**PRACTICA 3 ALGORITMIA**

El objetivo de esta práctica es estudiar la complejidad temporal y el tiempo de ejecución de distintos algoritmos recursivos Divide y Vencerás (con sustracción y con división). También se dará una solución a un problema concreto con este tipo de algoritmos, se medirán tiempos de ejecución de dicho algoritmo y se analizarán los resultados obtenidos. Hay que tener en cuenta que si es por sustracción, si a = 1 la complejidad será O(nk+1), en caso contrario, la complejidad será O(an/b). Por otra parte, si se realiza por división: si a < bk la complejidad es O(nk); si a = bk la complejidad es O(nklog(n); si a > bk entonces la complejidad será O(nlogba).

1. Tras analizar las complejidades de las clases Sustracción1, Sustracción2 y Sustracción3 los resultados obtenidos son los siguientes:

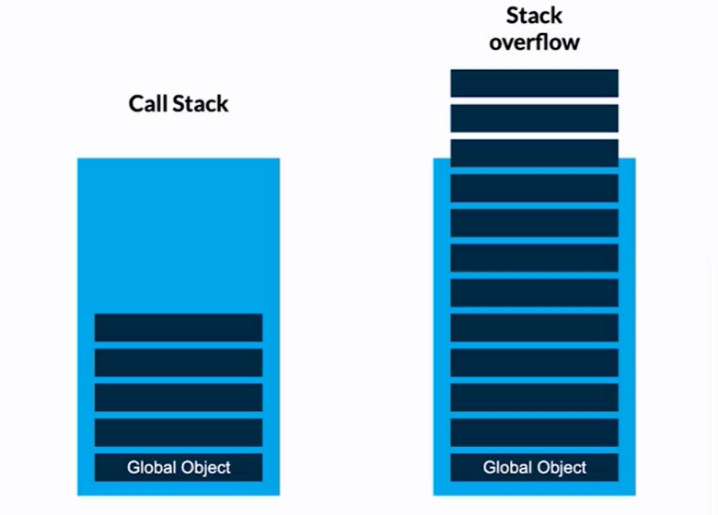
* Sustracción 1: Teniendo en cuenta que a = 1, b = 1 y k = 0 la complejidad será O(nk+1), es decir, O(n1), que es **O(n).**

Teniendo en cuenta que tiene una complejidad lineal, se verá a continuación si la gráfica cumple con lo esperado:

* Sustracción 2: Teniendo en cuenta que a = 1, b = 1 y k = 1 la complejidad será O(nk+1), es decir, **O(n2).**Si se observa la siguiente gráfica se puede ver que los tiempos obtenidos son los esperados de una complejidad cuadrática:
* Sustracción 3: Teniendo en cuenta que a = 2, b = 1 y k = 0 la complejidad será O(an/b), es decir, **O(2n).**

Si se observa la gráfica se puede ver que cumple lo esperado de una función exponencial.

Los métodos de las clases Sustraccion1 y Sustraccion2 dejan de dar tiempos a partir de n = 8000. Esto sucede ocasionando un error: java.lang.StackOverflowError. Es así porque en cada llamada a un método se utiliza una parte de la memoria, denominada stack. Si se hacen muchas llamadas puede ocurrir que ese espacio se llene y genera un stack overflow. Básicamente ocurre un desbordamiento de tareas. Algunos de los motivos pueden ser: bucles infinitos, funciones recursivas sin control, cambios de estado continuo…



A continuación, se quiere determinar cuantos años tardaría en finalizar la ejecución de Sustracción 3, pues ya se vio anteriormente que la complejidad de este método crece exponencialmente.

* El tiempo expresado en milisegundos es: 8,50856E+20 ms.
* El tiempo expresado en años es: 26.980.469.001 años.

Es decir, el método Sustracción 3 tarda millones de años en ejecutarse para n = 80.

Para continuar con la práctica se implementan las clases Sustracción 4 y Sustracción 5 con complejidades O(n3) y O(3n/2) respectivamente.

* Sustracción 4: Este algoritmo, como bien se especificó anteriormente, tiene una complejidad O(n3). Como se ve, los datos cumplen lo esperado de una complejidad de este tipo.

|  |  |
| --- | --- |
| **n** | **Algoritmo Sustracción 4** |
| 100 | 1.88 |
| 200 | 175 |
| 400 | 1195 |
| 800 | 9095 |
| 1600 | 73426 |
| 3200 | 659954 |

* Sustracción 5: Este algoritmo tiene una complejidad O(3n/2). Como se observa, los datos obtenidos tanto en la tabla como en la gráfica son los esperados.

|  |  |
| --- | --- |
| **n** | **Algoritmo Sustracción 5** |
| 30 | 4930 |
| 32 | 14857 |
| 34 | 54850 |
| 36 | 135243 |
| 38 | 409864 |

¿Cuánto tardará en finalizar la ejecución Sustracción 5 para n = 80?

* Tardará 4,17713E+15 ms
* Expresado en años será: 132.456,0136 años

1. Ahora se realizarán algoritmos Divide y Vencerás por división, cuyas fórmulas sobre el cálculo de complejidad se explicaron al principio del documento.