

Algoritmos y Estructuras de Datos Cursada 2024

Análisis de Algoritmos

1.- Ordene las siguientes funciones: \sqrt{n} , n , 3^n , n^2 , cte, 2^n , $\log_2^2(n)$, $\log_3(n)$, $\log_2(n)$ según su velocidad de crecimiento.

1- cte, $\log_2(n)$, $\log_3(n)$, $\log_2^2(n)$, \sqrt{n} , n , n^2 , 2^n , 3^n

2.- Exprese de qué orden es el siguiente fragmento de código

```
for (int j = 4; j < n; j=j+2) {
    val = 0;
    for (int i = 0; i < j; ++i) {
        val = val + i * j;
        for (int k = 0; k < n; ++k){
            val++;
        }
    }
}
```

(a) $O(n \log n)$

(b) $O(n^2)$

(c) $O(n^2 \log n)$

(e) $O(n^3)$

2- Paso	J	El segundo FOR se ejecuta J veces.	Paso	K
(0)	4		(0)	0
(1)	6		(1)	1
(2)	8		(2)	2
(K)	$4+2.K$		(K)	K

$4+2.K = n-1$
 $K = \frac{n-4}{2}$
 $K = \frac{(n-4)}{2}$

$$\sum_{j=4}^{(n-4)/2} \left(c_1 + \sum_{i=1}^j [c_2 + \sum_{k=1}^n c_3] \right) = \sum_{j=4}^{(n-4)/2} \left(c_1 + \sum_{i=1}^j [c_2 + n c_3] \right) = \sum_{j=4}^{(n-4)/2} c_1 + j \cdot n =$$

$c_2 + c_3 = c_4$

$$= \frac{n \cdot c_1 \cdot ((n-4)/2) \cdot ((n-4)/2 + 1)}{2} = \frac{n \cdot c_1 \cdot n^2 - 6n + 8}{8} \therefore O(n^3)$$

3.- Suponga que dispone de un algoritmo A, que resuelve un problema de **tamaño n**, y su función de tiempo de ejecución es $T(n) = n * \log(n)$. Este algoritmo se ejecuta en una computadora que procesa **10.000 operaciones** por segundo. Determine el **tiempo** que requerirá el algoritmo para resolver un problema de tamaño **n=1024**.

3- $T(n) = n * \log(n) : 10.000 \text{ op / seg.}$

$T(1024) = 1024 * \log(1024) = 1024 * 10 = 10240 \text{ op} \Rightarrow 10240 / 10000 = 1,024 \text{ segundos.}$

4.- ¿Cuál es el resultado de la siguiente sumatoria?

$$\sum_{i=3}^8 n \cdot i =$$

- a) $(8-3+1) \cdot n$
- b) $(8-3+1) \cdot i \cdot n$
- c) **$33n$**
- d) $5n$
- e) $8 \cdot i$
- f) Ninguna de las otras opciones

$$4.- \sum_{i=3}^8 n \cdot i = n \cdot \left(\sum_{i=1}^8 i - \sum_{i=1}^2 i \right) = n \cdot \left(\frac{8 \cdot (8+1)}{2} - \frac{2 \cdot (2+1)}{2} \right) = 33n =$$

5.- ¿Cuál de las siguientes sentencias es correcta, según la definición vista en clase?

- a) n^2 es $O(n^2)$
- b) n^2 es $O(n^3)$
- c) n^2 es $O(n^2 \log n)$
- d) Opciones a y b
- e) **Opciones a, b y c**
- f) Ninguna de las otras opciones

5- Opción e) : n^2 crece más lentamente que n^3 y $n^2 \cdot \log(n)$, entonces n^2 más lento que las otras $f(x)$.

6.- Dado el siguiente algoritmo

```
void ejercicio5 (int n) {
    if (n >= 2) {
        2 * ejercicio5 (n/2);
        n = n/2;
        ejercicio5 (n/2);
    }
}
```

i) Indique el $T(n)$ para $n \geq 2$

- (a) $T(n) = d + 3 \cdot T(n/2)$
- (b) $T(n) = d + 2 \cdot T(n/2) + T(n/4)$
- (c) **$T(n) = d + T(n/2) + T(n/4)$**
- (d) $T(n) = d + T(n/2) + T(n/2)$
- (e) $T(n) = d + T(n/2) + T(n/2) + T(n/4)$

7.- Dada la recurrencia

$$T(n) = \begin{cases} 1 & \text{para } n \leq 1 \\ T(n/3) + c & \text{para } n > 1 \end{cases}$$

i) ¿Cómo se reemplaza $T(n/3)$, considerando $n/3 > 1$?

- (a) $T(n/3) + c$
- (b) Ninguna de las otras opciones
- (c) $T(n/3) + 1$
- (d) **$T(n/3/3) + c$**
- (e) $T(n/3/3) + 1$

ii) Desarrolle la función $T(n)$

$$T(n) = \begin{cases} 1 & n=1 \\ T(n/3) + c & n>1 \end{cases}$$

Con $n>1$

Paso 1: $T(n) = T(n/3) + c$ Paso i: $T(n) = T(n/3^i) + i \cdot c$

Paso 2: $T(n) = T(n/9) + c + c$ $n/3^i = 1 \Rightarrow n = 3^i$

Paso 3: $T(n) = T(n/27) + 2c + c$ $\Rightarrow i = \log_3 n$

Reemplazo i y n:

ii) $T(n) = T(3^i/3^i) + \log_3(n) \cdot c = T(1) + \log_3(n) \cdot c = 1 + \log_3(n) \cdot c \therefore O(\log_3(n))$

i) Opción d)

8.- Considere el siguiente fragmento de código:

```
int count = 0; int n = a.length;
for (int i = 0; i < n; i+=n/2) {
    for (int j = 0; j < n; j++) {
        a[j]++;
    }
}
```

Este algoritmo se ejecuta en una computadora que procesa 100.000 operaciones por cada segundo. Determine el tiempo aproximado que requerirá el algoritmo para resolver un problema de tamaño $n=1000$.

- (a) 0,01 seg
- (b) 0,1 seg
- (c) 1 seg
- (d) Ninguna de las opciones anteriores.

$$8 - \sum_{i=1}^2 \sum_{j=1}^n c = 2nc \therefore O(n) \Rightarrow 100.000 \text{ p/seg}$$

$$T(1000) = 1000 \text{ operaciones}$$

$$100.000 \text{ op} \text{ --- } 1 \text{ seg}$$

$$1000 \text{ op} \text{ --- } 1000 / 100.000 = 0,01 \text{ seg} \Rightarrow \text{Opción a)}$$

9.- Considere la siguiente recurrencia:

$$T(1) = 4$$

$$T(n) = 2T(n/2) + 5n + 1 \quad (n \geq 2)$$

¿Cuál es el valor de $T(n)$ para $n = 4$?

- (a) 51
- (b) 38
- (c) 59
- (d) 79
- (e) Ninguna de las opciones anteriores

$$\begin{aligned}
 9.- T(4) &= 2T(4/2) + 5 \cdot 4 + 1 \\
 T(4/2) &= 2 \cdot [2T(2/2) + 5 \cdot 2 + 1] + 20 + 1 \\
 T(2) &= 4T(1) + 20 + 2 + 2 \cdot 1 = 4 \cdot 4 + 43 = 59 \Rightarrow \text{Opción c) }
 \end{aligned}$$

10.- Expresar la función $T(n)$ del siguiente segmento de código:

```

public static void ejercicio (int n) {
    int x = 0;
    int j = 1;
    while (j <= n) {
        for (int i = n*n; i >= 1; i = i - 3)
            x = x + 1; j = j * 2;
    }
}

```

- (a) $T(n) = (1/3) * n^2 + \log_2(n)$
 (b) $T(n) = n^2 + (1/3) * \log_2(n)$
 (c) $T(n) = (1/3) * \log_2(n)$
 (d) $T(n) = (1/3) * n^2 * \log_2(n) + \log_2(n)$

Paso	j	Paso	i
0	1	0	n^2
1	2	1	$(n^2-3)^2$
2	4	2	$(n^2-3)^2$
K	2^K	K	$(n^2-3)^2$

$$2^K = n \quad K = \log_2 n$$

$$n^2 - 3K = 1 \quad K = \frac{n^2 - 1}{3}$$

$$T(n) = \sum_{j=0}^{\log_2(n)} \cdot \sum_{i=0}^{n^2/3} c = \sum_{j=0}^{\log_2(n)} \cdot n^2/3 = 1/3 \cdot n^2 \cdot \log_2(n) \Rightarrow \text{Opción a)}$$

11.- ¿Cuál es el tiempo de ejecución del siguiente método?

```

void fun(int n, int arr[]){
    int i = 0;
    j = 0;
    for (; i < n; ++i)
        while (j < n && arr[i] < arr[j])
            j++;
}

```

M- Paso	1	Paso	J	
0	0	0	0	La variable J no se inicializa para cada valor de i
1	1	1	1	
2	2	2	2	
K	K	K	K	

$$T(n) = \sum_{i=1}^n \sum_{j=1}^n c_1 + c_2 = n^2 \cdot c_1 + c_2 = O(n^2)$$

\downarrow
 $c_1 + c_2$

12.- ¿Cuál es el valor que retorna el método fun1?

```
int fun1 (int n) {
    int i, j, k, p, q = 0;
    for (i = 1; i < n; ++i) {
        p = 0;
        for (j = n; j > 1; j = j/2)
            ++p;
        for (k = 1; k < p; k = k*2)
            ++q;
    }
    return q;
}
```

12. El método fun1 retorna q, que es el resultado de $(n-1) \cdot \log_2(\log_2(n))$. Por cada iteración del primer bucle, el tercer bucle se ejecuta $\log_2(\log_2(n))$ veces, ya que el segundo bucle se ejecuta $\log_2(n)$.

13.- ¿Cuál es el tiempo de ejecución del siguiente código?

```
void fun(int n) {
    for (int i = 0; i < n / 2; i++)
        for (int j = 1; j + n / 2 <= n; j++)
            for (int k = 1; k <= n; k = k * 2)
                System.out.print("AyED");
}

int main() {
    int n=8;
    fun(3);
}
```

13- TERCER FOR:	SEGUNDO FOR:	PRIMER FOR:
Paso K	Paso J	Paso 1
0 1	0 1	0 0
1 2	1 2	1 1
2 4	2 3	2 2
M 2^M	K K	K K
$2^M = n$	$K \cdot \log_2 n = n$	$K = n/2$
$M = \log_2(n)$	$K = n - n/2$	
	$K = n/2$	
$T(n) = \sum_{i=1}^{n/2} \sum_{j=1}^{(n/2)^1} \sum_{k=1}^{\log_2(n)+1} cte = \sum_{i=1}^{n/2} \sum_{j=1}^{(n/2)+1} (\log_2(n)+1) cte = \sum_{i=1}^{n/2} \frac{n}{2} + 1 \cdot [(\log_2(n)+1) \cdot cte] =$		
$T(n) = \frac{n}{2} \cdot \frac{n}{2} \cdot \log_2(n) \cdot cte \therefore O(n^2 \cdot \log_2(n))$		

14.- ¿Cuál es el tiempo de ejecución del siguiente código?

```
void fun(int a, int b) {
    // Consider a and b both are positive integers
    while (a != b) {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
}
```

14.- El for se ejecuta $O(\max(a \text{ div } 2, b \text{ div } 2))$ veces $\therefore O(\max(a, b))$

WHILE:			WHILE		
CANT PASOS	A	B	CANT PASOS	A	B
1	2	4	1	5	2
2	2	2	2	3	2
			3	1	2
			4	1	1

15.- ¿Cuál es el tiempo de ejecución del siguiente código?

```
void fun(int n) {
    for(int i=0; i*i<n; i++)
        System.out.print("AyED");
}
```

15-	Paso	i
	0	0
	1	1
	2	2
	k	k

$$\begin{aligned}
 k \cdot k &= n \\
 k^2 &= n \\
 k &= \sqrt{n}
 \end{aligned}
 \Rightarrow T(n) = \sum_{i=1}^{\sqrt{n}} . cte = \sqrt{n} \cdot cte \therefore O(\sqrt{n})$$

16.- ¿Cuál es el tiempo de ejecución del siguiente código?

```

int fun(int n) {
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j < n; j += i) {
            // Some O(1) task
        }
    }
}

```

Nota: Tenga en cuenta que $(1/1 + 1/2 + 1/3 + \dots + 1/n)$ se puede acotar con O

16- PRIMER FOR:	SEGUNDO FOR:
Paso i	Paso j
0 1	0 1
1 2	1 1+i
k k+1	2 1+2i
	3 1+3i
	k 1+ki

El FOR se ejecuta n+1 veces
 $k = n+1$

El FOR se ejecuta con $i=1$, n veces
 El FOR se ejecuta con $i=2$, $n/2$ veces
 El FOR se ejecuta con $i=n$, n/n veces

$$T(n) = \sum_{i=1}^{n+1} . \log(n) \cdot cte \Rightarrow (n+1) \cdot \log(n) \therefore O(n \cdot \log(n))$$

$\sum_{i=1}^n n/i = 1/2 \sum_{i=1}^n 1/i$
 Serie armónica es
 calada por n:
 Se resume en $\log(n)$