

Algoritmia

Práctica 1.2

15/2/2026

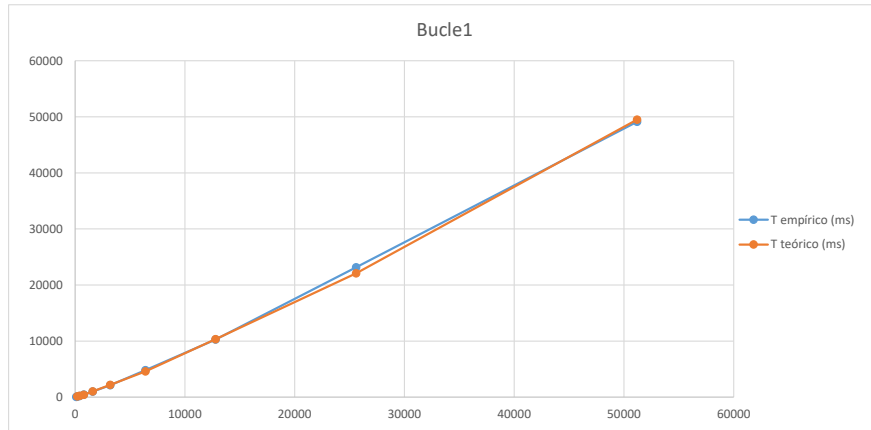
Contenidos

1.	Bucle 1	2
2.	Bucle 2	3
3.	Bucle 3	4
4.	Bucle 4	5
5.	Comparativas entre bucles	6
6.	Bucle 5	7
7.	Bucle 6	8
8.	Bucle 7	9
9.	Python vs Java (Bucle4)	10
10.	Condiciones de medida	10

1. Bucle 1

El método `bucle1` tiene dos partes anidadas. La primera de ellas recorre n^2 elementos, multiplicando el contador de índice por 3 en cada paso. Así, la complejidad teórica esperada es de $O(\log n^2) = O(\log n)$. La segunda abarca $2n$ elementos saltando de 3 en 3, luego la complejidad de esta parte es $O(n)$. En total, `bucle1` es $O(n \log n)$.

n	T empírico (ms)	Repeticiones	T teórico (ms)
100	51	10.000	-
200	99	10.000	117,35
400	192	10.000	223,90
800	458	10.000	428,42
1.600	993	10.000	1.010,98
3.200	2.111	10.000	2.172,59
6.400	4.793	10.000	4.584,59
12.800	10.285	10.000	10.344,15
25.600	23.169	10.000	22.077,64
51.200	49.114	10.000	49.502,33



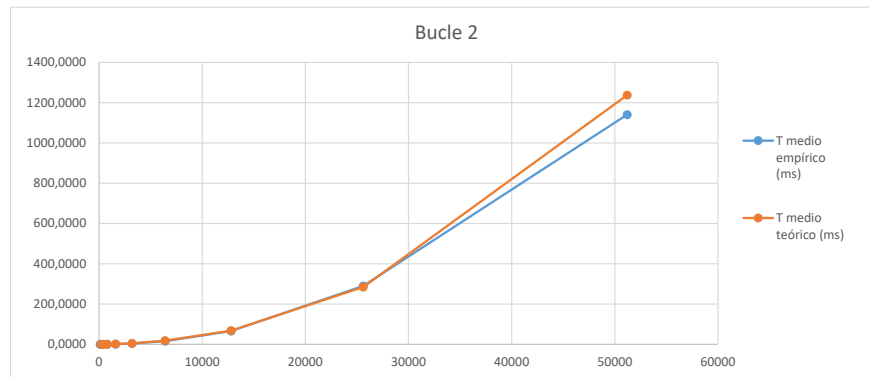
Comparación de tiempos empíricos y teóricos de `bucle1`

2. Bucle 2

El método `bucle2` contiene un bucle externo que divide su variable de control entre 3 en cada iteración hasta llegar a 1, lo que supone $O(\log n)$. En cada una de esas iteraciones se ejecutan dos bucles anidados: el primero recorre n elementos y el segundo aproximadamente $n/2$, por lo que juntos aportan una complejidad $O(n^2)$. En consecuencia, la complejidad total de `bucle2` es $O(n^2 \log n)$.

n	$T_{emp} \text{ total (ms)}$	Repeticiones	$T_{teo} \text{ total (ms)}$	$\overline{T}_{emp} \text{ (ms)}$	$\overline{T}_{teo} \text{ (ms)}$
100	54	10.000	-	0,0054	-
200	151	10.000	248,5112	0,0151	0,0249
400	694	10.000	683,0177	0,0694	0,0683
800	3.159	10.000	3.097,1530	0,3159	0,3097
1.600	10.200	10.000	13.946,2643	1,0200	1,3946
3.200	40.955	10.000	44.633,1972	4,0955	4,4633
6.400	156	10	177,8892	15,6000	17,7889
12.800	663	10	673,3520	66,3000	67,3352
25.600	2.897	10	2.846,3732	289,7000	284,6373
51.200	11.410	10	12.379,3216	1.141,0000	1.237,9322

Tiempos empíricos y teóricos totales (tiempo de todas las repeticiones) y medios (entre el número de repeticiones)

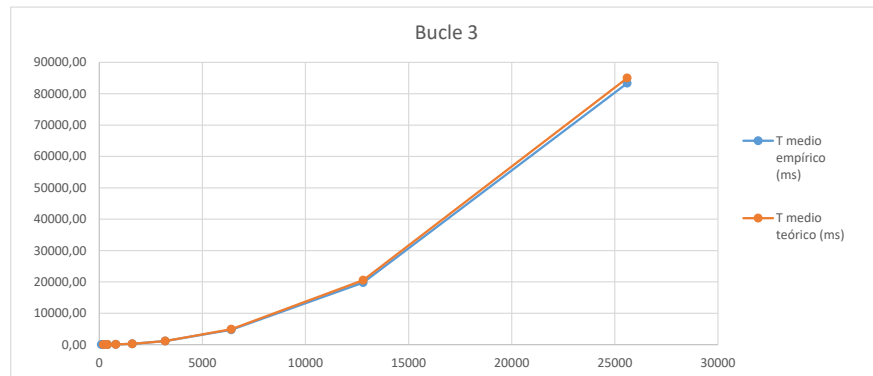


Las repeticiones, luego hemos de graficar usando los tiempos medios

3. Bucle 3

El método `bucle3` tiene su bucle más externo que recorre $2n$ elementos, por lo que su complejidad es $O(n)$. En cada iteración, un segundo bucle recorre desde el valor actual (`i`) hasta 0 decrementando de 2 en 2, lo que supone una complejidad $O(n)$. Además, existe un tercer bucle que duplica su variable hasta alcanzar n , aportando una capa más de complejidad $O(\log n)$. Así, se obtiene una complejidad total $O(n^2 \log n)$.

n	$T_{emp} \text{ total (ms)}$	<i>Repeticiones</i>	$T_{teo} \text{ total (ms)}$	$\overline{T}_{emp} \text{ (ms)}$	$\overline{T}_{teo} \text{ (ms)}$
100	82	100	-	0,82	-
200	350	100	377,3689	3,50	3,7737
400	1.498	100	1.583,1536	14,98	15,8315
800	6.422	100	6.685,2091	64,22	66,8521
1.600	26.802	100	28.351,6648	268,02	283,5166
3.200	11.298	10	11.728,0289	1.129,80	1.172,8029
6.400	47.576	10	49.073,1884	4.757,60	4.907,3188
12.800	19.801	1	20.535,5104	19.801,00	20.535,5104
25.600	83.367	1	85.009,1037	83.367,00	85.009,1037
51.200	Fdt	-	-	-	-

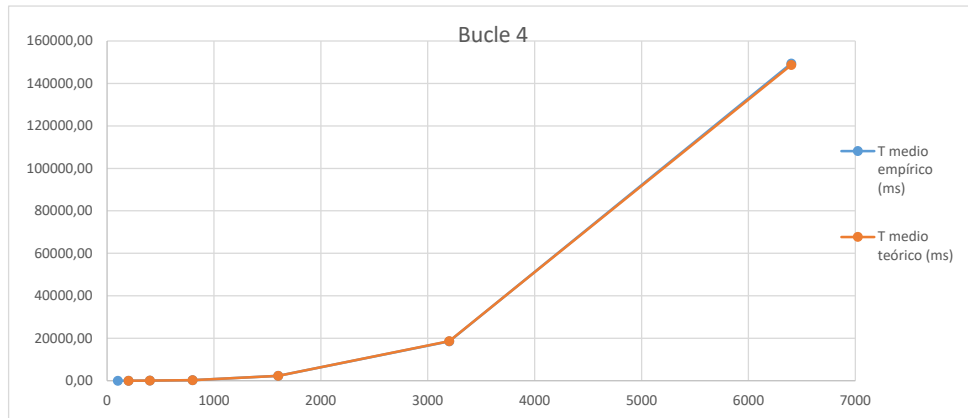


Una vez más, comparamos las medias en igualdad de unidades

4. Bucle 4

El método `bucle4` consta de tres bucles anidados donde el primero recorre n elementos. El segundo bucle, en la iteración i -ésima del primero recorre i elementos. Análogamente, el tercer bucle recorre j (índice del segundo) elementos. En total, la complejidad del método es de $O(n^3)$.

n	$T_{emp} \text{ total (ms)}$	$Repeticiones$	$T_{teo} \text{ total (ms)}$	$\overline{T}_{emp} \text{ (ms)}$	$\overline{T}_{teo} \text{ (ms)}$
100	70	100	-	0,70	-
200	478	100	560,00	4,78	5,60
400	3.560	100	3.824,00	35,60	38,24
800	28.111	100	28.480,00	281,11	284,80
1.600	2.325	1	2.248,88	2.325,00	2.248,88
3.200	18.583	1	18.600,00	18.583,00	18.600,00
6.400	149.392	1	148.664,00	149.392,00	148.664,00
12.800	FdT	-	-	-	-



5. Comparativas entre bucles

Cociente `bucle1` y `bucle2`

n	T_1 (ms)	T_2 (ms)	T_1/T_2
100	0,0051	0,0054	0,9444
200	0,0099	0,0151	0,6556
400	0,0192	0,0694	0,2767
800	0,0458	0,3159	0,1450
1.600	0,0993	1,0200	0,0974
3.200	0,2111	4,0955	0,0515
6.400	0,4793	15,6000	0,0307
12.800	1,0285	66,3000	0,0155
25.600	2,3169	289,7000	0,0080
51.200	4,9114	1.141,0000	0,0043

Los cocientes van decreciendo claramente en cada iteración, en línea con el resultado que esperábamos por sus complejidades teóricas.

Cociente `bucle3` vs `bucle1`

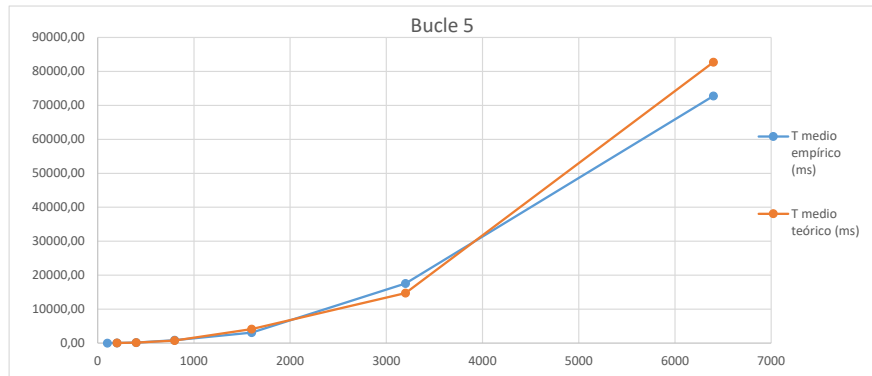
n	T_3 (ms)	T_2 (ms)	T_3/T_2
100	0,82	0,0054	151,8519
200	3,50	0,0151	231,7881
400	14,98	0,0694	215,8501
800	64,22	0,3159	203,2922
1.600	268,02	1,0200	262,7647
3.200	1.129,8	4,0955	275,8638
6.400	4.757,6	15,6000	304,9744
12.800	19.801	66,3000	298,6576
25.600	83.367	289,7000	287,7701
51.200	-	1.141,0000	-

Los cocientes se estabilizan para n grande, en línea con lo que esperábamos (mismas complejidades teóricas).

6. Bucle 5

Para lograr un bucle de complejidad $O(n^2 \log^2 n)$ de manera sencilla, hemos optado por usar dos bucles de complejidad $O(n)$ y otros dos de complejidad $O(\log n)$, todos ellos anidados.

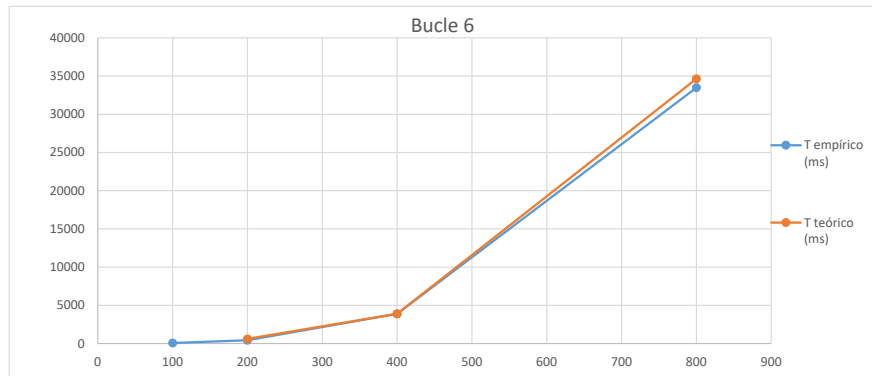
n	$T_{emp} \text{ total (ms)}$	Repeticiones	$T_{teo} \text{ total (ms)}$	$\overline{T_{emp}} \text{ (ms)}$	$\overline{T_{teo}} \text{ (ms)}$
100	699	100	-	6,99	-
200	3.181	100	3.701,0226	31,81	37,0102
400	15.338	100	16.270,980	153,38	162,7098
800	848	1	763,6865	848,00	763,6865
1.600	3.073	1	4.131,9246	3.073,00	4.131,9246
3.200	17.538	1	14.710,1879	17.538,00	14.710,1879
6.400	72.779	1	82.719,0401	72.779,00	82.719,0401
12.800	FdT	-	-	-	-



7. Bucle 6

Para lograr un bucle de complejidad $O(n^3 \log n)$ usamos tres bucles de complejidad $O(n)$ y otro de complejidad $\log n$, todos ellos anidados.

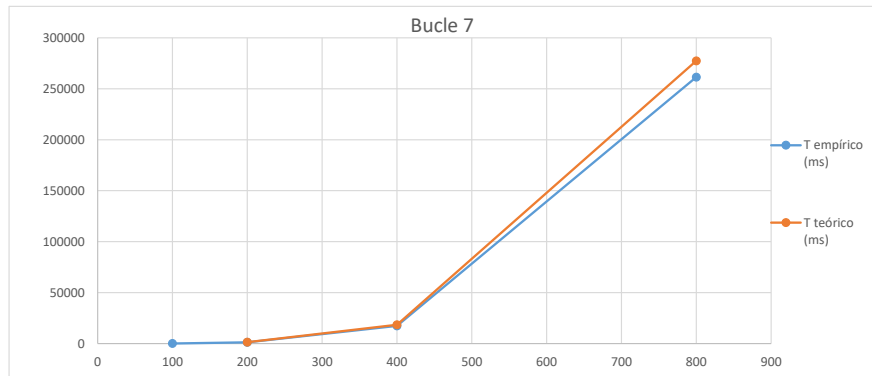
n	$T_{emp} (ms)$	Repeticiones	$T_{teo} (ms)$
100	66	1	-
200	429	1	607,4719
400	3.880	1	3.880,9880
800	33.469	1	34.630,9899
1.600	FdT	-	-



8. Bucle 7

Para lograr un bucle de complejidad $O(n^4)$ podemos usar 4 bucles anidados de complejidad $O(n)$.

n	$T_{emp} \text{ (ms)}$	$Repeticiones$	$T_{teo} \text{ (ms)}$
100	85	1	-
200	1.151	1	1.360
400	17.339	1	18.416
800	261.464	1	277.424
1.600	FdT	-	-



9. Python vs Java (Bucle4)

n	$T_{Python} (ms)$	$T_{Java total} (ms)$	$Repeticiones Java$	$T_{Java medio} (ms)$	T_{Py}/T_J
200	113	468	100	4,68	24,1453
400	800	234	1	234,00	3,4188
800	7.985	1.569	1	1.569,00	5,0892
1.600	69.665	14.186	1	14.186,00	4,9108
3.200	FdT	110.528	1	110.528,00	-
6.400	FdT	FdT	-	-	-

El cociente de tiempos se va estabilizando.

10. Condiciones de medida

Todas las medidas en Java han sido realizadas con Eclipse IDE usando la opción `-Xint` como argumento para la JVM.

Los apartados 1 al 5 de esta memoria han usado las siguientes especificaciones: [CPU 12th Gen Intel(R) Core(TM) i5-12400 @ 2,5GHz + 16GB RAM] , y el resto [CPU: Intel Core i7-6700HQ CPU @ 2,60GHz + 16GB RAM].