Program Structures and Alogorithms

Spring 2023

Assignment 6

Name: Srikanth Nandikonda

NUID: 002737724

Task:

Task is to determine--for sorting algorithms--what is the best predictor of total execution time: comparisons, swaps/copies, hits (array accesses), or something else.

Evidence:

Stats for Merge sort algo

Graphical user interface, application

Description automatically generated

Table

Description automatically generated

Stats for Heapsort

Graphical user interface

Description automatically generated

Stats for quicksort with dual pivot

Table

Description automatically generated

Number of elements vs Time

Among the three algorithms, Quicksort appears to be the fastest for all array sizes. It is followed by Merge sort, which takes slightly longer, and Heapsort, which is the slowest.

Hits comparison

Table

Description automatically generated

The parameters under consideration in sorting algorithms include compares, copies, and swaps, all of which involve accessing the array. Therefore, each time these operations are performed, there is an associated "hit" to the array. This suggests that the number of hits could be a useful metric for evaluating the efficiency of different algorithms. Generally, algorithms with more hits would be expected to perform worse than those with fewer hits, if all other operations performed by the algorithms take the same amount of time. In such cases, the number of hits could be a reliable predictor of the actual execution time of the algorithms. We can also see this from the above two graphs, heap sort has the highest number of hits so it took highest time to perform the sorting ,which is followed by merge sort and quick sort which has least hits and in turn least amount of time to perform sorting.

Number of Compares for various algo.

Table

Description automatically generated

Table

Description automatically generated

In the scenario where sorting is performed on the data structure were comparison will take more time when compared to other operations, then Comparison is the better metric for evaluating the efficiency of different algorithms. In the current case, where sorting is performed on the string array, Comparison operations for strings typically involve comparing each character in two strings until a difference is found. The number of comparisons required will depend on the length of the strings and the specific strings being compared. With these assumptions, if the comparisons take more time , then algorithm with higher comparisons will be the worst performing algorithm. For the data we can see heap sort has the highest compares and thus performance is worst for this algorithm. Followed by heap sort, Quick sort takes next high number of compares to perform sorting, but it takes less amount of time than merge sort. The difference between the time taken to perform sorting between quick sort and merge sort is very less in the case of smaller arrays and this increases as the length increases. This may be because of the higher number of copy operations performed in the case of merge sort, and zero copy operations in the Quicksort.

Testcase report:

HeapSortTest.java

Text

Description automatically generated

Text

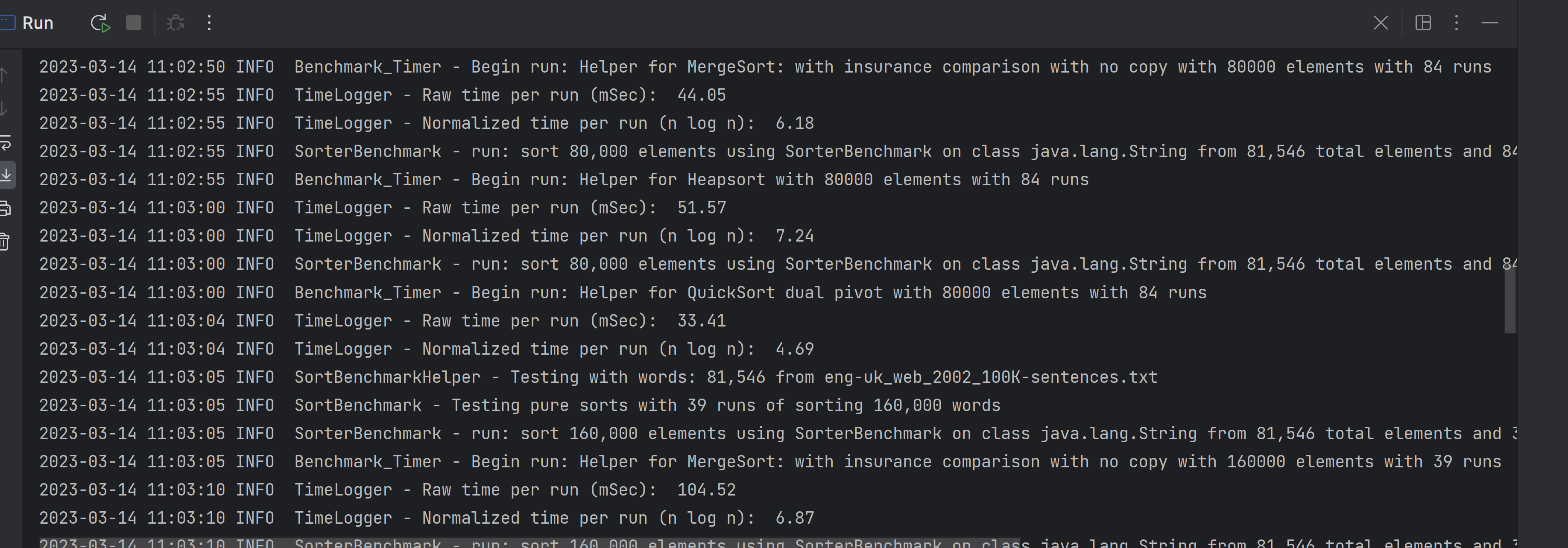
Description automatically generated**QuickSortTest**

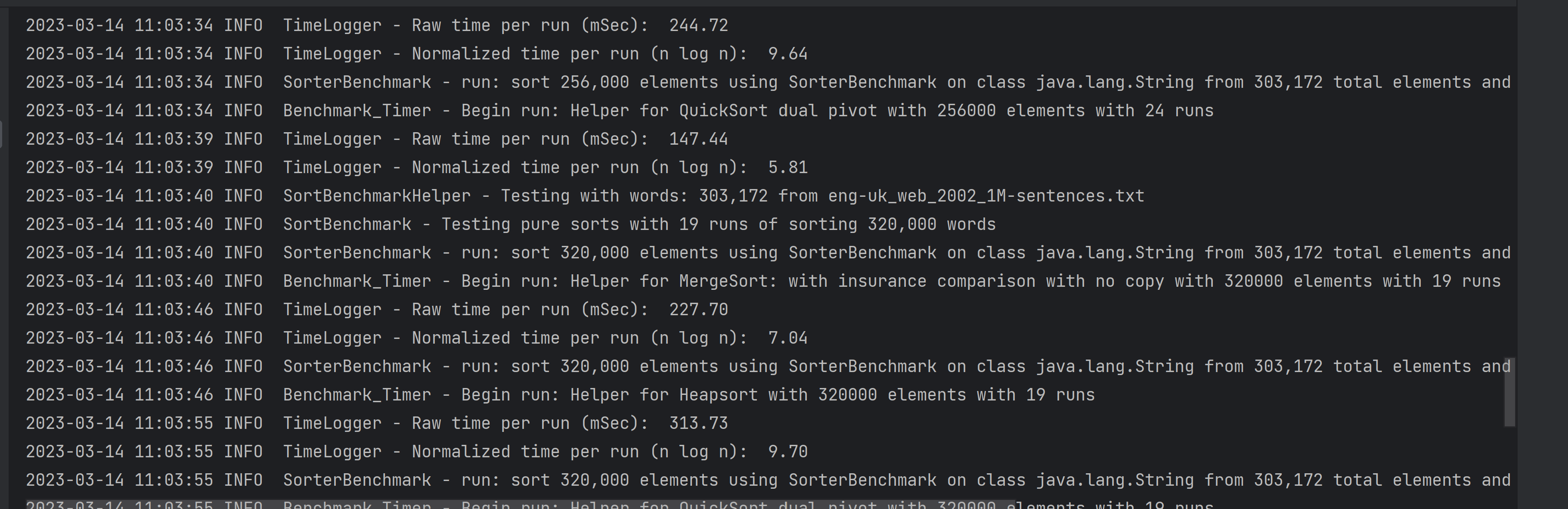
Text

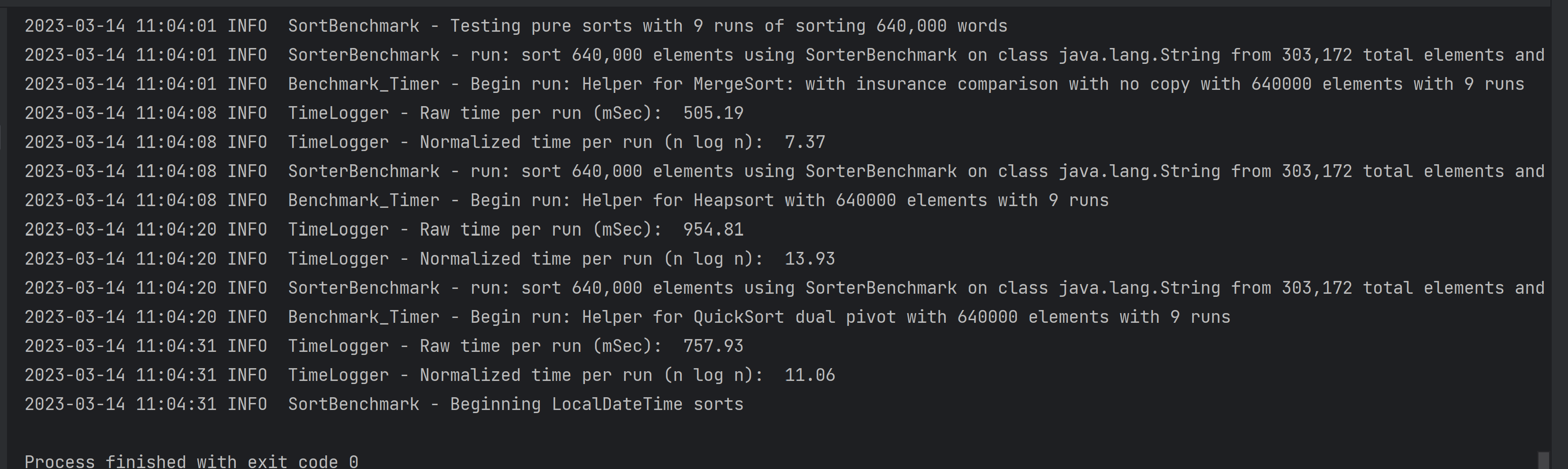
Description automatically generated**MergeSortTest**

Output:

Sorting without instrumentation







Merge sort with instrumentation

Text

Description automatically generated

A screen shot of a computer

Description automatically generated with low confidence

Graphical user interface

Description automatically generated with low confidence

Heapsort with instrumentation

A picture containing text

Description automatically generated

Text

Description automatically generated

Quick sort with dual pivot no instrumentation

A picture containing text

Description automatically generated

Text

Description automatically generated

Quick sort with dual pivot with instrumentation

A computer screen capture

Description automatically generated with low confidence

A picture containing text, outdoor

Description automatically generated

A picture containing text, outdoor, screenshot

Description automatically generated