

# Paso de parámetros por valor y por referencia

13/12/2021

Alcántara Estrada Kevin Isaac

## 1 Comparación de Ordenamientos

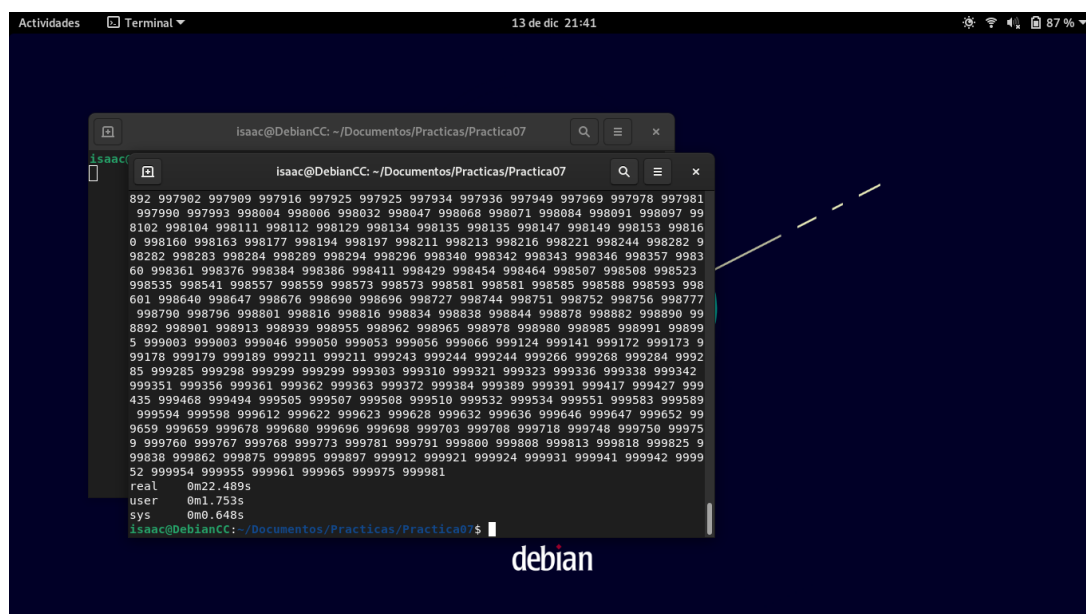
Tras ordenar un arreglo de 100000 elementos generados de manera aleatoria entre 0 y un número dado por el usuario, fue posible notar que el algoritmo de Bubble Sort es mucho más lento que el de Quick Sort, prueba de ello es el tiempo que tardó el programa en iniciar y finalizarse.

## 2 ¿Por qué?

Esto ocurre porque el algoritmo de Quick Sort toma un elemento para fijarlo y ordenar los elementos que son mayores y menores a este, sin importar mucho el orden; sin embargo, ahora se ordenan los elementos que quedaron antes y después del elemento pivote, aplicando el divide y vencerás.

Por otro lado, el algoritmo Bubble Sort primero lleva el elemento de mayor valor al final del arreglo, esto a través de compararlo con el resto de elementos del arreglo y repite el procedimiento con cada uno de los elementos del arreglo.

Esta diferencia procedimental lleva a que la complejidad de los algoritmos y el tiempo que tardan en finalizar su ejecución sean distintos. Particularmente la complejidad del algoritmo Bubble Sort es de  $O(n^2)$ , mientras que la de Quick Sort es de  $O(n * \log_2 n)$  (menor que la de Bubble Sort, aunque en el peor de los casos, la complejidad de Quick Sort será igual a la de Bubble Sort).



```
isaac@DebianCC: ~/Documentos/Practicas/Practica07
892 997982 997989 997916 997925 997925 997934 997936 997949 997969 997978 997981
997990 997993 998004 998006 998032 998047 998068 998071 998084 998091 998097 99
8102 998104 998111 998112 998129 998134 998135 998135 998147 998149 998153 99816
0 998160 998163 998177 998194 998197 998211 998213 998216 998221 998244 998282 9
98282 998283 998284 998289 998294 998296 998340 998342 998343 998346 998357 9983
60 998361 998376 998384 998386 998411 998429 998454 998464 998507 998508 998523
998535 998541 998557 998559 998573 998573 998581 998581 998585 998588 998593 998
601 998640 998647 998676 998690 998696 998727 998744 998751 998752 998756 998777
998790 998796 998801 998816 998816 998834 998838 998844 998878 998882 998890 99
8892 998901 998913 998939 998955 998962 998965 998978 998980 998985 998991 99899
5 999003 999003 999046 999050 999053 999056 999066 999124 999141 999172 999173 9
99178 999179 999189 999211 999211 999243 999244 999244 999266 999268 999284 9992
85 999285 999298 999299 999299 999303 999310 999321 999323 999336 999338 999342
999351 999356 999361 999362 999363 999372 999384 999389 999391 999417 999427 999
435 999468 999494 999505 999507 999508 999510 999532 999534 999551 999583 999589
999594 999598 999612 999622 999623 999628 999632 999636 999646 999647 999652 99
9659 999659 999678 999680 999696 999698 999703 999708 999718 999748 999750 99975
9 999760 999767 999768 999773 999781 999791 999800 999808 999813 999818 999825 9
99838 999862 999875 999895 999897 999912 999921 999924 999931 999941 999942 9999
52 999954 999955 999961 999965 999975 999981
real 0m22.489s
user 0m1.753s
sys 0m0.648s
isaac@DebianCC: ~/Documentos/Practicas/Practica07$
```

Figure 1: Tiempo que tardó el algoritmo de Quick Sort

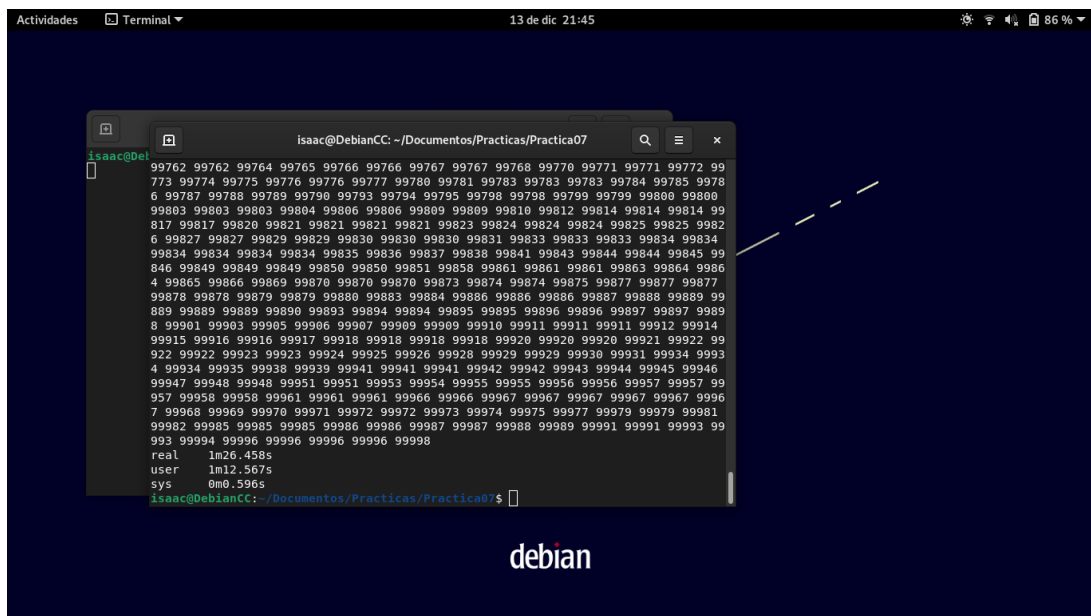


Figure 2: Tiempo que tardó el algoritmo Bubble Sort