Activity 1. Iterative models

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | tLoop1 | tLoop2 | tLoop3 | tLoop4 |
| 100 | 0.0072 | 0.257 | 0.89 | 1.01 |
| 200 | 0.0151 | 1.024 | 3.6 | 7.41 |
| 400 | 0.0336 | 4.644 | 16.69 | 55.96 |
| 800 | 0.077 | 21.24 | 69 | 469 |
| 1600 | 0.1663 | 90.3 | 298.54 | 3583 |
| 3200 | 0.3435 | 395.9 | 1399 | 28014 |
| 6400 | 0.7518 | 1538.5 | 6015 | OoT |
| 12800 | 1.7211 | 7008 | 24777 | OoT |
| 25600 | 3.5645 | 33278 | OoT | OoT |
| 51200 | 7.522 | OoT | OoT | OoT |

Their complexities are:

Loop1: O(nlogn)

Loop2: O(n2logn)

Loop3: O(n2logn)

Loop4: O(n3)

The times match their complexity as the size is multiplied by 2 so the times gets multiplied by 2log2, 4log2, 4log2 and 8 respectively.

Activity 2. Creation of iterative models

|  |  |  |  |
| --- | --- | --- | --- |
| N | tLoop5 | tLoop6 | tLoop7 |
| 100 | 2.15 | 81 | 102 |
| 200 | 9.26 | 713 | 1542 |
| 400 | 47.8 | 6096 | 23018 |
| 800 | 241 | 531645 | OoT |
| 1600 | 1037 | OoT | OoT |
| 3200 | 5623 | OoT | OoT |
| 6400 | 23556 | OoT | OoT |

Their complexities are:

Loop5: O(n2log2n)

Loop6: O(n3logn)

Loop7: O(n4)

The times match their complexity as the size is multiplied by 2 so the times gets multiplied by 4log22, 8log2 and 16 respectively.

Activity 3. Two algorithms with different complexity

|  |  |  |  |
| --- | --- | --- | --- |
| n | tLoop1 | tLoop2 | t1/t2 |
| 100 | 0.0012 | 0.257 | 0.028015564 |
| 200 | 0.0151 | 1.024 | 0.014746094 |
| 400 | 0.0336 | 4.644 | 0.007235142 |
| 800 | 0.077 | 21.24 | 0.003625235 |
| 1600 | 0.1663 | 90.3 | 0.001841639 |
| 3200 | 0.3435 | 395.9 | 0.000867643 |
| 6400 | 0.7518 | 1538.5 | 0.000488658 |
| 12800 | 1.7211 | 7008 | 0.000245591 |
| 25600 | 3.5645 | 33278 | 0.000107113 |
| 51200 | 7.522 | OoT | - |

As the quotient is less than 1, that means that the Loop1 has a better performance than Loop2, which matches the theoretical performance because O(nlogn) is better than O(n2logn).

Activity 4. Two algorithms with the same complexity

|  |  |  |  |
| --- | --- | --- | --- |
| n | tLoop3 | tLoop2 | t3/t2 |
| 100 | 0.89 | 0.257 | 3.463035019 |
| 200 | 3.6 | 1.024 | 3.515625 |
| 400 | 16.69 | 4.644 | 3.593884582 |
| 800 | 69 | 21.24 | 3.248587571 |
| 1600 | 298.54 | 90.3 | 3.306090808 |
| 3200 | 1399 | 395.9 | 3.533720637 |
| 6400 | 6015 | 1538.5 | 3.909652259 |
| 12800 | 24777 | 7008 | 3.535530822 |
| 25600 | OoT | 33278 | - |
| 51200 | OoT | OoT | - |

As the quotient is higher than 1, that means that the Loop2 has a better performance than Loop3. Even though they have the same complexity, O(n2logn), Loop2 is better implemented, which result in an implementation constant around 3.4.

Activity 5. Same algorithm in different development environments

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | tLoop4  (Python) -t4.1 | tLoop4  (Java without optimization) – t4.2 | tLoop4  (Java with optimization) – t4.3 | t4.2/t4.1 | t4.3/t4.2 |
| 200 | 31 | 7.41 | 0.0913 | 0.239032258 | 0.019009901 |
| 400 | 234 | 55.96 | 0.4238 | 0.239145299 | 0.012321188 |
| 800 | 1984 | 469 | 2.6113 | 0.236391129 | 0.007573267 |
| 1600 | 16548 | 3583 | 19.33 | 0.216521634 | 0.005567804 |
| 3200 | OoT | 28014 | 132.26 | - | 0.004721211 |
| 6400 | OoT | OoT | 993 | - | - |

As the quotient is lower than 1 in both cases, that means that Java with optimization has better performance than Java without optimization, which also has better performance than Python. As Python is a higher-level language than Java, the performance of Java is better, and the optimization in Java is design to improve performance, so it is logical that the performance improves when using the optimization. As the size of the problem gets bigger, Java optimization outperforms Java without it by even more. That is because the optimization is able to save more time if there are more operations to perform.