**AIM:** Implementation of Quick Sort and Merge Sort (both Iterative and Recursive) and also computing the memory utilization of it.

**EXPERIMENT:** Implement merge sort and quick sort using iterative and recursive methods. The

number of inputs elements has to be passed from command line arguments. The

elements have to be generated randomly within the code. Compute:

a. Check the performance of program by varying the number of elements.

b. Compute the time taken by each case (for particular number of inputs).

c. Plot a graph with number of inputs to time taken in seconds.

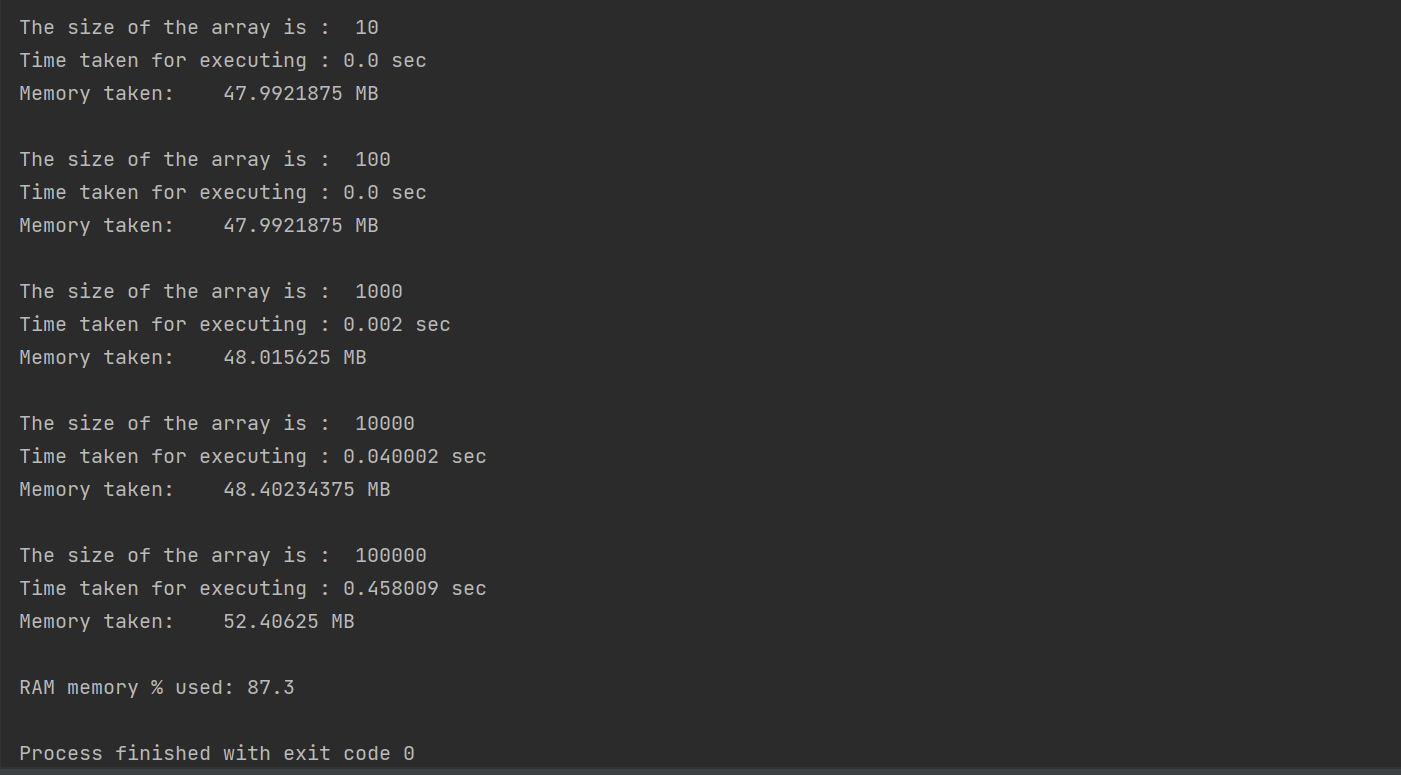
d. Compute and compare the memory taken by recursive and iteration implementation of the two sorting algorithms.

**MERGE SORT**

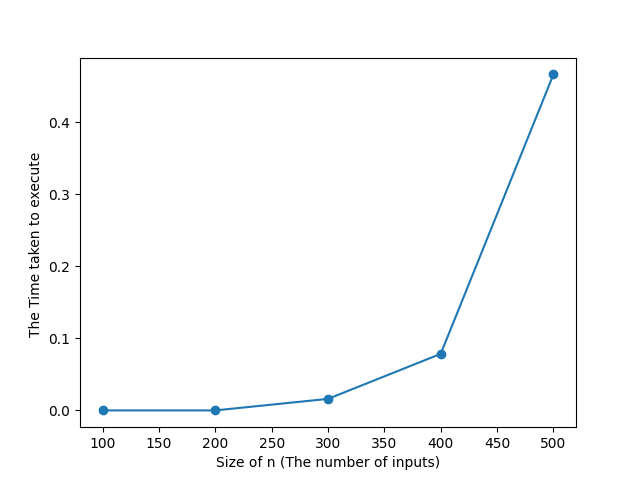
Merge Sort divides the input array into two halves, calls itself for the two halves, and then merges the two sorted halves.

**(ITERATIVE)**

**SHOWING TIME OF EXECUTION AND MEMORY USED WITH RESPECT TO THE SIZE OF INPUT**

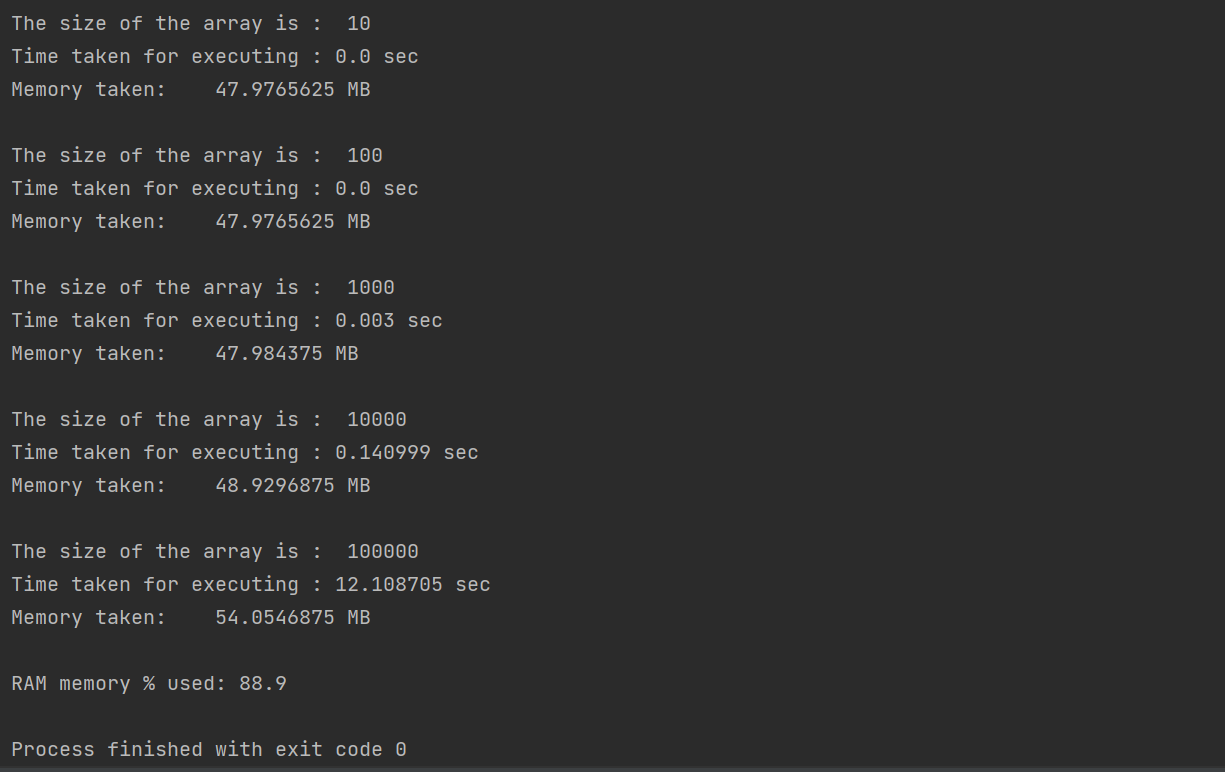


**(GRAPH)**

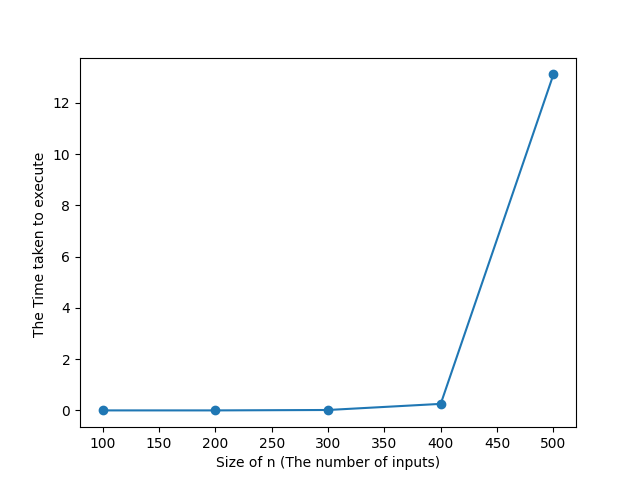


**(RECURSIVE)**

**SHOWING TIME OF EXECUTION AND MEMORY USED WITH RESPECT TO THE SIZE OF INPUT**



**(GRAPH)**

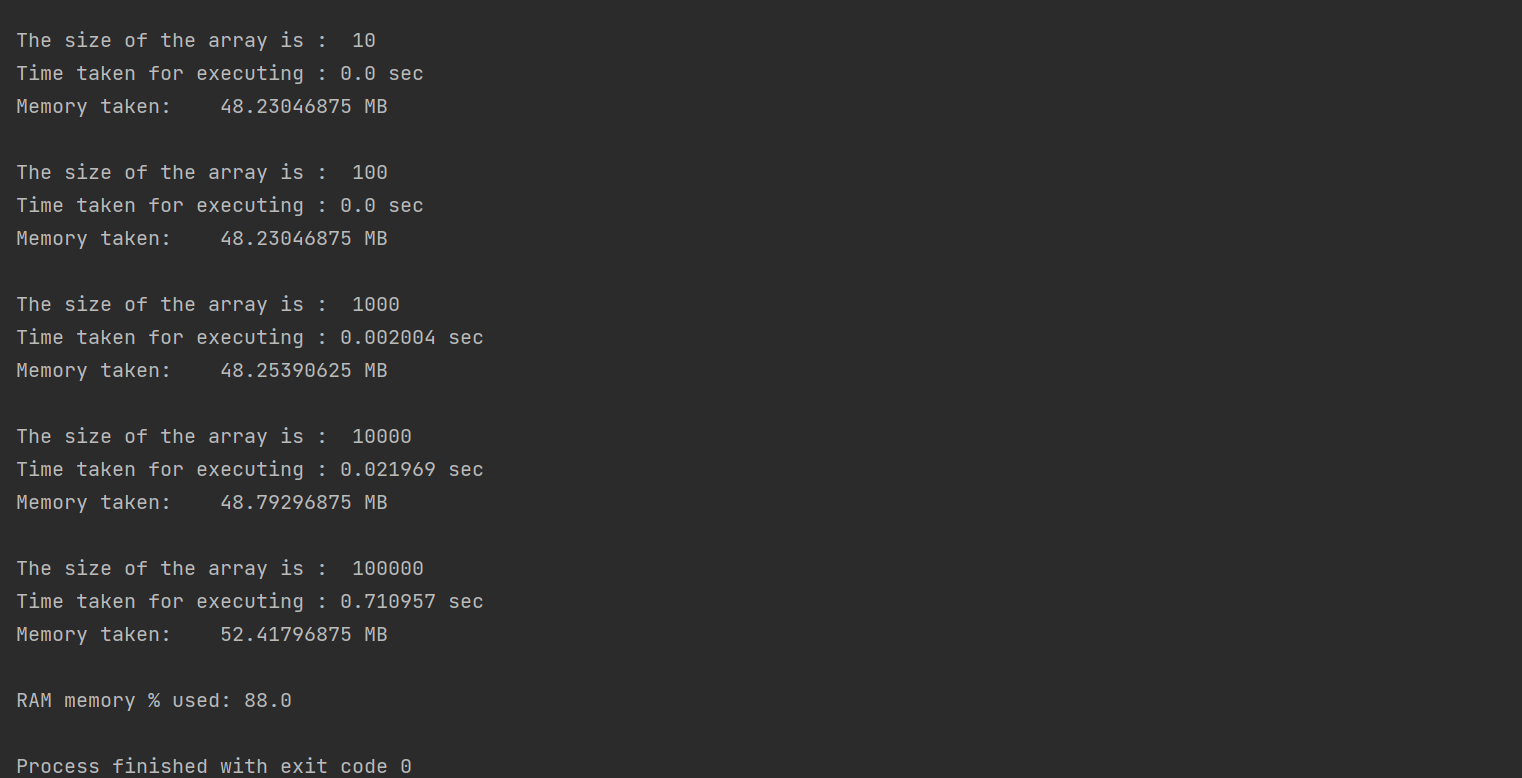


**QUICK SORT**

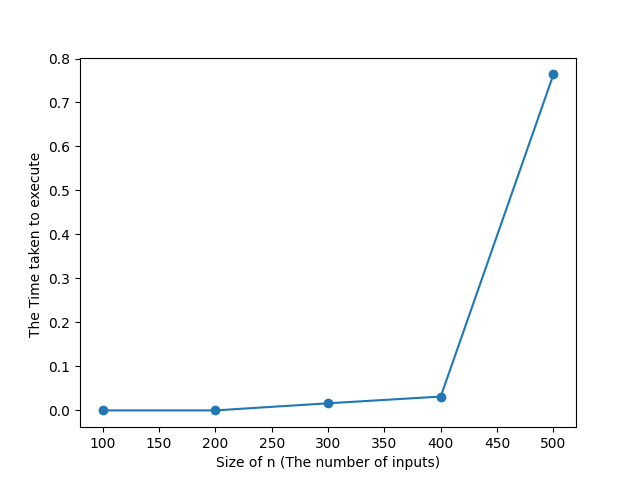
QuickSort is a Divide and Conquer algorithm. It picks an element as pivot and partitions the given array around the picked pivot.

**ITERATIVE**

**SHOWING TIME OF EXECUTION AND MEMORY USED WITH RESPECT TO THE SIZE OF INPUT**

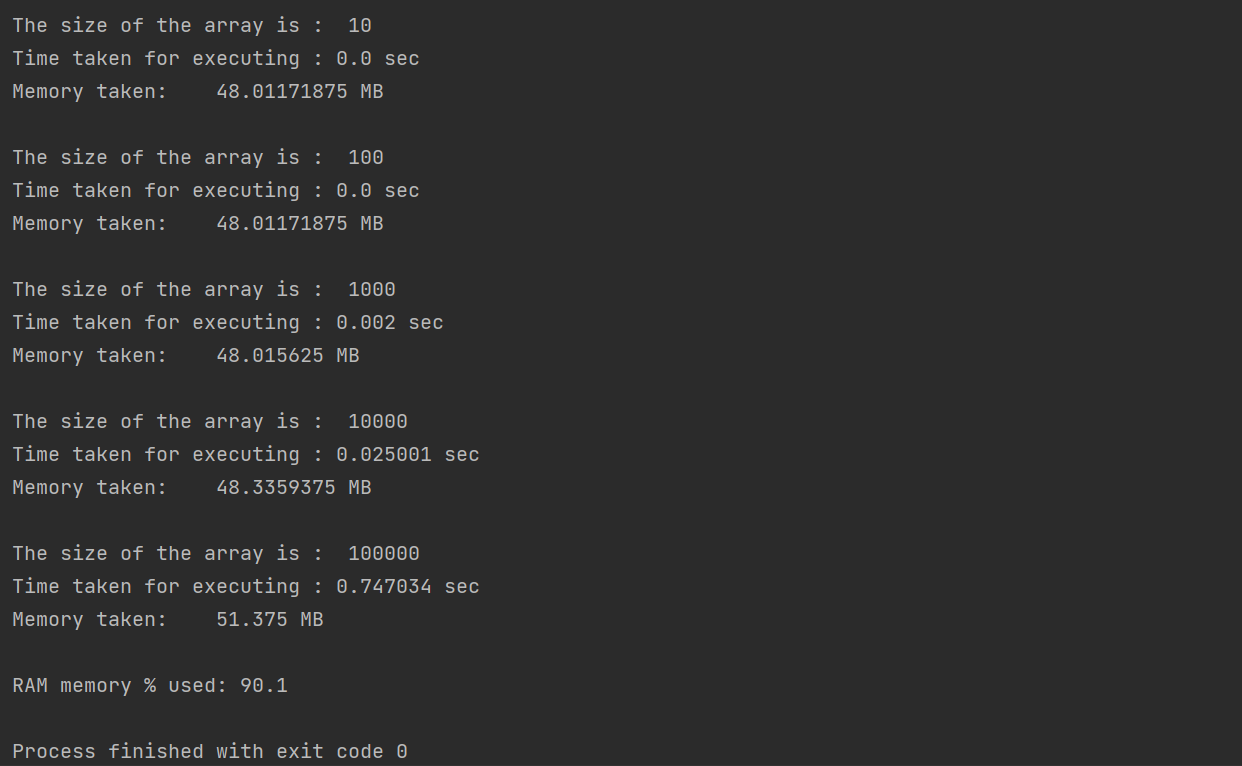


**(GRAPH)**

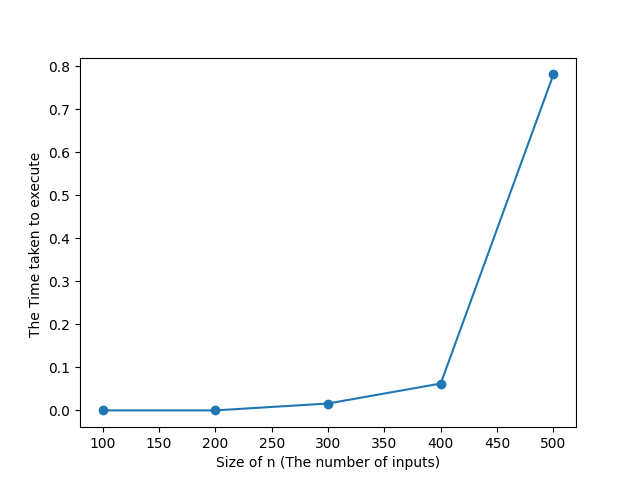


**RECURSIVE**

**SHOWING TIME OF EXECUTION AND MEMORY USED WITH RESPECT TO THE SIZE OF INPUT**



**(GRAPH)**



Conclusion :- Iterative Sorting algorithms are a bit faster in comparison to Recursive one.

Iterative Sorting algorithm also take less Auxiliary Space in comparison to Recursive one.