: La complejidad del apartado e) debe ser sensiblemente inferior a la calculada en el apartado b).

```
b) res = -1;
for (int i = V[0]; i < V.length - 1; i++) {
    if (V[i + 1] != V[i] + 1) {
        res = i;
    }
}
if (res == -1) {
    res = V[V.length - 1] + 1;
}
```

d y c)

```
// Sea un array ordenado estrictamente creciente, con A[0] = k > 0 obtener el menor elemento superior a k
// que no está en el array. Si están todos, el que falta es el siguiente.
// Ejemplos: [21, 22, 23, 28, 36, 37] → El que falta es el 24
// Ejemplos: [21, 22, 23] → El que falta es el 24

public static void main(String[] args) {
    int lista[] = { 1, 2, 3, 4, 5, 6, 8, 9 };
    int lista2[] = { 1, 2, 3, 23, 24, 25, 26 };

    // llamamos a la función recursiva con los parámetros iniciales
    System.out.println(falta(lista2, 0, lista2.length - 1));
}

public static int falta(int[] lista, int inicio, int fin) {
    //Cálculamos los parámetros, incluyendo mit, que irá cambiando en cada iteración.
    int res = -1;
    int mit = inicio + (fin - inicio) / 2;
    int primer = lista[0];

    //En el primer if contemplamos el caso del segundo ejemplo
    if (mit < lista.length) {
        //Estos son los criterios, cuando ya tenemos localizado el último elemento bien colocado.
        if (fin == inicio + 1) {
            //Si no cumple el caso base, hacemos la llamada recursiva con medio array, actualizando los parámetros.
            if (lista[mit] == mit + primer) {
                //Para que funcione, importante asignar la función a la variable que devolveremos.
                res = falta(lista, inicio, mit - 1);
            } else {
                res = falta(lista, inicio, mit - 1);
            }
        } else {
            //Esto es lo que devolvemos en el caso base
            res = lista[mit] + 1;
        }
    } else {
        res = lista[lista.length - 1] + 1;
    }

    return res;
}
```

a) **Recorrido:** $V[i \in (0, \text{length})], V[\text{length}] > 0, V[i] < V[i+1], V[i+1] > 0$

c) Contarás los accesos al array. Calculando para el peor caso

$$\frac{1 + \sum_{i=1}^n + 1}{\text{acceso}} = \text{punto}$$

log a x directamente a la ecuación de recurrencia por caso

$$T(n) = 3 + T(n/2) \rightarrow \text{Teorema maestro reducido}$$

$$T(n) = \frac{1}{a} T(\frac{n}{b}) + \frac{c}{b} \rightarrow a = b^d \rightarrow T(n) \in \Theta(1 \cdot \lg n)$$

Basando mejor que el caso dividir

mit F

$$\frac{n^{\frac{1}{2}}}{n \lg n + n} \leftarrow f(n) > n^{\frac{1}{2}}$$

$$T(n) = 9T(n/3) + n \lg n + n$$

$$T(3^k) = 9T(3^{k-1}) + 3^k \lg 3^k + 3^k \rightarrow n = 3^k \rightarrow k = \lg_3 n$$

$$T(3^k) = C(k)$$

$$C(k) = 9T(k-1) + 3^k \lg 3^k + 3^k$$

$\underbrace{(k-1)}_{(k-1)} \quad \underbrace{(k-1)}_{(k-1)}$

$$r_1 = 3 \quad m = 3$$

$$r_2 = 9 \quad m = 1$$

WUOLAH