


Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz	 Escuela de Ingeniería Informática Universidad de Oviedo	
	Name: Martin		



Activity 1. TWO ALGORITHMS WITH THE SAME COMPLEXITY

For all the following activities, a PC of eight 8GB of RAM and a dual-core CPU of 3.00GHz has been used for making the measurements.

The measurement of Loop1, 2 and 3 were made executing the loop for each size 100 times (nTimes = 100), and then I computed the average time for that size.

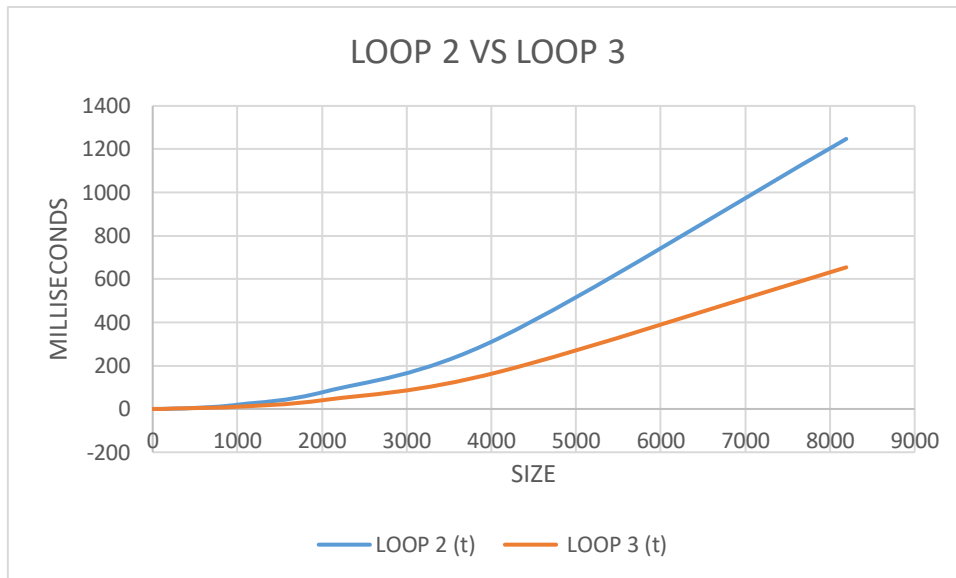
On the other hand, the measurement for Loop 4 and 5 where made only 10 times (nTimes = 10) for each value of N, due to the high complexity of both algorithms.

N	LOOP 2 (t)	LOOP 3 (t)	LOOP2(t)/LOOP3(t)
8	0,07	0,01	7
16	0,07	0,02	3,5
32	0,11	0,08	1,375
64	0,24	0,08	3
128	0,37	0,17	2,176470588
256	1,71	2,11	0,81042654
512	5,04	4,23	1,191489362
1024	20,58	10,86	1,895027624
2048	81,29	42,37	1,918574463
4096	327,88	171,87	1,907720952
8192	1247,43	654,3	1,906510775

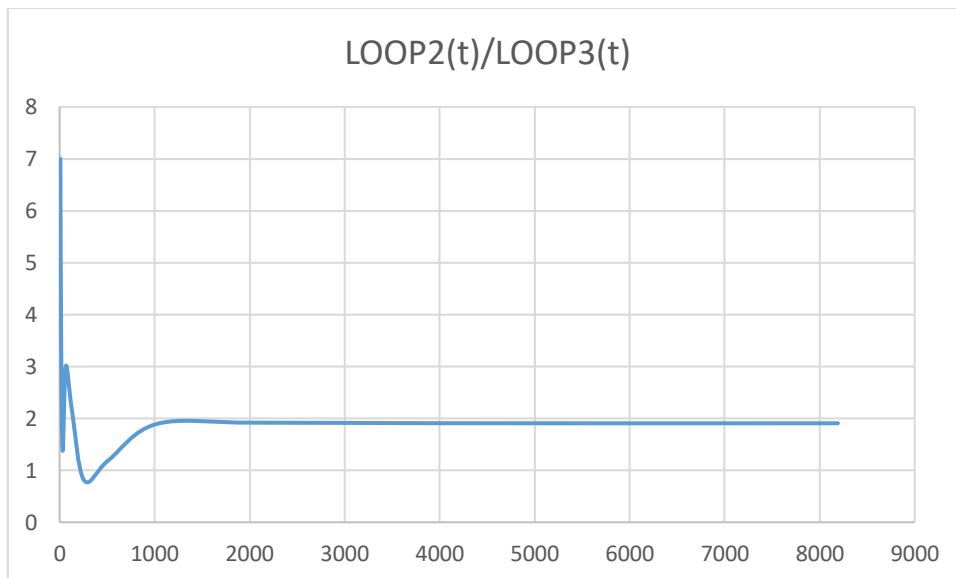
Looking at the code, it can be seen that both Loop 1 and Loop 2 have a temporal complexity of $O(N^2)$, but their division ratio tends to an approximate value of $1.9 \sim 2.0$. So as this is greater than 1, besides both have $O(N^2)$, we can conclude that Loop 3 has a better performance.

We can check that Loop 3 is better in the following chart:

Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz		
	Name: Martin		



In the next chart we can see how the division ratio approaches 1.9:



Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz		
	Name: Martin		

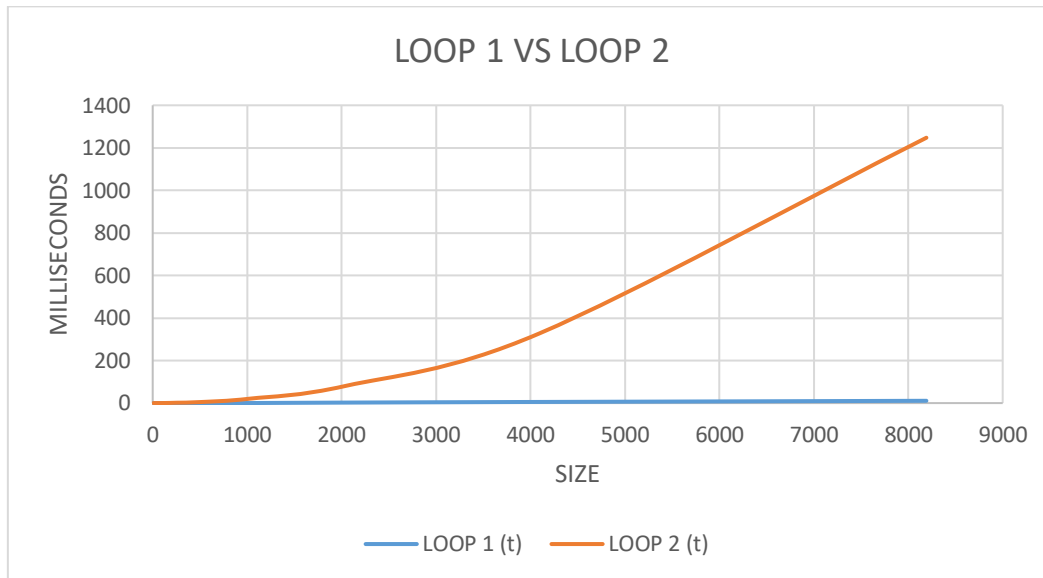
Activity 2. TWO ALGORITHMS WITH DIFFERENT COMPLEXITY

N	LOOP 1 (t)	LOOP 2 (t)	LOOP1(t)/LOOP2(t)
8	0	0,07	0
16	0,02	0,07	0,285714286
32	0,01	0,11	0,090909091
64	0,01	0,24	0,041666667
128	0,01	0,37	0,027027027
256	0,04	1,71	0,023391813
512	0,08	5,04	0,015873016
1024	0,26	20,58	0,012633625
2048	2,43	81,29	0,029892976
4096	5,37	327,88	0,016377943
8192	10,77	1247,43	0,008633751

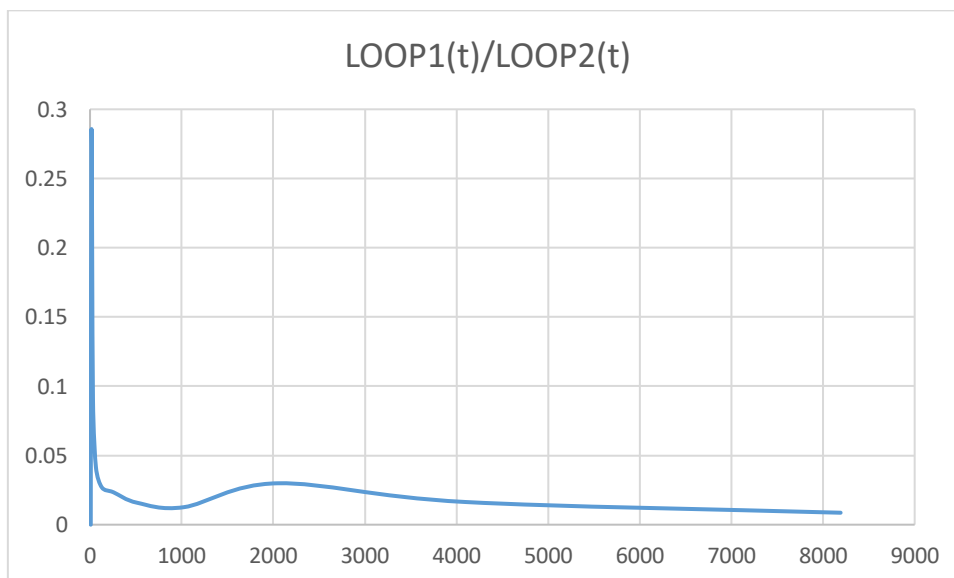
Looking at the code we see that Loop 1 has a complexity of $O(N \cdot \log N)$, and, as mentioned before, Loop 2 has a $O(N^2)$. This explains why their division ratio tends to 0, because the complexity of the numerator is lower than the complexity of the denominator.

Then, it is obvious that Loop 1 is better than Loop 2:

Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz		
	Name: Martin		



Again, in the next chart, we see the division ratio that tends to 0:



Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz		
	Name: Martin		

Activity 3. COMPLEXITY OF OTHER ALGORITHMS

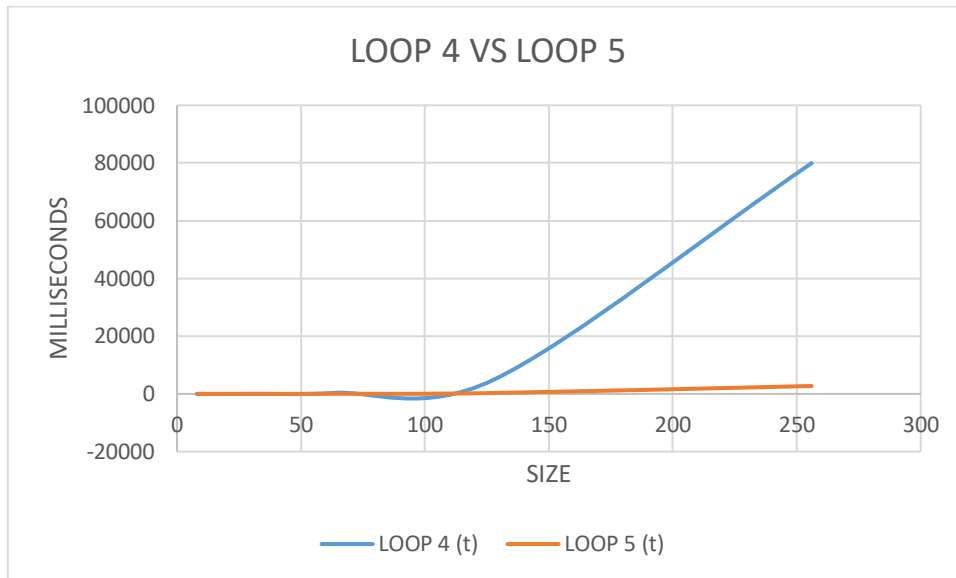
As requested, Loop 4 has a complexity of $O(N^4)$ and Loop 5 $O(N^3 * \log N)$. Due to their high complexity, only 10 measurements were made for each size N, and only a value of N = 256 was reached, since it was taking a lot to make all the measurements.

N	LOOP 4 (t)	LOOP 5 (t)	LOOP4(t)/LOOP5(t)
8	0,9	0,3	3
16	3,9	3,4	1,147058824
32	32,5	5,1	6,37254902
64	385,4	47,3	8,147991543
128	5019,3	311,4	16,11849711
256	79979	2749,9	29,08433034

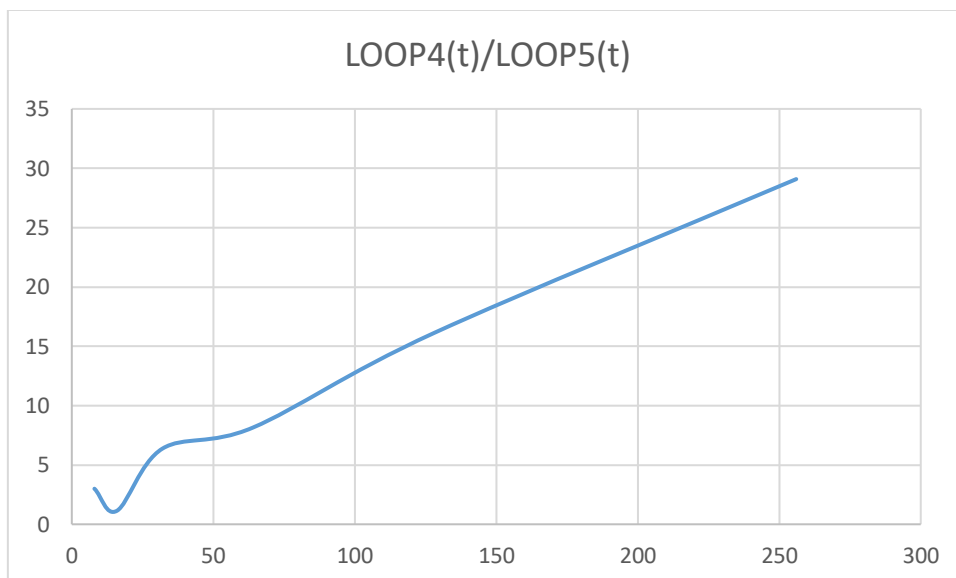
As the complexity of the numerator (Loop 4) is higher than the complexity of the denominator (Loop 5), the division ratio tends to infinite.

We can see in the following chart how Loop 4 increases much faster than Loop 5:

Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz		
	Name: Martin		



Here, the chart for the division ratio that tends to infinite:



Algorithmics	Student information	Date	Number of session
	UO: 276244	19/02/2021	1_2
	Surname: Beltran Diaz		
	Name: Martin		

Activity 4. STUDY OF UNKNOWN.JAVA

N	UNKNOWN
8	0
16	0,1
32	0
64	0,02
128	0,17
256	1,24
512	5,23
1024	27,89
2048	171,58
4096	1344,91
8192	8281,6

The theoretical complexity that I have calculated is, at most, $O(N^3)$, but that is only in the worst case possible, that is, when “i” in the first “for” statement equals “n”, and when at the same time, in the second “for”, “j” equals “i”. This is why the times are not as high as they were with, for example, Loop 5, that had a worse complexity $O(N^3 * \log N)$, but also are not as low times measured with Loop 2 of complexity $O(N^2)$.

So, summing up, the theoretical complexity is $O(N^3)$, but it only reaches that complexity in a few cases (the worst cases).

Here is the table with its growth:

