OPTIONAL PROJECT 4

ANALYSIS REPORT

QUESTION 1:

|  |  |  |
| --- | --- | --- |
| **Input size** | **Quick sort Running time (seconds)** | **Dual Pivot Quick Sort Running Time (seconds)** |
| 2^10 | 0.020092912 | 0.005793434 |
| 2^11 | 0.015924565 | 0.005601844 |
| 2^12 | 0.024262969 | 0.021531541 |
| 2^13 | 0.032642429 | 0.023858409 |
| 2^14 | 0.045647398 | 0.024697467 |
| 2^15 | 0.089709482 | 0.067281149 |
| 2^20 | 2.195746806 | 1.726274271 |
| 2^21 | 4.233250811 | 3.76262147 |
| 2^22 | 8.185044756 | 6.792845941 |
| 2^23 | 17.99926743 | 14.93224971 |
| 2^24 | 39.69546349 | 32.2732536 |
| 2^25 | 93.63087357 | 72.86503613 |
| **When the elements are not disticnt** | | |
| **Input size** | **Quick sort Running time (seconds)** | **Dual Pivot Quick Sort Running Time (seconds)** |
| 2^10 | 0.00003635 | 0.000017106 |
| 2^11 | 0.000032501 | 0.000016251 |

From the analysis it is evident that dual pivot quick sort is more efficient that quick sort for any case. When the input size is larger or when the input has duplicates, the dual pivot quick sort is much more fast and efficient than quick sort.

QUESTION 2:

|  |  |  |  |
| --- | --- | --- | --- |
| **Input size** | **K value** | **Select algorithm Running time (seconds)** | **Priority-queue-based algorithm Running Time (seconds)** |
| 2^10 | 100 | 0.004393293 | 0.008109182 |
| 2^10 | 200 | 0.002635206 | 0.010653298 |
| 2^20 | 200 | 0.209546785 | 0.08336523 |
| 2^20 | 1000 | 0.237154439 | 0.193207395 |
| 2^25 | 100 | 2.327220157 | 0.651898764 |

We can clearly see that as the input size and the k value increases, select algorithm takes more time than priority-queue based algorithm.