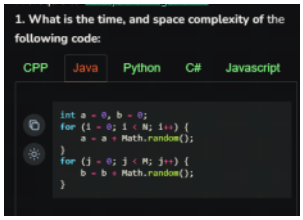
Analisis de algoritmos

miércoles, 12 de junio de 2024 16:21

https://www.geeksforgeeks.org/practice-questions-time-complexity-analysis/

ARN



$\sum_{i=0}^{N} c_0 = N.c_0$ | $\sum_{j=0}^{M} c_1 = M.c_1$

$T(N) + (N+M)$

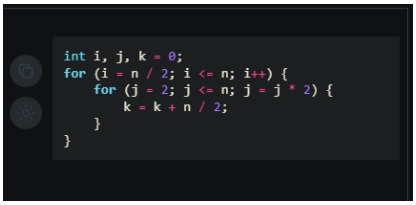
$O(1)$ ORDEN CONSTANTE NO DEPENDE DE LA ENTRADA

- SI $N = 0$, NO ENTRA AL FOR!
  ∴ ANALIZAMOS CON UN $N \geq 1$
- VAMOS DE ADENTRO HACIA AFUERA
- EL BUCLE INTERNO DEPENDE DEL EXTERNO i
  ∴ BUSCAMOS UN PATRON
  - $i = 0$ J VARIA DE 1 A N — N iteraciones
  - $i = 1$ J VARIA DE 2 A N — N-1 (Si es <2 no entraria para un $i = 1$)
  - $i = 2$ J VARIA DE 3 A N — N-2
  - $i = N-1$ J VARIA DE $N - N$

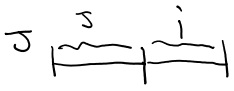
$$A = \sum_{i=0}^{N-1} (N-i)$$

$$N.\sum_{i=0}^{N-1} i = \frac{N.(N-1)}{2} = \frac{N^2+N-1}{2} \quad \therefore$$

$$O(N^2)$$



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



$j \times 2 = 2 \quad i = 1$

$2 \times 2 = 4 \quad i = 2$

$4 \times 2 = 8 \quad i = 3$

PARA LA ITERACION k VA SER 2^k DONDE $2^k \leq N$

RESOLVEMOS

$$k \leq \log_2(N)$$

EL BUCLE INTERNO SE EJECUTA $\log_2(N)$

EL BUCLE EXTERNO SE EJECUTA

$\frac{N}{2}$ VECES $\therefore$ ES $O(N)$ LINEAL

COMPLEJIDAD total $O(N) \times O(\log_2(N)) = O(N \log_2 N)$

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.

$$cte, \log_3(N), \log_2(N), \log_2^2(N), \sqrt{N}, N, N^2, 2^N, 3^N$$

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 logn) (b) O(n^3) (c) O(n^2 log n) (e) O(n^2)

$\frac{N}{2} - 4 \quad \sum_{i=0}^{\frac{N}{2}-4} \left[\sum_{k=0}^{N} K \right]$

ESTO SE VA A ITERAR POR

$\frac{N}{2} - 4 \quad \sum_{i=0}^{\frac{N}{2}-4} N \cdot K = \frac{N}{2} - 4 \cdot N = \frac{N^2}{2} - 4$

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

```
for (int j = 4; j < n; j=j+2) {
```

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 logn) (b) O(n^2) (c) O(n^2 log n) (e) O(n^3)

$$\left(\sum_{J=1}^{\frac{N-1}{2}}\left(\sum_{i=0}^{J}\left(\sum_{k=0}^{N} C_0\right)\right)\right)-\left(\sum_{J=1}^{3}\left(\sum_{i=1}^{4}\left(\sum_{k=0}^{N} C_0\right)\right)\right)$$

$$\left(\sum_{J=1}^{\frac{N-1}{2}} J \cdot N \cdot C_0\right)-\left(\sum_{J=1}^{3} 4 \cdot N \cdot C_0\right)$$

$$N \cdot C_0\left(\frac{\frac{N-1}{2}\cdot\left(\frac{N-1}{2}+1\right)}{2}\right)-\left(\frac{N \cdot C_0(3\times 4)}{2}\right) \quad (J)$$

$$N \cdot C_0\left(\frac{\frac{N-1}{2}\cdot\left(\frac{N-1}{2}+1\right)}{2}\right)-\left(\frac{N \cdot C_0(12)}{2}\right)$$

$$N \cdot C_0\left(\frac{(N-1)\cdot(N-1+2)}{8}\right)-\left(N \cdot C_0 \cdot 6\right)$$

$$N \cdot C_0\left(\frac{(N-1)\cdot(N+1)}{8}\right)-\left(N \cdot C_0 \cdot 6\right)$$

$$N \cdot C_0\left(\frac{N^2+N-N-1}{8}\right)-\left(N \cdot C_0 \cdot 6\right)$$

$$N \cdot C_0\left(\frac{N^2-1}{8}\right)-\left(N \cdot C_0 \cdot 6\right)$$

$$\left(\frac{N^3}{8} C_0-\frac{N}{8}\right)-(6\,NC_0)$$

$$O(N^3)$$

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**.

COMPUTADORA 10K x SEGUNDO

TAMAÑO DE ENTRADA N = 1024

$T(1024) = 1024 \times \log(1024) = 1024 \times 10 = 10240$

10000 — 1

10240 — x?

$\frac{1024}{60} \sim 17$ minutos

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

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

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

$$N \sum_{i=1}^{8} i = \frac{8 \times 9}{2} \Big) - \Big(N \sum_{i=1}^{2} i = \frac{2 \times 3}{2} \Big)$$

$$(36N) - (3N)$$

$$33N$$

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

Para determinar por qué la opción correcta es la E, repasemos cada afirmación usando la notación Big-O, que describe una cota superior para el crecimiento de una función.

Afirmación (a): n^2 es $O(n^2)$

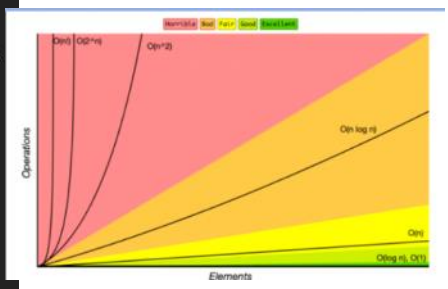
- Esta afirmación es verdadera porque n^2 es claramente una cota superior de sí mismo. De hecho, n^2 es $\Theta(n^2)$, lo que significa que es a la vez una cota superior y una cota inferior.

Afirmación (b): n^2 es $O(n^3)$

- Esta afirmación también es verdadera porque n^3 crece más lentamente que n^3. Formalmente, existe una constante c tal que $n^2 \leq c \cdot n^3$ para n suficientemente grande. Por ejemplo, si $c = 1$, entonces $n^2 \leq n^3$ cuando $n \geq 1$.

Afirmación (c): n^2 es $O(n^2 \log n)$

- Esta afirmación es verdadera porque n^2 crece más lentamente que $n^2 \log n$. Formalmente, $n^2 \leq n^2 \log n$ para n suficientemente grande, ya que el término $\log n$ es positivo y crece con n.



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>=2

(a) T(n) = d + 3 * T(n/2)
(b) T(n) = d + 2 * 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)

$$T(N) = \begin{cases} cte & N \geq 2 \\ cte + EJERCICIO5(N/2) + EJERCICIO5(\frac{N}{4}) \end{cases}$$

$$cte + T(\frac{N}{2}) + T(\frac{N}{4})$$

7.- Dada la recurrencia

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

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)

$$T(\frac{N}{3}) + 2.C$$

$$T(\frac{N}{6}) + 2.C$$

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

A PAPA MOMO CON BANANAS VERDES NO ENTRA

$$\sum^{2} \left(\sum^{N} c.o \right)$$

BANANAS VERDES NO EH!

$$\sum_{i=0}^{2}\left(\sum_{j=0}^{N} c_0\right)$$

$$\sum_{i=0}^{2} N.c_0 = 2.N.c_0 \quad O(N)$$

$2.1000 = 2000$ Si

100.000 — 1

2000 — x? 0,02 SEG

9.- Considere la siguiente recurrencia:

T(1) = 4

T(n) = 2 T(n/2) + 5n + 1 (n ≥ 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.

$$t(N)\begin{cases} T(1) = 4 \\ 2\,T(N/2) + 5N + 1 \quad (N \geq 2) \end{cases}$$

$T(4) = 2\left(2T\left(\frac{2}{2}\right) + 5.2 + 1\right) + 5.4 + 1$

$t(4) = 2\left(2\left(T\left(\frac{2}{2}\right)\right) + 10 + 1\right) + 20 + 1$

$t(4) = 4\left(T\left(\frac{2}{2}\right)\right) + 20 + 2 + 1$

$t(4) = 4.4 + 43$

$t(4) = 16 + 43 = 59$

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) $T(n) = (1/3) * n^2 * \log_2(n) + \log_2(n)$

$2^k \leq N$

$k \leq \log_2(N)$

$i - 3 \geq N^2$

$i \geq N^2 + 3$

N = 4

16 PASO 1

13 PASO 2

10 PASO 3

$\frac{N^2}{3} = \frac{N \times N}{3}$ PASO K

$$\sum_{i=1}^{\log_2(N)}\left(\sum_{i=1}^{\frac{N^2}{3}} c_0\right)$$

LA CANT DE VECES QUE ITERA TAMBIEN ES $\frac{N^2}{3}$

$$\sum_{i=1}^{\log_2(N)} \frac{N^2}{3}.c_0 = \log_2(N).\frac{N^2}{3}.c_0$$

AL DENOMINADOR 3 SE LO PUEDE DEJAR COMO

$\frac{1}{3}.\log_2(N).N^2.c_0$

AL DENOMINADOR 3 SE LO PUEDE DEJAR COMO

$\frac{1}{3} \cdot Log_2(N) \cdot N^2 \cdot C_0$

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++;
}
```

$$\sum_{i=1}^{N} \left(\sum_{j=1}^{N} cte \right) = \sum_{i=1}^{N} N \cdot cte = N^2 \cdot cte$$

EL TIEMPO ES CUADRATICO $N^2 \cdot cte$

Y ES DE $O(N^2)$

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;
}
```

RTA

$j = \frac{n}{2}$

$j = \frac{n}{4}$

$j = \frac{n}{8}$

$j = \frac{n}{2^{k-1}}$

$2^{k-1} > 1$

$k-1 > Log_2(1)$

$k > Log_2(1)+1$

$k \times 2 < p$

$k < \frac{p}{2}$

$k^2 < p$

$k < Log_2(p)$

$$\sum_{i=1}^{N-1} \left(\sum_{j=1}^{Log_2(N)+1} \left(\sum_{k=1}^{Log_2(p)} cte \right) \right)$$

$N-1 \cdot Log_2(Log_2(p)) + 1$

RETORNA ESTO

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);
}
```

$$\sum_{i=1}^{N/2-1} \left(\sum_{j=1}^{\frac{N}{2}+1} \left(\sum_{k=1}^{Log_2(N)} cte \right) \right)$$

UN J

$k^2 \leq N$

$k \leq Log_2(N)$

$t(N) = \frac{N}{2} - 1 \cdot \frac{N}{2} + 1 \cdot Log_2(N) \cdot cte$

$t(N) = \frac{N}{2} \cdot \frac{N}{2} \cdot Log_2(N) \cdot cte$

$t(N) = O(N^2 \cdot Log_2(N))$

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;
    }
}
```

NA DECREMENTADO TANTO UNO COMO EL OTRO

∴ O(LA ENTRADA MAX)

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");
}
```

$i^2 < N$

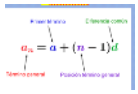
$i < \sqrt{N}$

$$\sum_{i=1}^{\sqrt{N}-1} cte = \sqrt{N}-1 \cdot cte = O(\sqrt{N})$$

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 + ... + 1/n) se puede acotar con O(log n)



$$\sum_{i=1}^{N} \left(\sum_{j=1}^{Log_2(N)} cte \right)$$

$T(N) = N \cdot Log_2(N) \cdot cte$

∴

$O(N \cdot Log(N))$

1

1 + 2 = 3

3 + 3 = 6

6 + 4 = 10

PARA CADA INCREMENTO DE i EL NUMERO DE DE ITERACIONES SE REDUCE

∴ POR DEF LO PODEMOS TOMAR COMO $Log_2(N)$