Report for P4, ikn3

I don’t really know how the report is supposed to go. I am just going to graph the mean and median and show the trend and explain all the graphs and tables I put up.

For all of the graphs, length of array will be on the x axis and running time in nanoseconds will be on the y axis.

QUICKSORT FOR SORTED ARRAYS:

|  |  |  |
| --- | --- | --- |
| Length of  Sorted Array | Mean | Median |
| 1000 | 349560 | 153215 |
| 10000 | 254503 | 255114 |
| 100000 | 2895079 | 2885061 |
| 1000000 | 18160256 | 17607998 |

QuickSort for Sorted Array Table

Graph for QuickSort

We see that through both through the table and graph that there is almost a linear increase in running time as the number of elements increase. This can also be illustrated by the equations on the graph that depict a linear relationship with an R squared value of close to 1.

MERGESORT FOR SORTED ARRAYS

|  |  |  |
| --- | --- | --- |
| Length of Sorted Array | Mean | Median |
| 1000 | 174474.3 | 105564 |
| 10000 | 560932.7 | 561910 |
| 100000 | 7991496 | 6977882 |
| 1000000 | 79745487 | 74774063 |

MergeSort for Sorted Array Table

Graph for MergeSort

Again, we see much the same thing for Merge Sort with the number of elements linearly increasing with the running time for the sorted arrays.

|  |  |  |
| --- | --- | --- |
| Length of Sorted Array | Mean | Median |
| 1000 | 235198.3333 | 143685 |
| 10000 | 1182712.333 | 1166707 |
| 100000 | 16147812.67 | 14493848 |
| 1000000 | 163269667.7 | 163009422 |

HeapSort for Sorted Array Table

HeapSort Graph

It is pretty much the same as before; as the number of elements linearly increases, the running time for the sorted arrays using heapsort linearly increases.

I’m not going to bother for the reverse sorted arrays and the random arrays as they follow much the same pattern: the linear increase of both the running time with the number of elements. Instead, I’m going to compare the running time of the three sorting methods.

|  |  |  |  |
| --- | --- | --- | --- |
| Length of Sorted Array | HeapSort | QuickSort | MergeSort |
| 1000 | 235198.3333 | 349560 | 174474.3333 |
| 10000 | 1182712.333 | 254503 | 560932.6667 |
| 100000 | 16147812.67 | 2895079 | 7991495.667 |
| 1000000 | 163269667.7 | 18160256 | 79745487 |

Graph Sorted Arrays Using the Methods (uses mean)

As we see here, the HeapSort method by far takes the most time to run through a sorted array followed by MergeSort and then QuickSort, which zooms through the sorted array.

|  |  |  |  |
| --- | --- | --- | --- |
| Length of Reverse Sorted Array | QuickSort | MergeSort | HeapSort |
| 1000 | 15273 | 46184 | 84671 |
| 10000 | 159446 | 580238 | 1120522 |
| 100000 | 1653231 | 6625389 | 13282789 |
| 1000000 | 17044255 | 87816155 | 16158931 |

Graph Reverse Sorted Arrays Using the Methods (uses mean)

We see here that the MergeSort takes a ton of time to process the reverse sorted array, while the heapsort and quicksort are fairly even with the heapsort being slower at first.

|  |  |  |  |
| --- | --- | --- | --- |
| Length of Random Array | QuickSort | MergeSort | HeapSort |
| 1000 | 60992 | 93212 | 108203 |
| 10000 | 685618 | 1154757 | 1416249 |
| 100000 | 7852343 | 13334325 | 17876637 |
| 1000000 | 89683717 | 166099416 | 261819622 |

Graph Random Arrays Using the Methods (uses mean)

We see that quicksort is the fastest running method followed by mergesort in the middle and heapsort dead last. Now we move onto the specific methods for the various arrays.

|  |  |  |  |
| --- | --- | --- | --- |
| Length of Array | Random | Reverse | Sorted |
| 1000 | 60992 | 15273 | 349560 |
| 10000 | 685618 | 159446 | 254503 |
| 100000 | 7852343 | 1653231 | 2895079 |
| 1000000 | 89683717 | 17044255 | 18160256 |

QuickSort on Various Arrays

Graph for QuickSort on Various Arrays

We see here that Quicksort works equally well on Reverse and sorted arrays while experiencing a big increase in running time for the random array.

|  |  |  |  |
| --- | --- | --- | --- |
| Length of Array | Random | Reverse | Sorted |
| 1000 | 93212 | 46184 | 349560 |
| 10000 | 1154757 | 580238 | 254503 |
| 100000 | 13334325 | 6625389 | 2895079 |
| 1000000 | 166099416 | 87816155 | 18160256 |

MergeSort on Various Arrays

Graph for MergeSort on Various Arrays

We see here that the mergesort works best on sorted arrays, experiences a dropoff in reverse arrays, and by far performs the worst on random arrays.

|  |  |  |  |
| --- | --- | --- | --- |
| Length of Array | Random | Reverse | Sorted |
| 1000 | 108203 | 84671 | 235198.3333 |
| 10000 | 1416249 | 1120522 | 1182712.333 |
| 100000 | 17876637 | 13282789 | 16147812.67 |
| 1000000 | 261819622 | 16158931 | 163269667.7 |

HeapSort on Various Arrays Table

Graph for Heapsort On Various Arrays

We see here that heapsort is extremely good on reverse arrays, is pretty bad at reverse arrays but by far is the worst with random arrays.

**Conclusion:**

Quicksort is usually the best method to use for sorting any kind of array, whether it is sorted, reverse sorted, or completely random. The random arrays take the most time to sort no matter which method. Therefore, use quicksort and avoid random arrays.