**THEORETICAL ANALYSIS:**

The sorting techniques which are being examined in this are Merge, Heap, Inseryion and 3 way merge sorts.

The sorting algorithms are applied on input files of lengths 20000, 40000, 80000, 160000, 320000. The data are present in different formats such as randomly present, sorted integers, reverse sorted integers, equal integers

INSERTION SORT:

Insertion sort works best for the smaller dataset. As the numer of elements increases the time taken to sort also increases. It takes around O(1) if the data is already sorted.

As we know the insertion sort has a time complexity of O(n2) which is for the worst case scenario(i.e if the data is reverse sorted).

The insertion sort is called the in-place algorithm which is no extra space is needed for the sorting of integers.

But the algorithm is not recommended as it takes O(n2) time complexity which is not appreciable.

MERGE SORT:

Merge sort is one of the fast sorting algorithms. The merge sort takes less amount of time to sort the integers irrespective of their order. The merge sort algorithm takes O(nlogn) to sort the integers.

Though it takes less time to sort, it is not space effiecient as it needs 2 arrays to sort the integers. This will take more space to get the desired output

HEAP SORT:

The Heap sort is the algorithm which is in-place as well as the time complexity is similar to merge sort i.e O(nlogn). The algorithm sorts the integers in less amount of comparitively.

3 WAY MERGE SORT:

The 3 way merge is similar to 2-way merge but the algorithm divides the array into 3 parts every time the recursive function is called. The 3 way merge sort also sorts the integers in less amount of time just like the Merge and Heap sort. It takes O(nlogn) amount of time to sort the inetgers.

**EXPERIMENTAL ANALYSIS AND DISCUSSION**

With the help of the results obtained, the following points have been discovered

* The insertion sort takes more amount of time than the other algorithms(Merge, 3 way Merge, Heap) for the random, reverse sorted integers. It takes O(n2) for both random and reverse sorted integers.
* Merge sort(and 3 way merge sort) does equal number of comparisons on the integers irrespective of their order(sorted, random, equal,reverse sorted)
* 3 way merge sort makes more number of comparisons than the 2 way merge sort.
* Merge sort takes almost same amount of time for the input file of equal number of elements irrespective of their order.

**Results:**

**MERGE SORT**

|  |  |  |
| --- | --- | --- |
| **Number of elements** | **Number of comparisons** | **Time(in milliseconds)** |
| **20000(Random)** | **287232** | **152** |
| **20000(Equal)** | **287232** | **152** |
| **20000(Sorted)** | **287232** | **152** |
| **20000(Reverse sorted)** | **287232** | **168** |
| **40000(Random)** | **614464** | **312** |
| **40000(Equal)** | **614464** | **312** |
| **40000(Sorted)** | **614464** | **312** |
| **40000(Reverse sorted)** | **614464** | **304** |
| **80000(Random)** | **1308928** | **664** |
| **80000(Equal)** | **1308928** | **632** |
| **80000(Sorted)** | **1308928** | **640** |
| **80000(Reverse sorted)** | **1308928** | **640** |
| **160000(Random)** | **2777856** | **1440** |
| **160000(Equal)** | **2777856** | **1344** |
| **160000(Sorted)** | **2777856** | **1336** |
| **160000(Reverse sorted)** | **2777856** | **1344** |
| **320000(Random)** | **5875712** | **3056** |
| **320000(Equal)** | **5875712** | **2824** |
| **320000(Sorted)** | **5875712** | **2792** |
| **320000(Reverse sorted)** | **5875712** | **2841** |

**HEAP SORT**

|  |  |  |
| --- | --- | --- |
| **Number of elements** | **Number of Comparisons** | **Time(in milliseconds)** |
| **20000(Random)** | **759332** | **232** |
| **20000(Equal)** | **59994** | **23** |
| **20000(Sorted)** | **791741** | **232** |
| **20000(Reverse sorted)** | **727617** | **208** |
| **40000(Random)** | **1638000** | **504** |
| **40000(Equal)** | **119994** | **40** |
| **40000(Sorted)** | **1703757** | **504** |
| **40000(Reverse sorted)** | **1575420** | **448** |
| **80000(Random)** | **3516179** | **1088** |
| **80000(Equal)** | **239994** | **88** |
| **80000(Sorted)** | **3650653** | **1104** |
| **80000(Reverse sorted)** | **3387052** | **992** |
| **160000(Random)** | **7512180** | **2400** |
| **160000(Equal)** | **479994** | **176** |
| **160000(Sorted)** | **7783321** | **2266** |
| **160000(Reverse sorted)** | **7253430** | **2112** |
| **320000(Random)** | **15986883** | **5200** |
| **320000(Equal)** | **959994** | **344** |
| **320000(Sorted)** | **16528090** | **4936** |
| **320000(Reverse sorted)** | **15471481** | **5232** |

**3 WAY MERGE SORT**

|  |  |  |
| --- | --- | --- |
| **Number of elements** | **Number of comparisons** | **Time(in milliseconds)** |
| **20000(Random)** | **305810** | **178** |
| **20000** | **305810** | **177** |
| **20000** | **305810** | **182** |
| **20000** | **305810** | **179** |
| **40000(Random)** | **651946** | **381** |
| **40000** | **651946** | **373** |
| **40000** | **651946** | **383** |
| **40000** | **651946** | **363** |
| **80000(Random)** | **1414552** | **864** |
| **80000** | **1414552** | **858** |
| **80000** | **1414552** | **852** |
| **80000** | **1414552** | **834** |
| **160000(Random)** | **2914596** | **1645** |
| **160000** | **2914596** | **1607** |
| **160000** | **2914596** | **1607** |
| **160000** | **2914596** | **1568** |
| **320000(Random)** | **6278736** | **3793** |
| **320000** | **6278736** | **3654** |
| **320000** | **6278736** | **3650** |
| **320000** | **6278736** | **3567** |

**INSERTION SORT**

|  |  |  |
| --- | --- | --- |
| **Number of elements** | **Number of comparisons** | **Time(in milliseconds)** |
| **20000(Random)** | **99882485** | **38864** |
| **20000** | **19999** | **115** |
| **20000** | **19999** | **113** |
| **20000** | **199989801** | **77142** |
| **40000(Random)** | **399545583** | **158398** |
| **40000** | **39999** | **224** |
| **40000** | **39999** | **232** |
| **40000** | **799979225** | **329957** |
| **80000(Random)** | **16066977941** | **671829** |
| **80000** | **79999** | **701** |
| **80000** | **79999** | **475** |
| **80000** | **3199956766** | **1729292** |
| **160000(Random)** | **6389006966** | **3110861** |
| **160000** | **159999** | **917** |
| **160000** | **159999** | **986** |
| **160000** | **12799907197** | **5193885** |
| **320000(Random)** | **25598756988** | **14374173** |
| **320000** | **319999** | **1840** |
| **320000** | **319999** | **1919** |
| **320000** | **51199789183** | **20792946** |

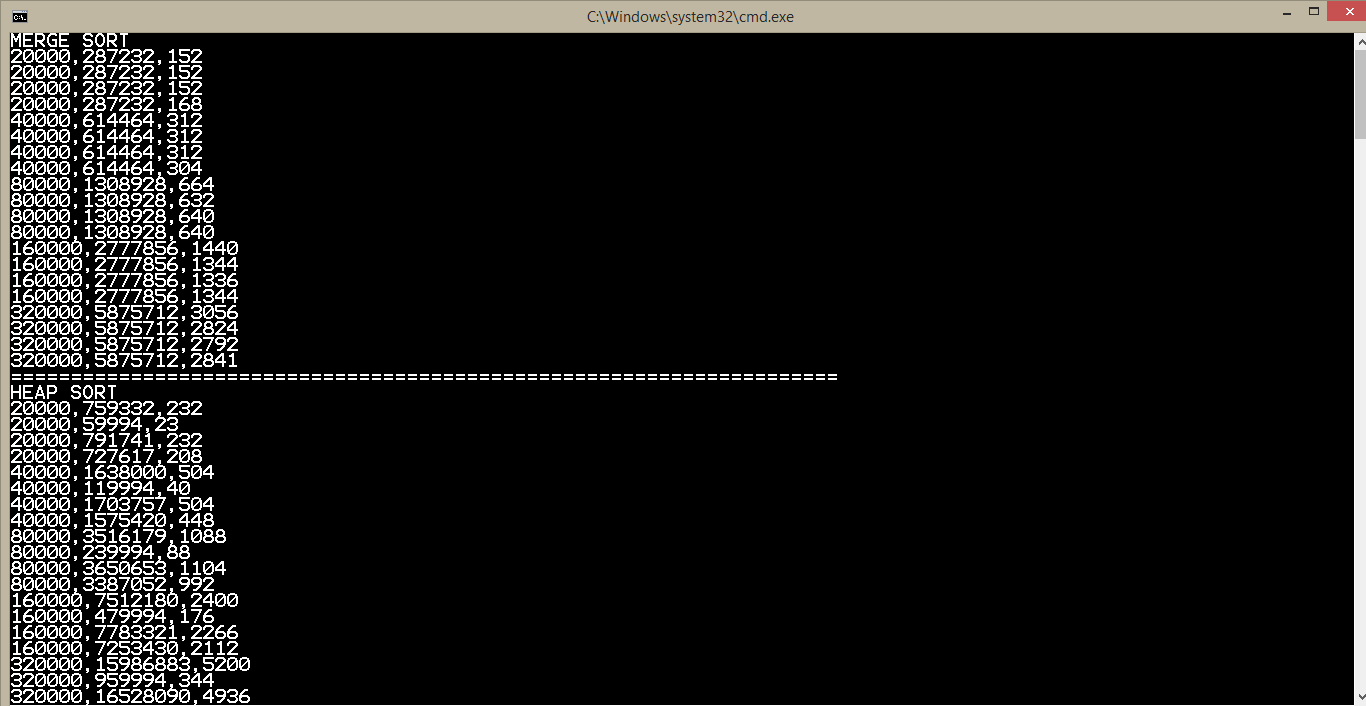
For Random Integers:

Time:

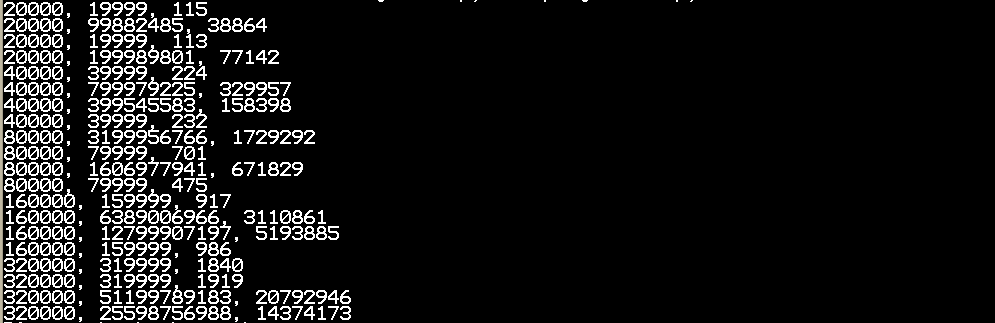
* Equal integers:
* Sorted integers:
* Reverse sorted integers:

For graphs regarding the comparisons , refer to Graphs excel workbook in the zip folder

Output screen:







**CONCLUSION:**

**As the number of elements increase, the time taken by any sorting algorithm increases. The time taken also depends on the computer on which the algorithm is running.**