


Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso	 Escuela de Ingeniería Informática Universidad de Oviedo	
	Name: Sergio		



Activity 1. Factor 1 (problem size)

Made with the computer in lab: i5-12400

n	execution time (ms)
10000	1605
20000	6407
40000	25609
80000	OoT
160000	OoT
320000	OoT
640000	OoT

Activity 2. Factor 2 (computer power)

i5-12400, 32GB DDR4(3600MHz):

n	execution time (ms)
10000	1605
20000	6407
40000	25609
80000	OoT
160000	OoT
320000	OoT
640000	OoT

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

Intel ultra 7 155H, 32GB DDR5(5600mhZ):

n	execution time (ms)
10000	1549
20000	6269
40000	25455
80000	OoT
160000	OoT
320000	OoT
640000	OoT

There is no much difference between the times of this CPU since they have more or less the same power, as we can see in this [web page](#):

Because of the similar performance, I executed the same program in another computer.

i5-13600KF, 32GB DDR5(5600mhZ):

n	execution time (ms)
10000	1238
20000	5202
40000	21027
80000	OoT
160000	OoT
320000	OoT
640000	OoT

Even with this last CPU which is quite better than the other, we run out of time (more than 60 seconds) when the problem size is 80000. This indicates that it doesn't really matter the power of the machine if the problem size is too big with a high complexity algorithm.

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

Activity 3. Factor 3 (Implementation environment)

Execution of the same algorithm in Java without optimization:

n	execution time (ms)
10000	38
20000	135
40000	482
80000	1808
160000	6777
320000	25570
640000	OoT

At least for this problem, working in Java seems much better, even without optimization.

Activity 4. Factor 4 (algorithm that is used)

PythonA1.py

n	execution time (ms)
10000	1549
20000	6269
40000	25455
80000	OoT
160000	OoT
320000	OoT
640000	OoT

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

PythonA2.py

n	execution time (ms)
10000	178
20000	647
40000	2412
80000	9157
160000	35208
320000	OoT
640000	OoT

PythonA3.py

n	execution time (ms)
10000	131
20000	419
40000	1464
80000	5448
160000	20879
320000	OoT
640000	OoT

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

JavaA1.java without optimization

n	execution time (ms)
10000	38
20000	135
40000	482
80000	1808
160000	6777
320000	25570
640000	OoT

JavaA2.java without optimization

n	execution time (ms)
10000	35
20000	130
40000	482
80000	1795
160000	6772
320000	26519
640000	OoT

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

JavaA3.java without optimization

n	execution time (ms)
10000	14
20000	52
40000	192
80000	714
160000	2673
320000	10205
640000	40779

JavaA1.java with optimization

n	execution time (ms)
10000	22
20000	31
40000	105
80000	394
160000	1467
320000	5592
640000	21348

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

JavaA2.java with optimization

n	execution time (ms)
10000	16
20000	30
40000	108
80000	402
160000	1474
320000	5585
640000	21140

JavaA3.java with optimization

n	execution time (ms)
10000	8
20000	18
40000	38
80000	137
160000	510
320000	1859
640000	7047

We can conclude a few things with these results:

- Python seems to work faster when using just one module with some functions that calling outside modules to perform these functions, while Java seems not to care about this (the performance is the same).
- As we can see in PythonA3.py and JavaA3 is really important to optimize our algorithms (we are reducing a lot the problem size by simply setting the limit of the loop to $n/2$, but we are not lowering the complexity, still $O(n^2)$), as the programs run much faster.

Algorithmics	Student information	Date	Number of session
	UO: 296503	06/02/2025	0
	Surname: Mulet Alonso		
	Name: Sergio		

-For this specific problem, java is much faster than python, even without optimization. The difference is really big and we must definitely take environment into account.

-Java optimization is impressive, even with the first algorithm (JavaA1.java) it is faster than all the algorithms without optimization. We can also see that JavaA3.java is more than five times faster than itself without optimization