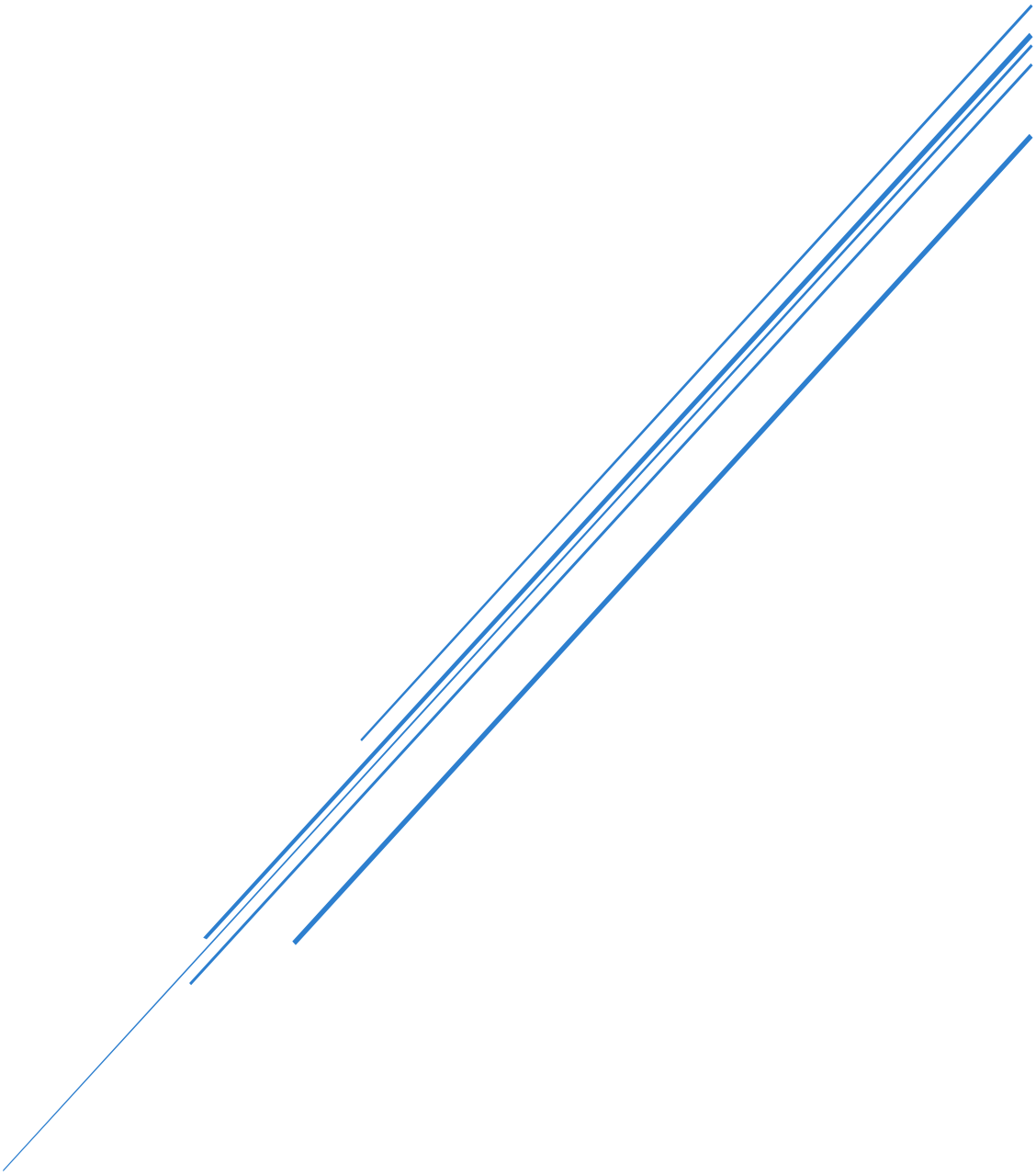
SESIÓN 1.1

Algoritmia-Prácticas de Laboratorio



Jorge Blanco Sánchez
UO293697

Contenido

El tiempo en Java

Java almacena el tiempo en formato long, lo que nos permite tener un total de 2^64 milisegundo, el tiempo 0 de Java fue figado en el día 1 de enero de 1970, y este formato durará más de 584 mil millones de años

A la hora de tomar tiempos en nuestro algoritmo nos podemos encontrar con tiempos de 0, aunque este tiempo sea 0, el procesador ha tardado una serie de micras de segundo que el formato no es capaz de almacenar, también tenemos que tener en cuenta que los tiempos inferiores a 50 msg no son fiables, debido a procesos interno de Java, para evitar dichos tiempos se requiere, y en ocasiones, de usar nuestro algoritmo en bucles una serie de iteración y dividir el tiempo obtenido entre dichas iteraciones, por ejemplo: para el programa **Vectores2** a una repetición el tiempo es de 0 msg, pero si hacemos la operación en un bucle a **20000000 de repeticiones** resulta un tiempo de 163 msg, lo que equivale a 8,15 nano segundos tras hacer la división.

Toma de tiempos pequeños en Algoritmos

Para esta primera parte tomaremos como ejemplo la clase **Vector4** , cuya complejidad es de O(n), esto quiere decir que si aumentamos la cantidad de elementos k*n veces, el tiempo resultante se va a multiplicar por k, esto resulta de aplicar la siguiente formula: $T_{resultado} = \frac{O(k*n)}{O(n)} * T_n$, al ser O(n) una función lineal, la fracción se va a simplificar quedando en la constante k y resultando un tiempo k veces el tiempo en n, si esto lo aplicamos para k=2, tenemos que el tiempo obtenido es doble del tiempo para n, para comprobarlo, tomamos tiempos, y obtenemos que para n =1000 el tiempo es de 0,085 ms y si duplicamos el problema, es decir, para n=2000, el tiempo resultante es de 0,164, comparando los tiempos $\frac{0,164}{0,085} \approx 2$ (un valor más aproximado es 1,93) y si sustituimos en la formula anterior $T_{resultado} = \frac{2*n}{n} * 0,085$ (O(n)=n; O(2*n)=2*n) nos resulta un valor de 0,17, por lo que los tiempo obtenidos son esperados a la complejidad del problema

n	Tsuma(ms)	Tmax(ms)	T coincidencias 1 (ms)	T coincidencias 2 (ms)
10000	0,0975	0,095	613	0,105714286
20000	0,15875	0,17	2438	0,165714286
40000	0,29625	0,326	9580	0,327142857
80000	0,59875	0,648	37567	0,637142857
160000	1,23625	1,29	155866	1,297142857
320000	2,50125	2,864	FdT	2,767142857
640000	4,70875	5,404	FdT	5,237142857
1280000	9,84625	10,952	FdT	10,77857143
2560000	17,99875	21,328	FdT	21,37285714
5120000	35,8475	42,962	FdT	43,25285714
10240000	73	84	FdT	87,72857143
20480000	137	164	FdT	175
40960000	287	335	FdT	351
81920000	571	699	FdT	812

Con los tiempos obtenidos, podemos concluir que cumplen con los esperado en función de su complejidad

Estos tiempos fueron tomados en un ordenador con procesador: Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz y 16 GB de memoria;

T suma ,T max,T coincidencias 1 y T coincidencias 2 corresponden a las clases Vectores4, Vectores5, Vectores6 y Vectores7, respectivamente