

Ejercicios

1. Ordenar las siguientes funciones de menor a mayor orden:

- | | | | |
|---------------------|-----------------|----------------|--|
| 1. n | 5. 2^n | 9. $n \log n$ | 13. $\ln n$ |
| 2. $n - n^3 + 7n^5$ | 6. $\log n$ | 10. \sqrt{n} | 14. e^n |
| 3. $n^2 + \log n$ | 7. n^2 | 11. 2^{n-1} | 15. $\log \log n$ |
| 4. n^3 | 8. $(\log n)^2$ | 12. $n!$ | 16. $n^{1+\epsilon}, 0 < \epsilon < 1$ |

Time	n	$n - n^3 + 7n^5$	$n^2 + \log n$	n^3	2^n	$\log n$	n^2	$(\log n)^2$	$n \log n$	\sqrt{n}	2^{n-1}	$n!$	$\ln n$	e^n
1	1	7	1	1	2	0	1	0	0	1	1	1	0	2.7182
5	5	21755	25.6989	125	32	0.6989	25	0.4885	34.948	2.236	16	120	1.6094	148.413
10	10	699,010	101	1000	1024	1	100	1	10	3.162	512	3,628,800	2.3026	22026.46
100	100	$\times 10^{10}$	10002	1000×10^3	2^{100}	2	10000	4	200	10	$\times 10^{30}$	4×10^{51}	4.6051	$\times 10^{13}$

Time	$\log \log n$	$n^{1+\epsilon}, 0 < \epsilon < 1$
1	?	1
5	-0.45591	11,1803
10	0	31,622
100	0.30	1000

$$n! > e^n > 2^n > 2^{n-1} > n - n^3 + 7n^5 > n^3 > n^2 + \log n > n^2 > n^{1+\epsilon}, 0 < \epsilon < 1$$

$$> n \log n > n > \sqrt{n} > (\log n)^2 > \ln n > \log n > \log \log n$$

MÉTODO DE TABLA DE VALORES

Así:

Ordenar las siguientes funciones de menor a mayor orden

$$\log \log n < \log n < \ln n < (\log n)^2 < \sqrt{n} < n < n \log n < n^{1+\epsilon}, 0 < \epsilon < 1$$

$$< n^2 < n^2 + \log n < n^3 < n - n^3 + 7n^5 < 2^{n-1} < 2^n < e^n < n!$$

2. Para las siguientes funciones, determinar el resultado como una función de n y representar el peor caso de ejecución con notación Big Oh:

<pre>function mystery(n) r := 0 for i := 1 to n - 1 do for j := i + 1 to n do for k := 1 to j do r := r + 1 return(r)</pre>	<pre>function pesky(n) r := 0 for i := 1 to n do for j := 1 to i do for k := j to i + j do r := r + 1 return(r)</pre>	<pre>function prestiferous(n) r := 0 for i := 1 to n do for j := 1 to i do for k := j to i + j do for l := 1 to i + j - k do r := r + 1 return(r)</pre>
---	---	---



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

$$\sum_{j=i+1}^n j = \sum_{j=1}^n j - \sum_{j=1}^i j$$

$$\sum_{x=1}^n x^2 = \frac{1}{6} n(n+1)(2n+1)$$

Determinar el resultado como una función de n y representar el peor caso de ejecución con notación Big Oh?

Revisa $i=1, j=1, k=1$

	#Cost(O)	#Times(O)
function mystery(n)		
r := 0	c1	1
for i := 1 to n - 1 do	c2	(n-1)+1
for j := i+1 to n do	c3	(n-1)(n-2+1)
for k := 1 to j do	c4	(n-1)(n-2)(n+1)
r := r + 1	c5	(n-1)(n-2)(n)
return(r)	c6	1
		$T(n) = O(n^3)$

	#Cost(O)	#Times(O)
function pesky(n)		
r := 0	c1	1
for i := 1 to n do	c2	n+1
for j := 1 to i do	c3	n(n+1)
for k := j to i+j do	c4	(n)(n)(n+1)
r := r + 1	c5	(n)(n)(n)
return(r)	c6	1
		$T(n) = O(n^3)$

(Almavero)

	$i=n, j=n, k=n$
function persistent(n)	$\#Cost(0)$ $\#Times(0)$
$r:=0$	C1 1
for $i:=1$ to n do	C2 $(n+1)$
for $j:=1$ to i do	C3 $(n)(n+1)$
for $k:=j$ to $i+j$ do	C4 $(n)(n)(n+1)$
for $l:=1$ to $i+j-k$ do	C5 $(n)(n)(n-1)(n+1)$
$r:=r+1$	C6 $(n)(n)(n-1)(n)$
return(r)	$n-1$ 1
	$T(n) = O(n^4)$

3. Implementar el algoritmo de *insertion sort* para ordenar en orden descendente en vez de ascendente.

```

1  START=1
2
3  def addElement(element, seq):
4      element_index = 0
5      while element_index < len(seq) and not (seq[element_index] > element):
6          element_index = element_index + 1
7      return seq[:element_index] + [element] + seq[element_index:]
8
9  def insertionSort(seq):
10     result = seq[:START]
11     for e in range(START, len(seq)):
12         result = addElement(seq[e], result)
13     return result[::-1]
14
15     input_list = [5, 8, -1, 4, 6]
16
17     print(insertionSort(input_list))
18

```

Run BUSCAMINAS CLASS MATERIAL 2

C:\Users\ANDERSSON\AppData\Local\Microsoft\WindowsApps\python3.11.exe "C:\User: [8, 6, 5, 4, -1]

Process finished with exit code 0