Activity 1. Some iterative models

With a number of 10000 repetitions I got the following results:  
As we are using this number of repetitions time units are milliseconds to the power of

10-5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | tLoop1 | tLoop2 | tLoop3 | tLoop4 |
| 100 | 50 | 1649 | 8337 | 6840 |
| 200 | 96 | 6265 | 36415 | 50121 |
| 400 | 206 | 29324 | OoT | OoT |
| 800 | 485 | OoT | OoT | OoT |
| 1600 | 991 | OoT | OoT | OoT |
| 3200 | 2177 | OoT | OoT | OoT |
| 6400 | 4684 | OoT | OoT | OoT |
| 12800 | 10632 | OoT | OoT | OoT |
| 25600 | 22472 | OoT | OoT | OoT |
| 51200 | 47339 | OoT | OoT | OoT |

For a better measurement I will use less number of repetitions for Loop2, Loop3 and Loop4.

Loop2: 1000 repetitions, milliseconds to the power of 10-3, that is microseconds

|  |  |
| --- | --- |
| N | tLoop2 |
| 100 | 185 |
| 200 | 659 |
| 400 | 3092 |
| 800 | 14437 |
| 1600 | 55229 |
| 3200 | OoT |
| 6400 | OoT |
| 12800 | OoT |
| 25600 | OoT |
| 51200 | OoT |

For Loop3 and Loop4 I will use 100 repetitions, that is milliseconds to the power of 10-2

|  |  |  |
| --- | --- | --- |
| N | tLoop3 | tLoop4 |
| 100 | 89 | 71 |
| 200 | 417 | 498 |
| 400 | 1730 | 3683 |
| 800 | 7152 | 29091 |
| 1600 | 28431 | OoT |
| 3200 | OoT | OoT |
| 6400 | OoT | OoT |
| 12800 | OoT | OoT |
| 25600 | OoT | OoT |
| 51200 | OoT | OoT |

Now let’s take everything together, in nanoseconds:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | tLoop1 | tLoop2 | tLoop3 | tLoop4 |
| 100 | 5.0 | 185 | 890 | 710 |
| 200 | 9.6 | 659 | 4170 | 4980 |
| 400 | 20.6 | 3092 | 17300 | 36830 |
| 800 | 48.5 | 14437 | 71520 | 290910 |
| 1600 | 99.1 | 55229 | 284310 | OoT |
| 3200 | 217.7 | OoT | OoT | OoT |
| 6400 | 468.4 | OoT | OoT | OoT |
| 12800 | 1063.2 | OoT | OoT | OoT |
| 25600 | 22472 | OoT | OoT | OoT |
| 51200 | 47339 | OoT | OoT | OoT |

If we look at the code, we can see that Loop1 has a complexity of nlogn2, which makes sense with the measured times.

The complexity of Loop2 is O(n2log3n) while Loop3 is O(n3log2n), both can be reduced to O(n2logn). Because of the complexity in detail, Loop3 get higher times than Loop2.

Finally, the complexity of Loop4 is O(n3) which is clearly shown in the times, for instance when n = 400 we get a time of 36830 and with n = 800 we get 290910, the times got almost 23 times higher (times 8).

Activity 2. Creation of iterative models of a given time complexity

I am using 10 as number of repetitions so milliseconds to the power of 10-1

|  |  |  |  |
| --- | --- | --- | --- |
| N | tLoop5 | tLoop6 | tLoop7 |
| 100 | 50 | 102 | 4184 |
| 200 | 208 | 830 | OoT |
| 400 | 1007 | 7316 | OoT |
| 800 | 4782 | OoT | OoT |
| 1600 | 22488 | OoT | OoT |
| 3200 | OoT | OoT | OoT |
| 6400 | OoT | OoT | OoT |

As Loop7 is OoT by n = 200 I will measure it with just one repetition:

|  |  |
| --- | --- |
| N | tLoop7 |
| 100 | 411 |
| 200 | 6638 |
| 400 | OoT |
| 800 | OoT |
| 1600 | OoT |
| 3200 | OoT |
| 6400 | OoT |

As the complexity of this method is n4, the times get too big too soon, but we can see that they follow the complexity, 6638/411 is almost 16, which is 24.

Putting it all together we have, in milliseconds:

|  |  |  |  |
| --- | --- | --- | --- |
| N | tLoop5 | tLoop6 | tLoop7 |
| 100 | 5.0 | 10.2 | 411 |
| 200 | 20.8 | 83.0 | 6638 |
| 400 | 100.7 | 731.6 | OoT |
| 800 | 478.2 | OoT | OoT |
| 1600 | 2248.8 | OoT | OoT |
| 3200 | OoT | OoT | OoT |
| 6400 | OoT | OoT | OoT |

Activity 3. Two algorithms with different complexity

|  |  |  |  |
| --- | --- | --- | --- |
| N | tLoop1 | tLoop2 | t1/t2 |
| 100 | 5 | 185 | 0,027027 |
| 200 | 9,6 | 659 | 0,0145675 |
| 400 | 20,6 | 3092 | 0,0066624 |
| 800 | 48,5 | 14437 | 0,0033594 |
| 1600 | 99,1 | 55229 | 0,0017943 |
| 3200 | 217,7 | OoT |  |
| 6400 | 468,4 | OoT |  |
| 12800 | 1063,2 | OoT |  |
| 25600 | 22472 | OoT |  |
| 51200 | 47339 | OoT |  |

The result of dividing t1 by t2 is lower than 1, then we know that tLoop1 is better, this makes sense since it has a better complexity also.

As we can see the quotient gets divided by more or less two in each problem size increase, this happens because while t1 has a complexity of O(**n**logn2), Loop2 has O(**n**2logn).

Activity 4. Two algorithms with the same complexity