Algorithm Analysis Project

For Algorithm Analysis class

Supervisor: Dr. Hala ElAarag

By:

Akram Aljarallah

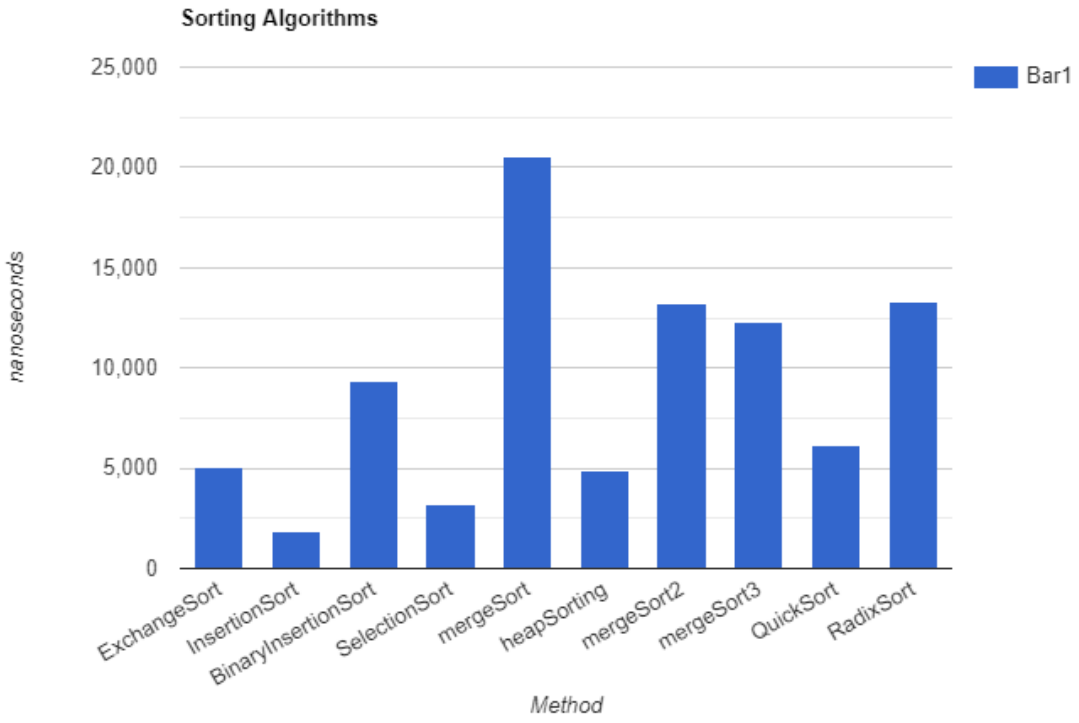
Saif Mostafa

Collected Data.

| | 10 | 100 | 1000 | 10,000 |
|---|---|---|---|---|
| ExchangeSort | 5073 | 168767 | 2844456 | 102811775 |
| InsertionSort | 1878 | 50668 | 341398 | 20138771 |
| BinaryInsertionSort | 9357 | 51467 | 414129 | 7559547 |
| SelectionSort | 3208 | 117611 | 779945 | 72912082 |
| mergeSort | 20575 | 88544 | 745019 | 74501923 |
| heapSorting | 4895 | 60651 | 156478 | 1464595 |
| mergeSort2 | 13248 | 84399 | 189905 | 1845262 |
| mergeSort3 | 12248 | 83399 | 179905 | 1745262 |
| QuickSort | 6113 | 67563 | 98361 | 859052 |
| RadixSort | 13288 | 70361 | 255768 | 1161397 |

Graphs

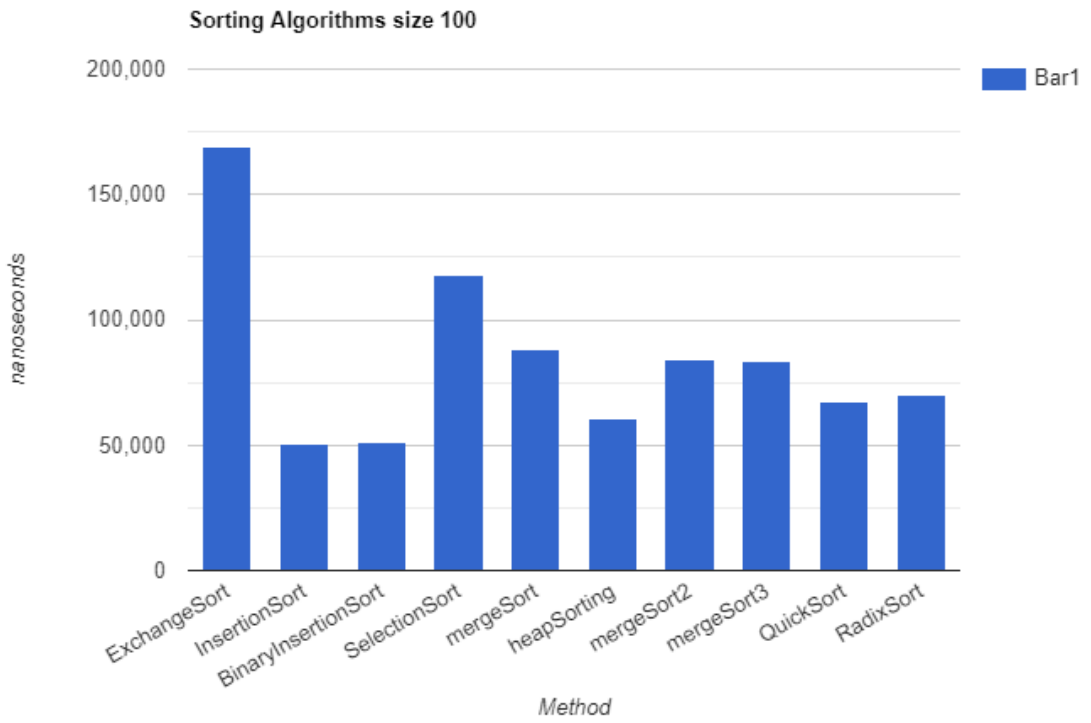
Size 10




Explanation:

- In arrays of small size such as 10, we can see merge sort with its different implementation has the highest runtime along with Radix sort.
- Insertion sort, Selection sort, and Heap sort are the most efficient sorting algorithms for array with small size.

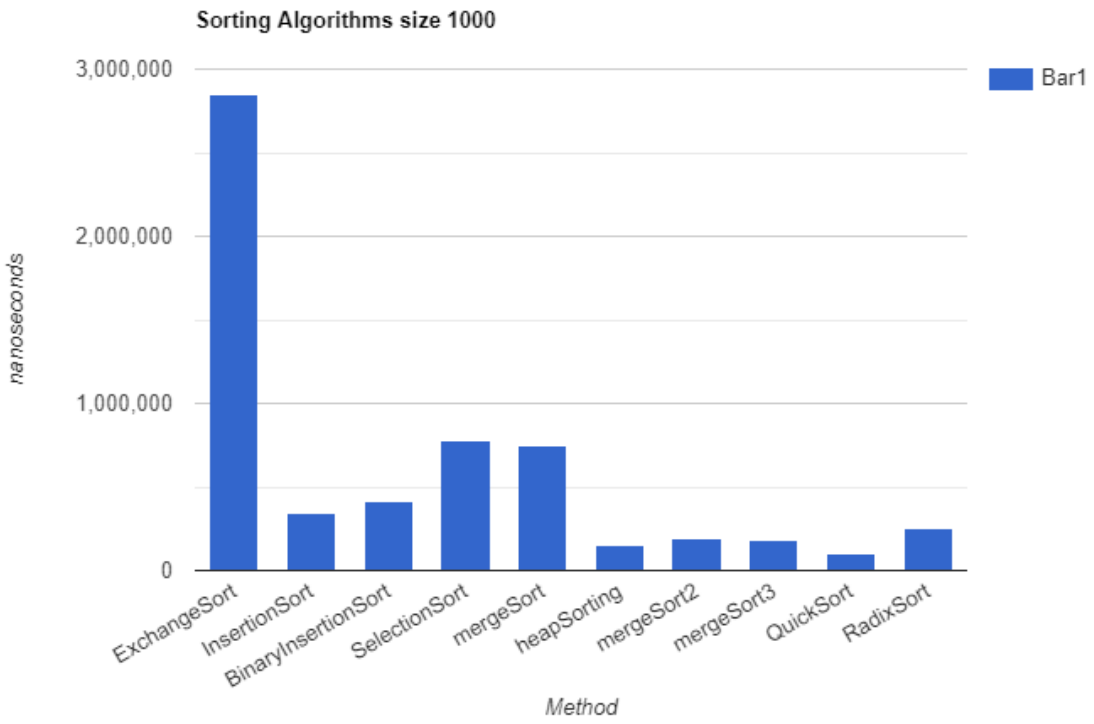
Size 100




Explanation:

- As we go bigger in size (100,) Merge sort becomes more efficient unlike Exchange sort which its runtime increased significantly.
- As we go bigger in size, Insertion sort and Heap sort are still very efficient algorithms. BinaryInsertion sort became very efficient.

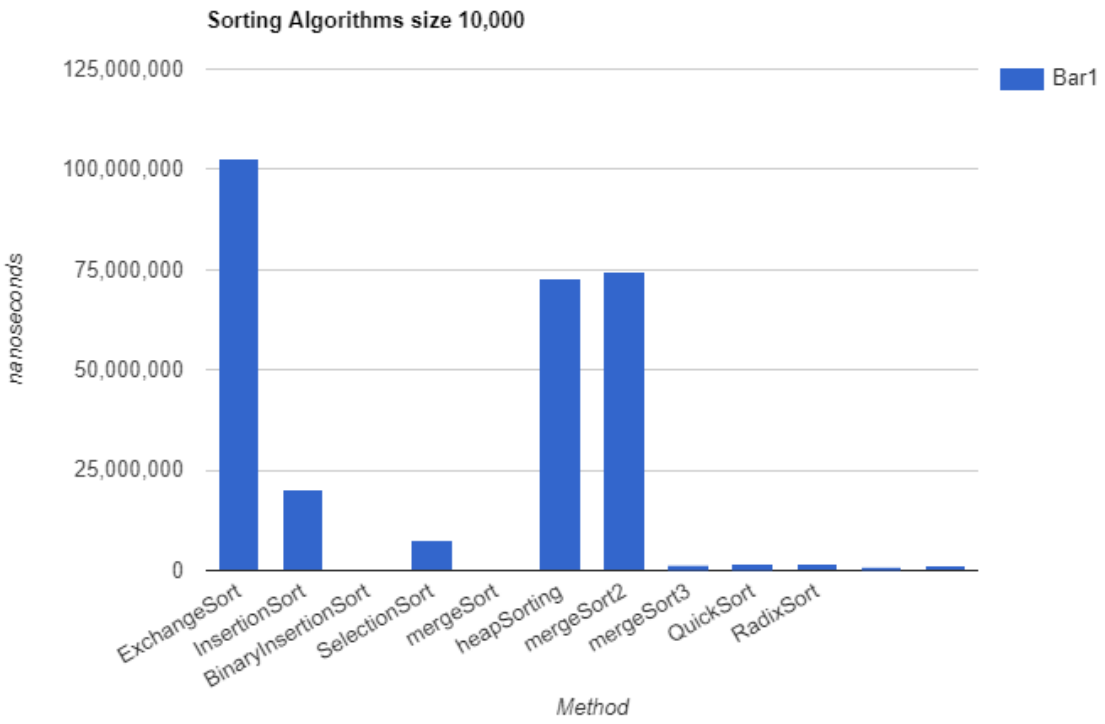
Size 1000




Explanation:

- Exchange sort is still the least efficient algorithm as we go bigger in size.
- Insertion sort became less efficient when we increased the size of the array to 1000 compared to other algorithms such as: Quick sort, Heap sort, and Radix sort.

Size 10000




Conclusion:

- Heap sort is very efficient no matter how big the array is.
- Exchange sort is the least efficient if we are working with an array size bigger than 15.
- BinaryInsertion sort became the Most efficient as we go higher in the size.