Algorithmics	Student information	Date	Number of session
	UO:300599	11/02/2025	1
	Surname: Alvarez Fernandez		
	Name: Ruben		

SESSION 1.2

TABLE1

N	tLoop1	tLoop2	tLoop3	tLoop4
100	44/10000*10-3 seconds	43/100*10-3 seconds	80/100*10-3 seconds	70*10-3 seconds
200	92/10000*10-3 seconds	88/100*10-3 seconds	333/100*10-3 seconds	466*10-3 seconds
400	195/10000*10-3 seconds	184/100*10-3 seconds	1415/100*10-3 seconds	3461*10-3 seconds
800	445/10000*10-3 seconds	443/100*10-3 seconds	6006/100*10-3 seconds	26884*10-3 seconds
1600	923/10000*10-3 seconds	927/100*10-3 seconds	25265/100*10 seconds -3	Oot
3200	1985/10000*10-3 seconds	1978/100*10-3 seconds	Oot	Oot
6400	4301/10000*10-3 seconds	4228/100*10-3 seconds	Oot	Oot
12800	9633/10000*10-3 seconds	9558/100*10-3 seconds	Oot	Oot
25600	20611/10000*10-3 seconds	20155/100*10-3 seconds	Oot	Oot
51200	Oot	43012/100*-3 seconds	Oot	Oot

Loop1: the complexity is O(log n) since we have a for loop which increases linearly inside a while loop which increases n^2 times each iteration, it makes sense with the results.

Loop2: the algorithm has O(n^2logn) complexity since we have two for loops where iterations increase linearly inside a do while loop which increases n*3 times each iteration, makes sense with the results.

Loop3: the complexity is O(n*log^2(n)) since we have two loops of logarithmic complexity and a loop of linear complexity, it makes sense with the results

Loop4: the complexity is cubic, it makes sense with the results

Algorithmics	Student information	Date	Number of session
	UO:300599	11/02/2025	1
	Surname: Alvarez Fernandez		
	Name: Ruben		

TABLE2 (times in milliseconds and WITHOUT OPTIMIZATION):

N	tLoop5	tLoop6	tLoop7
100	55/10 milliseconds	74 milliseconds	657 milliseconds
200	309/10 milliseconds	740 milliseconds	10359 milliseconds
400	1493/10 milliseconds	6486 milliseconds	Oot
800	7282/10 milliseconds	55695 milliseconds	Oot
1600	34428/10 milliseconds	Oot	Oot
3200	Oot	Oot	Oot
6400	Oot	Oot	Oot

Results are as estimated since the complexity of each loop correlates with the increase of the time execution of each one.

TABLE3 (times in milliseconds and WITHOUT OPTIMIZATION):

100	*tLoop1*	*tLoop2*	*t1/t2*
200	44/10000*10-3	43/100*10-3 seconds	1.02/100 *10-3
400	92/10000*10-3	88/100*10-3 seconds	1.04/100 *10-3
800	195/10000*10-3	184/100*10-3 seconds	1.059/100 *10-3
1600	445/10000*10-3	443/100*10-3 seconds	1.0045/100 *10-3
3200	923/10000*10-3	927/100*10-3 seconds	0.99/100 *10-3
6400	1985/10000*10-3	1978/100*10-3 seconds	0.99/100 *10-3
12800	4301/10000*10-3	4228/100*10-3 seconds	1.01/100 *10-3
25600	9633/10000*10-3	9558/100*10-3 seconds	1.007/100 *10-3
51200	20611/10000*10-3	20155/100*10-3 seconds	1.022/100 *10-3

Results are as expected since both loops have a similar complexity then the difference between both is nearly non existent.

Algorithmics	Student information	Date	Number of session
	UO:300599	11/02/2025	1
	Surname: Alvarez Fernandez		
	Name: Ruben		

TABLE4 (times in milliseconds and WITHOUT OPTIMIZATION):

100	*tLoop3*	*tLoop2*	*T3/t2*
200	80/100*10-3 seconds	43/100*10-3 seconds	1.860465
400	333/100*10-3 seconds	88/100*10-3 seconds	3.784091
800	1415/100*10-3 seconds	184/100*10-3 seconds	7.690217
1600	6006/100*10-3 seconds	443/100*10-3 seconds	13.55756
3200	25265/100*10 seconds -3	927/100*10-3 seconds	27.25458
6400	Oot	1978/100*10-3 seconds	Oot
12800	Oot	4228/100*10-3 seconds	Oot
25600	Oot	9558/100*10-3 seconds	Oot
51200	Oot	20155/100*10-3 seconds	Oot

Results are as expected since the complexity of loop3 is grows faster than the complexity of loop2 , then the result of the division is increasing with each iteration.

	Student information	Date	Number of session
Algorithmics	UO:300599	11/02/2025	1
	Surname: Alvarez Fernandez		
	Name: Ruben		

TABLE5 (times in milliseconds):

N	tLoop4(python) t41	tLoop4(java without optimization) t42	tLoop4(java with optimization) t43	*42/t41*	*43/t42*
100	343 milliseconds	101 milliseconds	70*10-3 seconds	3.66667 milliseconds	1.442857 milliseconds
200	3434 milliseconds	677 Milliseconds	466*10-3 seconds	8.961538 Milliseconds	1.45279 milliseconds
400	54664 Milliseconds	5138 Milliseconds	3461*10-3 seconds	9.874359 milliseconds	1.484542 milliseconds
800	Oot	Oot	26884*10-3 seconds	Oot	Oot
1600	Oot	Oot	Oot	Oot	Oot
3200	Oot	Oot	Oot	Oot	Oot
6400	Oot	Oot	Oot	Oot	Oot

Results are satisfactory since they prove that python is slower than java without Jit optimization because the difference of performance between compilated and interpretated languages and it’s also proven that jit optimization outperforms compilation without it.