Activity 1. Divide and Conquer by subtraction

* Subtraction1:

It stops giving times at n = 8192, due to stackOverflowError as it is mentioned in the pdf of the practice “Subtraction1.java and Subtraction2.java classes have an approach by subtraction with a=1, which involves a large expenditure of stack memory”, it overflows.

* Subtraction2:

It stops exactly at the same size, it makes sense since it has the same problem as Susbtraction1.

* How many years will take for Subtraction3 complete n=80?

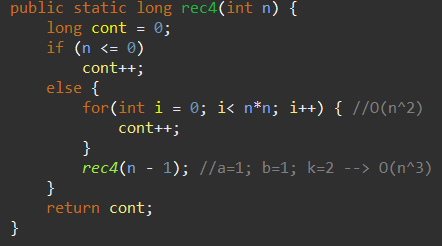
For calculating this we have to make the following operation: T(n) = 2 T(n-1)+O(1)

The last measurable value I got for Susbtraction3 is m=30 t=38705ms.

* Substraction4

We have to implement a divide and conquer by subtraction algorithm with complexity O(n3), in my case I implemented an algorithm with a=1; b=1; k=2.

As a=1 the complexity is O(nk+1) which is O(n3)



Times I got:

|  |  |
| --- | --- |
| n | t(ms) |
| 100 | 1 |
| 200 | 11 |
| 400 | 86 |
| 800 | 713 |
| 1600 | 5845 |
| 3200 | 47420 |
| 6400 | OoT |

Times match the complexity, let’s take for instance n=1600, with this size the algorithm finished at 5845milliseconds, by multiplying the size by 2, I got a time which is more or less 23 times bigger (n=3200; t = 47420), 47420/5845 = 8,11.

* Subtraction5:

We have to implement a divide and conquer by subtraction algorithm with complexity O(3n/2), for that, we have to implement an algorithm with a=3 and b=2 (in my case k=0)

As a > 1 the complexity is O(an/b).

Texto

El contenido generado por IA puede ser incorrecto.

Times I got:

|  |  |
| --- | --- |
| n | t(ms) |
| 30 | 399 |
| 32 | 1181 |
| 34 | 5535 |
| 36 | 10763 |
| 38 | 32119 |
| 40 | OoT |

For n=30 we get a time of 399ms, this match the complexity since as we increase n by 2, times are multiplied more or less by 3, and this is because n is divided by b, which is 2, so it is growing by one and then being multiplied by 3 make sense.

Activity 2. [TITLE OF THE ACTIVITY]

[ANSWER].

…