|  |  |  |  |
| --- | --- | --- | --- |
| **Git Size** | **Data Type** | **Algorithm** | **Time (ms)** |
| 1000 | Ascending | BubbleSort | 0.15 |
| 1000 | Ascending | InsertionSort | 0.26 |
| 1000 | Ascending | SelectionSort | 47.49 |
| 1000 | Ascending | MergeSort | 4.98 |
| 1000 | Ascending | QuickSort | 38.88 |
| 1000 | Ascending | QuickSortMedian3 | 1.99 |
| 1000 | Ascending | QuickSortRandom | 3.16 |
| 1000 | Descending | BubbleSort | 169.40 |
| 1000 | Descending | InsertionSort | 124.55 |
| 1000 | Descending | SelectionSort | 49.64 |
| 1000 | Descending | MergeSort | 4.81 |
| 1000 | Descending | QuickSort | 75.47 |
| 1000 | Descending | QuickSortMedian3 | 3.28 |
| 1000 | Descending | QuickSortRandom | 3.53 |
| 1000 | Random | BubbleSort | 124.34 |
| 1000 | Random | InsertionSort | 62.91 |
| 1000 | Random | SelectionSort | 43.19 |
| 1000 | Random | MergeSort | 4.94 |
| 1000 | Random | QuickSort | 2.06 |
| 1000 | Random | QuickSortMedian3 | 2.14 |
| 1000 | Random | QuickSortRandom | 2.98 |
| 1000 | NearlySorted | BubbleSort | 65.11 |
| 1000 | NearlySorted | InsertionSort | 8.08 |
| 1000 | NearlySorted | SelectionSort | 45.06 |
| 1000 | NearlySorted | MergeSort | 5.08 |
| 1000 | NearlySorted | QuickSort | 4.36 |
| 1000 | NearlySorted | QuickSortMedian3 | 2.39 |
| 1000 | NearlySorted | QuickSortRandom | 3.38 |
| 5000 | Ascending | BubbleSort | 0.72 |
| 5000 | Ascending | InsertionSort | 0.99 |
| 5000 | Ascending | SelectionSort | 1082.98 |
| 5000 | Ascending | MergeSort | 27.17 |
| 5000 | Ascending | QuickSort | 831.98 |
| 5000 | Ascending | QuickSortMedian3 | 12.07 |
| 5000 | Ascending | QuickSortRandom | 18.08 |
| 5000 | Descending | BubbleSort | 4242.13 |
| 5000 | Descending | InsertionSort | 3090.76 |
| 5000 | Descending | SelectionSort | 1133.16 |
| 5000 | Descending | MergeSort | 26.64 |
| 5000 | Descending | QuickSort | 1720.94 |
| 5000 | Descending | QuickSortMedian3 | 18.28 |
| 5000 | Descending | QuickSortRandom | 16.75 |
| 5000 | Random | BubbleSort | 2876.17 |
| 5000 | Random | InsertionSort | 1595.09 |
| 5000 | Random | SelectionSort | 1077.06 |
| 5000 | Random | MergeSort | 30.18 |
| 5000 | Random | QuickSort | 12.82 |
| 5000 | Random | QuickSortMedian3 | 13.45 |
| 5000 | Random | QuickSortRandom | 16.71 |
| 5000 | NearlySorted | BubbleSort | 1708.11 |
| 5000 | NearlySorted | InsertionSort | 193.06 |
| 5000 | NearlySorted | SelectionSort | 1072.66 |
| 5000 | NearlySorted | MergeSort | 29.42 |
| 5000 | NearlySorted | QuickSort | 37.92 |
| 5000 | NearlySorted | QuickSortMedian3 | 14.04 |
| 5000 | NearlySorted | QuickSortRandom | 17.45 |
| 10000 | Ascending | BubbleSort | 1.35 |
| 10000 | Ascending | InsertionSort | 2.05 |
| 10000 | Ascending | SelectionSort | 4324.87 |
| 10000 | Ascending | MergeSort | 59.80 |
| 10000 | Ascending | QuickSort | 3300.24 |
| 10000 | Ascending | QuickSortMedian3 | 24.10 |
| 10000 | Ascending | QuickSortRandom | 36.55 |
| 10000 | Descending | BubbleSort | 16979.73 |
| 10000 | Descending | InsertionSort | 12475.51 |
| 10000 | Descending | SelectionSort | 4523.25 |
| 10000 | Descending | MergeSort | 58.72 |
| 10000 | Descending | QuickSort | 7438.92 |
| 10000 | Descending | QuickSortMedian3 | 48.49 |
| 10000 | Descending | QuickSortRandom | 42.22 |
| 10000 | Random | BubbleSort | 12562.07 |
| 10000 | Random | InsertionSort | 6314.80 |
| 10000 | Random | SelectionSort | 4328.09 |
| 10000 | Random | MergeSort | 64.95 |
| 10000 | Random | QuickSort | 28.63 |
| 10000 | Random | QuickSortMedian3 | 27.59 |
| 10000 | Random | QuickSortRandom | 36.19 |
| 10000 | NearlySorted | BubbleSort | 7059.72 |
| 10000 | NearlySorted | InsertionSort | 818.66 |
| 10000 | NearlySorted | SelectionSort | 4338.27 |
| 10000 | NearlySorted | MergeSort | 63.49 |
| 10000 | NearlySorted | QuickSort | 84.80 |
| 10000 | NearlySorted | QuickSortMedian3 | 30.49 |
| 10000 | NearlySorted | QuickSortRandom | 37.95 |
| 20000 | Ascending | BubbleSort | 2.67 |
| 20000 | Ascending | InsertionSort | 4.05 |
| 20000 | Ascending | SelectionSort | 17493.26 |
| 20000 | Ascending | MergeSort | 125.52 |
| 20000 | Ascending | QuickSort | 13699.94 |
| 20000 | Ascending | QuickSortMedian3 | 53.55 |
| 20000 | Ascending | QuickSortRandom | 84.33 |
| 20000 | Descending | BubbleSort | 68736.02 |
| 20000 | Descending | InsertionSort | 49949.12 |
| 20000 | Descending | SelectionSort | 17992.31 |
| 20000 | Descending | MergeSort | 126.61 |
| 20000 | Descending | QuickSort | 27419.10 |
| 20000 | Descending | QuickSortMedian3 | 89.69 |
| 20000 | Descending | QuickSortRandom | 75.95 |
| 20000 | Random | BubbleSort | 45705.16 |
| 20000 | Random | InsertionSort | 24800.06 |
| 20000 | Random | SelectionSort | 17580.83 |
| 20000 | Random | MergeSort | 134.55 |
| 20000 | Random | QuickSort | 63.30 |
| 20000 | Random | QuickSortMedian3 | 58.75 |
| 20000 | Random | QuickSortRandom | 76.84 |
| 20000 | NearlySorted | BubbleSort | 27368.77 |
| 20000 | NearlySorted | InsertionSort | 3103.93 |
| 20000 | NearlySorted | SelectionSort | 17045.90 |
| 20000 | NearlySorted | MergeSort | 146.77 |
| 20000 | NearlySorted | QuickSort | 174.55 |
| 20000 | NearlySorted | QuickSortMedian3 | 67.96 |
| 20000 | NearlySorted | QuickSortRandom | 81.61 |

The runtime results demonstrate the performance characteristics of various sorting algorithms, aligning with their theoretical time complexities.

* **BubbleSort, InsertionSort and SelectionSort**, all O(N²), show significant performance degradation as array size increases.
  + ***BubbleSort*** is the slowest on large descending arrays (68,736.02 ms for N=20,000), due to excessive swaps.
  + ***InsertionSort*** performs best among O(N²) algorithms on nearly sorted data (8.08 ms for N=1,000), leveraging its O(N) best-case efficiency, but struggles with descending arrays (49,949.12 ms for N=20,000).
  + ***SelectionSort*** maintains consistent performance (around 17,000–18,000 ms for N=20,000) but does not benefit from partially sorted data, making it less practical.
* **MergeSort** consistently achieves O(N log N) complexity, with stable runtimes across all data types (125.52–146.77 ms for N=20,000), though its non-in-place nature requires additional memory.
* **QuickSort** with left-most pivot suffers worst-case O(N²) behavior on ascending and descending arrays (13,699.94 ms and 27,419.10 ms for N=20,000), due to unbalanced partitions, posing a stack overflow risk for larger arrays.
  + ***QuickSortMedian3*** and **QuickSortRandom** mitigate this, achieving near O(N log N) performance (58.75 ms and 76.84 ms for N=20,000 random), with **QuickSortMedian3** being the fastest due to its robust pivot selection.
  + On random and nearly sorted data, *QuickSort variants outperform MergeSort* slightly, benefiting from in-place sorting.
  + These results confirm the theoretical advantages of **O(NlogN)** algorithms, with QuickSortMedian3 being the most efficient for general-purpose sorting, while MergeSort is preferable for stability and predictability.