**Exploring Sorting Runtimes (Activity 4) Thilina Prasad Athukorala (Curtin ID: 21038875)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sorting Method** | **Run time for 100 Elements**  **(ns)(10-9s)** | **Run time for 500 Elements**  **(ns)(10-9s)** | **Run time for 2000 Elements**  **(ns)(10-9s)** | **Run time for**  **10000 Elements**  **(ns)(10-9s)** |
| Bubble Sort  (Ascending) | 99.66666 | 280.6666667 | 1566.666667 | 28627.00 |
| Bubble Sort  (Descending) | 155.3333333 | 341.3333333 | 2578.333333 | 57517.00 |
| Bubble Sort  (In random order) | 133.00 | 293.6666667 | 2881.333333 | 65774.66667 |
| Bubble Sort  [nearly sorted  (10% moved)] | 99.00 | 336.00 | 2040.666667 | 39911.33333 |
|  |  |  |  |  |
| Insertion Sort  (Ascending) | 1.33333333 | 9.00 | 39.66666667 | 185.00 |
| Insertion Sort  (Descending) | 136.00 | 295.00 | 1899.666667 | 26787.66667 |
| Insertion Sort  (In random order) | 68.00 | 185.33333 | 2175.666667 | 13711.33333 |
| Insertion Sort  [nearly sorted  (10% moved)] | 11.00 | 240.3333333 | 325.00 | 2557.666667 |
|  |  |  |  |  |
| Selection Sort  (Ascending) | 78.00 | 244.6666667 | 1241.666666 | 7668.666666 |
| Selection Sort  (Descending) | 80.00 | 257.6666666 | 3501.333333 | 28250.66666 |
| Selection Sort  (In random order) | 78.33333333 | 201.6666667 | 1732.00 | 27228.66667 |
| Selection Sort  [nearly sorted  (10% moved)] | 77.00 | 262.3333333 | 1922.00 | 26667.33333 |
|  |  |  |  |  |
| Merge Sort  (Ascending) | 52.333333 | 157.33333333 | 200.3333333 | 1129.66666666 |
| Merge Sort  (Descending) | 59.333333 | 148.0 | 200.666666 | 1042.333333 |
| Merge Sort  (In random order) | 41.33333333 | 65.333333333 | 195.333333 | 1069.0 |
| Merge Sort  [nearly sorted  (10% moved)] | 67.333333 | 147.333333 | 225.33333333 | 1173.33333333 |
| **Quick Sort (left pivot)** |  |  |  |  |
| Quick Sort  (Ascending) | 91.0 | 295.0 | 399.0 | 12377.0 |
| Quick Sort  (In random order) | 33.3333 | 38.333333 | 112.3333333 | 659.333333 |
| Quick Sort  [nearly sorted  (10% moved)] | 70.6666 | 134.6666666 | 309.33333 | 3226.6666 |
| **Quick Sort**  **(Median pivot)** |  |  |  |  |
| Quick Sort  (Ascending) | 66.66666666 | 185.66666666 | 3700.666666 | 10525.0 |
| Quick Sort  (In random order) | 30.66666666 | 45.0 | 121.0 | 629.33333333 |
| Quick Sort  [nearly sorted  (10% moved)] | 55.6666666 | 147.3333333 | 288.3333333 | 627.6666666666666 |
|  |  |  |  |  |

Exploring Sorting Runtimes (Activity 4)

Thilina Prasad Athukorala

Curtin ID: 21038875 | SLIIT ID : SI22458056

**Discussion**

Quick sort (Left-most pivot)- It picks the leftmost element as the pivot and partitions the given array around the picked pivot. There are many different versions of quickSort that pick pivot in different ways. The key process in quickSort is a partition(). The target of partitions is, given an array and an element x of an array as the pivot, put x at its correct position in a sorted array and put all smaller elements (smaller than x) before x, and put all greater elements (greater than x) after x.This is Relatively a slower approach than using a median pivot since the partition cannot be guaranteed to be done at the exact center when the leftmost pivot is selected, shows a time complexity of O(N log N) and the sorting method is unstable.

Quick sort (Median- pivot) – It is the same procedure as of the above-mentioned method but here the initial pivot is selected by using the median which could split the array into exact half of it. It is more faster than using the leftmost pivot and has a time complexity of O(N log N) and unstable.

Merge Sort- it is a recursive algorithm which continuously splits the array in half until it cannot be further divided. If the array becomes empty or has only one element left, the dividing will stop (it is the base case to stop the recursion). If the array has multiple elements, splits the array into halves and recursively invoke the merge sort on each of the halves. Finally, when both halves are sorted, the merge operation is applied. Merge operation is the process of taking two smaller sorted arrays and combining them to eventually make a larger one. When comparing the obtained results merge sort is really fast in sorting and is a stable sorting method. It has Best, average and worst time complexity of O(n log N).

When comparing the types is quick sorting and merge sorting, merge sorting is shown to be very fast compared to the quick sorting methods. But choosing the median for splitting initially gives the quick sorting a boosting property to split in half and continue.