


Algorithmics	Student information	Date	Number of session
	UO: 300218	3/3/2025	3
	Surname: Berdayes G.Pumarino	 Escuela de Ingeniería Informática Universidad de Oviedo	
	Name: Víctor		



Activity 1. Stack Overflow Exception

The complexity of the classes Subtraction1 is $O(n)$ and Subtraction2 is $O(n^2)$

Subtraction1 and Subtraction2 stop giving values after $n = 16384$ and $n = 32768$ respectively as they are adding values to the stack faster that it can process them throwing a stack overflow error when there is no more space in the stack.

Activity 2. Subtraction3 approximation for $n = 80$

Taking into account the complexity of the Subtraction 3, $O(2^n)$, we can approximate the time it would take to execute for $n = 80$ as $2^{80} = 1.209 \cdot 10^{24}$ ms.

(1,208,925,819,614,629,174,706,176)

$$1.209 \cdot 10^{24} \text{ms} \cdot \frac{1 \text{ s}}{1000 \text{ ms}} \cdot \frac{1 \text{ h}}{3600 \text{ s}} \cdot \frac{1 \text{ d}}{24 \text{ h}} \cdot \frac{1 \text{ y}}{365.25 \text{ d}} \simeq$$

38.309 trillion years (american trillion)

Activity 3. Subtraction4 and Subtraction5 tables

n	Subtraction4 times(ms)
100	1
200	1
400	3
800	33
1600	214
3200	1483
6400	10829
12800	10829
25600	OoT

Algorithmics	Student information	Date	Number of session
	UO: 300218	3/3/2025	3
	Surname: Berdayes G.Pumarino		
	Name: Víctor		

n	Substraction5 times(ms)
30	27
32	83
34	134
36	697
38	1188
40	6292
42	10753
44	56782
46	OoT

To compute the time it would take for Substraction5 to execute for n=80 we can calculate by doing $3^{(80/2)} = 3^{40} \approx 1.216 \cdot 10^{19} \text{ ms. } (12,157,665,459,056,928,801)$

$$1.216 \cdot 10^{19} \cdot \frac{1 \text{ s}}{1000 \text{ ms}} \cdot \frac{1 \text{ h}}{3600 \text{ s}} \cdot \frac{1 \text{ d}}{24 \text{ h}} \cdot \frac{1 \text{ y}}{365.25 \text{ d}} \approx 385.327 \text{ million years}$$

Activity 4. Divide and Conquer with division

For Division1 and Division3 their complexities are $O(n)$ and for Division2 the complexity is $n \cdot \log(n)$ this is accurate as the times for Division1 and Division3 are almost identical and for Division2 we can see it's slightly worse.

Algorithmics	Student information	Date	Number of session
	UO: 300218	3/3/2025	3
	Surname: Berdayes G.Pumarino		
	Name: Víctor		

Activity 5. Fibonacci and VectorSum

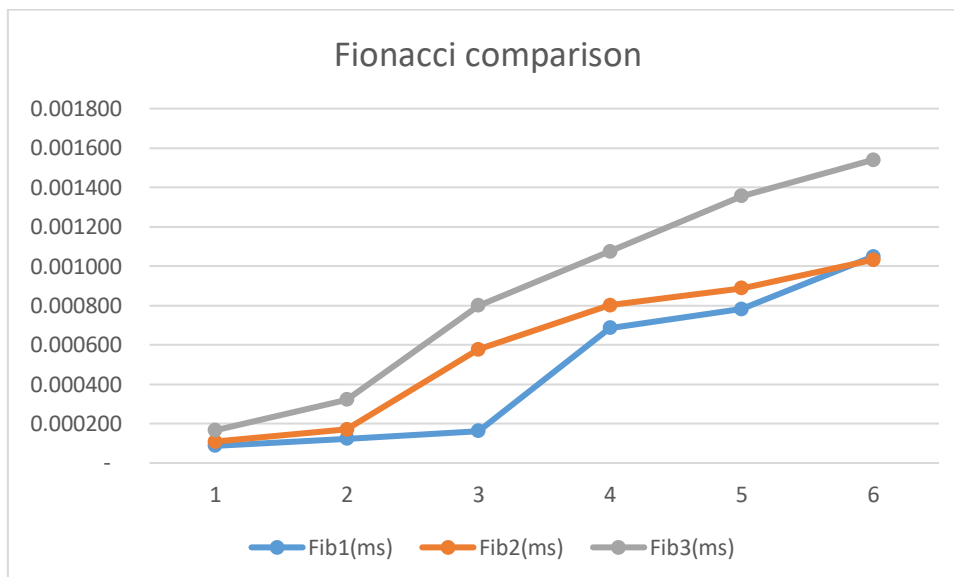
n	Fib1(ms)	Fib2(ms)	Fib3(ms)	Fib4(ms)
10	0.000087	0.000109	0.000166	0.002118
20	0.000123	0.000171	0.000321	0.6947
30	0.000162	0.000576	0.000799	34.83
40	0.000685	0.000801	0.001074	7159.34
50	0.000782	0.000888	0.001355	OoT
60	0.001049	0.001032	0.001540	OoT

n	Sum1(ms)	Sum2(ms)	Sum3(ms)
3	0.00004	0.00008	0.0001
6	0.00007	0.00012	0.00019
12	0.00008	0.00022	0.00037
24	0.00012	0.0004	0.00075
48	0.00022	0.00078	0.0015
96	0.00039	0.00151	0.00301
192	0.00075	0.00296	0.00601
384	0.00143	0.00599	0.01189
768	0.00272	0.01246	0.04560
1536	0.00551	0.04737	0.10791
3072	0.01092	0.10306	0.21708
6144	0.03906	0.20614	0.43144
12288	0.09859	StackOverflow	0.85594
24576	0.21360	StackOverflow	2.0881
49152	0.39465	StackOverflow	3.8553
98304	0.79658	StackOverflow	8.1396

Algorithmics	Student information	Date	Number of session
	UO: 300218	3/3/2025	3
	Surname: Berdayes G.Pumarino		
	Name: Víctor		

Comparison of the algorithms:

fib1/fib2	fib1/fib3	fib1/fib4	fib2/fib3	fib2/fib4	fib3/fib4
0,798165	0,524096	0,798165	0,656627	0,051464	0,078376
0,719298	0,383178	0,719298	0,53271	0,000246	0,000462
0,28125	0,202753	0,281250	0,720901	1,65E-05	2,29E-05
0,855181	0,637803	0,855181	0,74581	1,12E-07	1,5E-07
0,880631	0,577122	OoT	0,655351	OoT	OoT
1,016473	0,681169	OoT	0,67013	OoT	OoT



Here we can see that fib4 is the least efficient as it is exponential and does not fit in the graph, followed by fib3 and fib2 and lastly there is fib1 being the most efficient out of the algorithms although we can see that at higher numbers it is very close to fib2 and for n=60 is worse, but this is probably an outlier.

Algorithmics	Student information	Date	Number of session
	UO: 300218	3/3/2025	3
	Surname: Berdayes G.Pumarino		
	Name: Víctor		

sum1/sum2	sum1/sum3	sum2/sum3
0,5	0,4	0,8
0,583333	0,368421	0,631579
0,363636	0,216216	0,594595
0,3	0,16	0,533333
0,282051	0,146667	0,52
0,258278	0,129568	0,501661
0,253378	0,124792	0,492512
0,238731	0,120269	0,503785
0,218299	0,059649	0,273246
0,116318	0,051061	0,438977
0,105958	0,050304	0,474756
0,189483	0,090534	0,477795
	0,115183	
	1,02E-05	
	1,02E-05	
	9,79E-06	

We can see that sum1 is the most efficient algorithm closely followed sum2 and the worst of all them is sum3.

Activity 6. Petanque championship