Group 26

Sharath Chalya Nagaraju

Ankitha Karunakar Shetty

Sandeep Avula

Short Project 5

*Partition algorithms*

Short Project 5

Partition algorithms

**Problem 1**

Run time comparison of 2 version of partition, note that all the sorting were done using generics.

|  |  |  |
| --- | --- | --- |
| Input size | Runtime (milliseconds) | |
| Partition type 1 | Partition Type 2 |
| 1000000 | 415 | 301 |
| 5000000 | 2439 | 2128 |
| 8000000 | 3554 | 3512 |
| 10000000 | 5240 | 4383 |
| Descending order | | |
| 1000000 | 170 | 131 |
| 5000000 | 954 | 611 |
| 10000000 | 1851 | 1332 |

We can see that the version 2 of partition consistently performs better than version 1. The algorithm is actually performing much better for the array sorted in descending order.

**Problem 2**

Dual Partition was also implemented using generics, below are the results

|  |  |  |
| --- | --- | --- |
| Input size | Runtime (milliseconds) | |
| Dual Partition | Partition Type 2 |
| 1000000 | 420 | 401 |
| 5000000 | 2135 | 2385 |
| 8000000 | 3236 | 3576 |
| 10000000 | 4711 | 4249 |
| With duplicates | | |
| 5000000 | 2255 | 2675 |
| 10000000 | 3889 | 5293 |

Dual partition implementation clearly outperforms normal quick sort when there are duplicates in the input.

**Problem 3**

Below are the run times for different K and N

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Array Size (N) | K | Runtime in milliseconds | | |
| Max heap | Min Heap | Select |
| 1000000 | 100 | 146 | 38 | 82 |
| 1000000 | 1000 | 147 | 54 | 62 |
| 10000000 | 1000000 | 2103 | 2378 | 1639 |

Following are the observations

1. We can see that the implementation of select algorithm using generics is adding some run time so it looks slower than min heap implementation.
2. If we increase the value of k then min heap implementation takes a hit as the log(k) factor increases in its nlog(k) runtime. While the implementation using selection sort will not vary with the variation in k.
3. Whereas Max heap always takes more time since its runtime is nlogn

**Problem 4**

|  |  |  |
| --- | --- | --- |
| Input size | Runtime (milliseconds) | |
| Dual Partition | Merge Sort |
| With duplicates | | |
| 5000000 | 1630 | 2004 |
| 8000000 | 2765 | 3639 |
| 10000000 | 3988 | 4396 |
| Without duplicates | | |
| 5000000 | 3928 | 2016 |
| 10000000 | 4287 | 4254 |

With duplicates dual partition algorithm clearly beats the optimized merge sort, and without duplicates merge seems to work well

**Problem 5**

|  |  |  |
| --- | --- | --- |
| Input Size M=1000000 | External MergeSort | External QuickSort |
| 2M | 7360ms | 8523ms |
| 4M | 15548ms | 27353ms |
| 8M | 36690ms | 69396ms |
| 10M | 39703ms | 179160ms |

External MergeSort clearly beats the running time of External QuickSort.