Algorithmics	Student information	Date	Number of session
	UO: 294866	7/2/2024	1
	Surname: Menendez		
	Name: Javier		

Escuela de Ingeniería Informática
Universidad de Oviedo

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Activity 1. [Iterative Models]

	N	tLoop1	tLoop2	tLoop3	tLoop4
2					
3	100	100	3777	11243	11699
4	200	207	13159	49527	Oot
5	400	436	60103	Oot	Oot
6	800	1035	Oot	Oot	Oot
7	1600	2201	Oot	Oot	Oot
8	3200	4703	Oot	Oot	Oot
9	6400	10007	Oot	Oot	Oot
10	12800	22703	Oot	Oot	Oot
11	25600	56205	Oot	Oot	Oot
12	51200	Oot	Oot	Oot	Oot
13					
14					
15	Rep:	10000			
16					

Yes, the times agree with the theoretical complexities of each algorithm, because the tLoop1 has the better complexity and the tLoop4 the worse one and between them are the rest. For measuring this I used (CPU: Intel® Core ™ i7-4790 CPU @ 3.60GHz, RAM Memory: 8,0 GB DDR3). The complexities are:

Loop1: (n*log(n))

Loop2: (n^2 * log(n))

Loop3: (n^2 * log(n))

Loop4: (n^3)

Algorithmics	Student information	Date	Number of session
	UO: 294866	7/2/2024	1
	Surname: Menendez		
	Name: Javier		

Activity 2. [Creation of iterative models of a complexity.]

N	tLoop5	tLoop6	tLoop7
100	53	257	997
200	230	2324	14968
400	1202	20839	Oot
800	6086	Oot	Oot
1600	26295	Oot	Oot
3200	Oot	Oot	Oot
6400	Oot	Oot	Oot
Rep:	60		

As before when I get a high complexity like in tLoop4 all the measurement were Oot, so I decided to reduce the number of repetitions because the complexities in tLoop5, 6 and 7 are even higher. Repetitions = 60

So now we can see the same as increasing the complexity worse times. And we must consider that the previous measurements were taken with 10000 repetitions while these ones were taken with 20 repetitions. Also, this part was done with a CPU: Intel® Core™ i5-9400 CPU @ 2.90GHz and a RAM: 16,0 GB 2666mHz.

Algorithmics	Student information	Date	Number of session
	UO: 294866	7/2/2024	1
	Surname: Menendez		
	Name: Javier		

Activity 3. [Algorithms with different complexity]

N	tLoop1	tLoop2	tLoop1/tLoop2
100	100	3777	0,026476039
200	207	13159	0,015730679
400	436	60103	0,007254214
800	1035	Oot	
1600	2201	Oot	
3200	4703	Oot	
6400	10007	Oot	
12800	22703	Oot	
25600	56205	Oot	
51200	Oot	Oot	

It obviously agrees with what it was expected because the complexity of the first loop is (n*log(n)) while the complexity of the second loop is (n^2 * log(n)) so is clear that the second loop is worse, and the results talking about time demonstrate it.

Activity 4. [Algorithms with same complexity]

N	tLoop3	tLoop2	t3/t2
100	11243	3777	2,976701086
200	49527	13159	3,763735846
400	Oot	60103	
800	Oot	Oot	
1600	Oot	Oot	
3200	Oot	Oot	
6400	Oot	Oot	
12800	Oot	Oot	
25600	Oot	Oot	
51200	Oot	Oot	

Algorithmics	Student information	Date	Number of session
	UO: 294866	7/2/2024	1
	Surname: Menendez		
	Name: Javier		

Now we checked the time and the division ratio of two algorithms which have the same complexity but the tLoop2 seems to be faster than loop3 this could because the number of iterations or any different issue.

Activity 5. [Different environments for same algorithm]

n	tLoop4(py)-t41	tLoop4(java withoutOpt)-t42	tLoop4(javawithopt)-t43	t42/t41	t43/t42
200	30	85	2	2,833333333	0,02352941
400	241	668	6	2,771784232	0,00898204
800	10345	5256	38	0,508071532	0,00722983
1600	Oot	41755	265	-	0,00634655
3200	Oot	Oot	1576	-	-
6400	Oot	Oot	9972	-	-
Rep:	10				

The measurements for the tLoop4 without optimization are not the same as before because I used another pc and less repetitions, but all this table was at the same moment made for getting better results. The same as in the first practice we can see that the python file takes more time and then the division ratio shows as that java is better than python (in terms of response) and that java improves a lot when the optimization is allowed.