Algorithmics	Student information	Date	Number of session
	UO: 301022	06/03/2025	5
	Surname: Canga		
	Name: Martín		

Escuela de Ingeniería Informática
Universidad de Oviedo

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Activity 1. [Divide and Conquer by Subtraction]

We can see that:

- Substraction1 follows the next complexity: $O(n)$
- Substraction2 follows the next complexity: $O(n^2)$
- Substraction3 follows the next complexity: $O(2^n)$

We cannot get to any **n value** for which the program expends more than 1 minute since we will always get a stack overflow error that stops the execution of the program.

Regarding the execution of Substraction3. There are about 2^{80} calls, which, at an optimistic rate of one nanosecond per call, would take roughly 38 million years.

Subtraction-4 times:

N	Time(ms)
30	284
32	790
34	793
35	2408
36	2386
37	7181
38	7496
39	21987
40	21691

Subtraction-5 times:

N	Time(ms)
30	284
32	790
34	793
35	2408
36	2386
37	7181
38	7496
39	21987
40	21691

Calculating the time, it would take for Subtraction-5 to execute for $n = 80$

Would be approximately around $T(80) = 1982.4 \text{ seconds} = 33.04 \text{ minutes}$

This result is based on the time per iteration of $n = 34$

Algorithmics	Student information	Date	Number of session
	UO: 301022	06/03/2025	5
	Surname: Canga		
	Name: Martín		

Escuela de Ingeniería Informática
Universidad de Oviedo

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

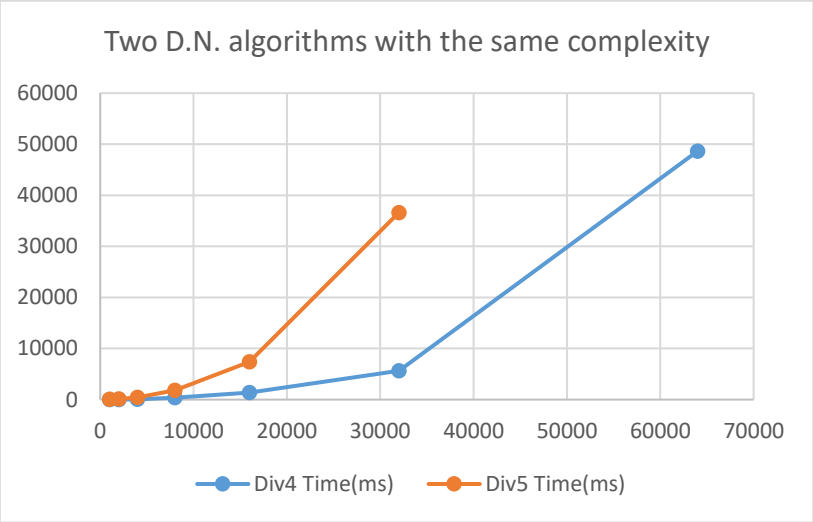
Activity 2. [Divide and Conquer by Division]

Division-4 times:

N	Time(ms)
1000	5
2000	23
4000	89
8000	351
16000	1394
32000	5639
64000	48655

Division-5 times:

N	Time(ms)
1000	32
2000	114
4000	454
8000	1827
16000	7369
32000	36625



We can see that the although the size of the trends is different the growth rate is almost the same hence the complexities are the same.

Activity 3. [Check space complexity]

The vector sum performs an algorithm $O(1)$ but as we increase the option parameter, more calls are made to perform the same operation. So, the constant parameter increases in the following order:

- sum1 $O(n)$ (Iterative, O(1) memory space)
- sum3 $O(n)$ (Recursive D&C by division, O(log(n)) memory space)
- sum2 $O(n)$ (Recursive, O(n) memory space)

The Fibonacci succession algorithm performs $O(n)$ but as we increase the option parameter, more calls are done to perform the same operation.

The constant parameter increases by:

- fib1 $O(n)$ (Iterative, O(1) memory space)
- fib2 $O(n)$ (Iterative, O(n) memory space)
- fib3 $O(n)$ (Recursive with handler, O(n) memory space)
- fib4 $O(n^2)$ fib1(Recursion, O(n) memory space)

Activity 4. [Check space complexity]

N	Time(ms)
2	LoR
4	LoR
8	LoR
16	LoR
32	LoR
64	LoR
128	LoR
256	LoR
512	LoR
1024	121
2048	443
4096	1699
8192	6544
16384	25871

The times measured matches with the algorithm theoretical complexity $O(n^2)$