# ITCS 6114/8114 – Algorithm and Data Structures

## Project Part - 1

*- Submitted by: TARUN CHUNCHU*
*(800888999)*

## 1.Merge Sort:

Data structures chosen: Array of N elements which are randomly generated using random function.

Methods involved:

- Merge (left, right)
- Merge_sort (A)

Complexity Analysis:

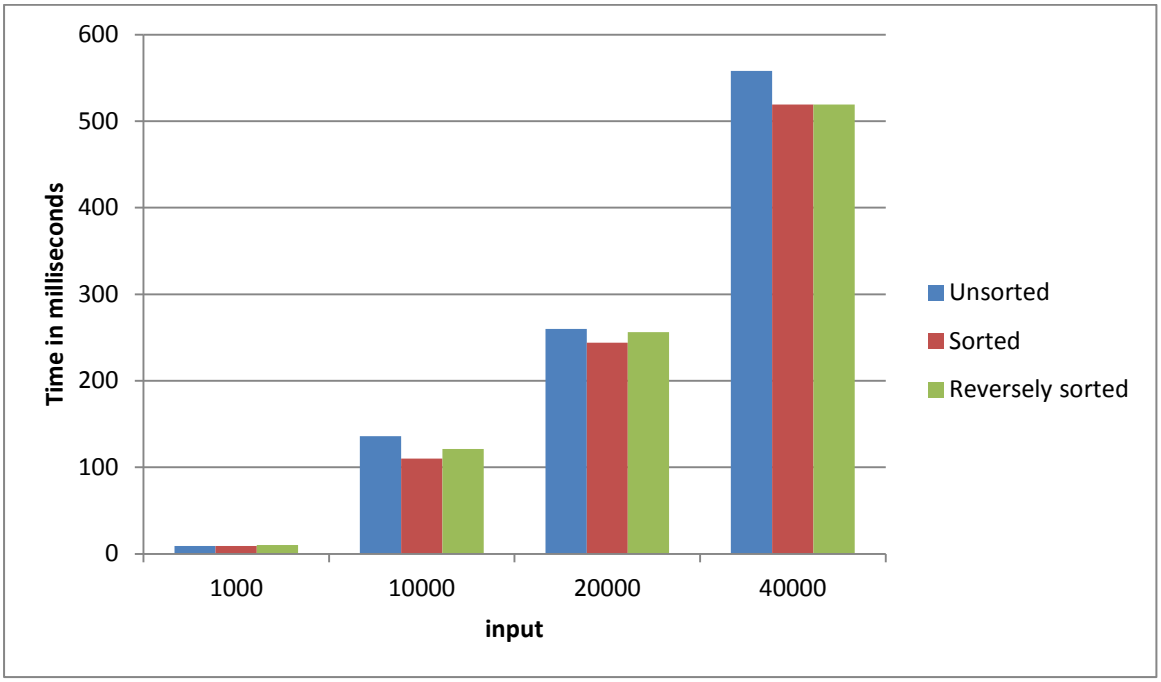
Merge sort: This method runs in O (nlogn) time. It is used to sort an array in place.

Best case: O(nlog(n))

Averagecase:O(nlog(n))

Worst case: O(nlog(n))

Graph for merge sort showing execution time for 3 cases:



## 2.Heap Sort:

Data structures chosen: Array of N elements which are randomly generated using random function.

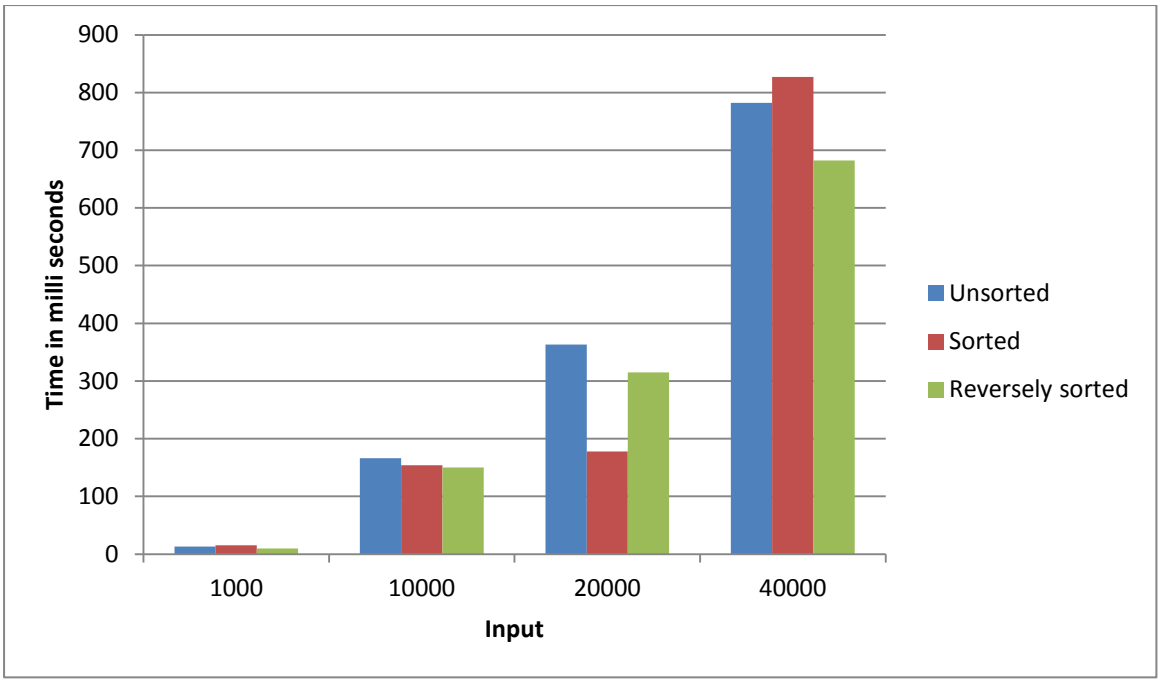
Complexity analysis:

Best case: O(nlog(n))

Average case: O(nlog(n))

Worst case: O(nlog(n))

Graph showing execution times of heap sort in 3 cases:



## 3. In – Place Quick Sort:

Data structures chosen: Array of N elements which are randomly generated using random function. Here Pivot is chosen randomly.

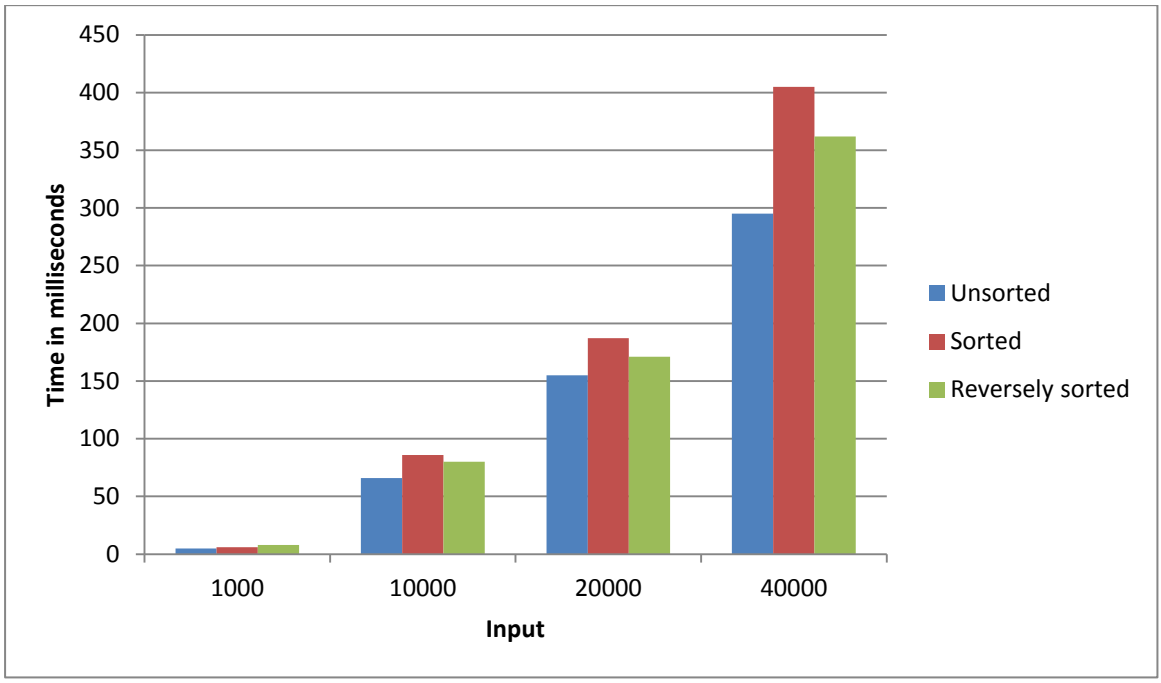
Complexity analysis:

Best case: O(nlog(n))

Average case: O(nlog(n))

Worst case: O(n^2)

Graph showing execution time of Quick sort algorithm in 3 cases:




## 4. In – Place Quick Sort – Median of Three and Insertion sort:

Data structures chosen: Array of N elements which are randomly generated using random function.

In this method we take first , middle and last element of the array and we compare among them then we move median to the center and then we move it side of first position.

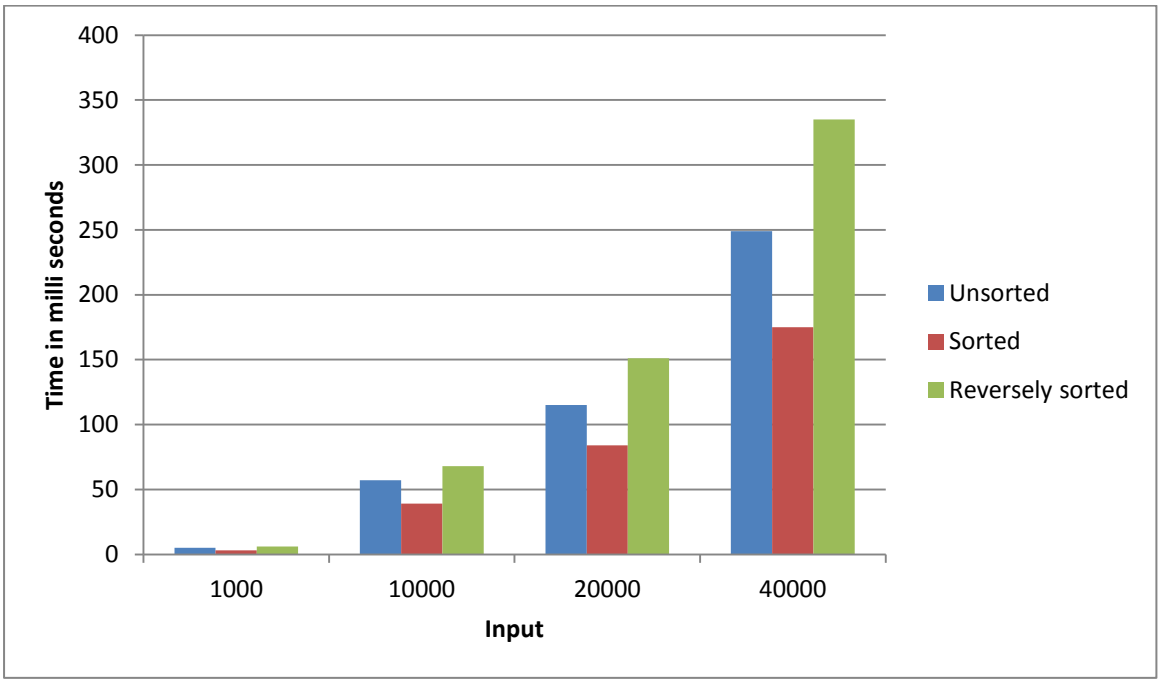
Complexity analysis:

Best case: O(nlog(n))

Average case: O(nlog(n))

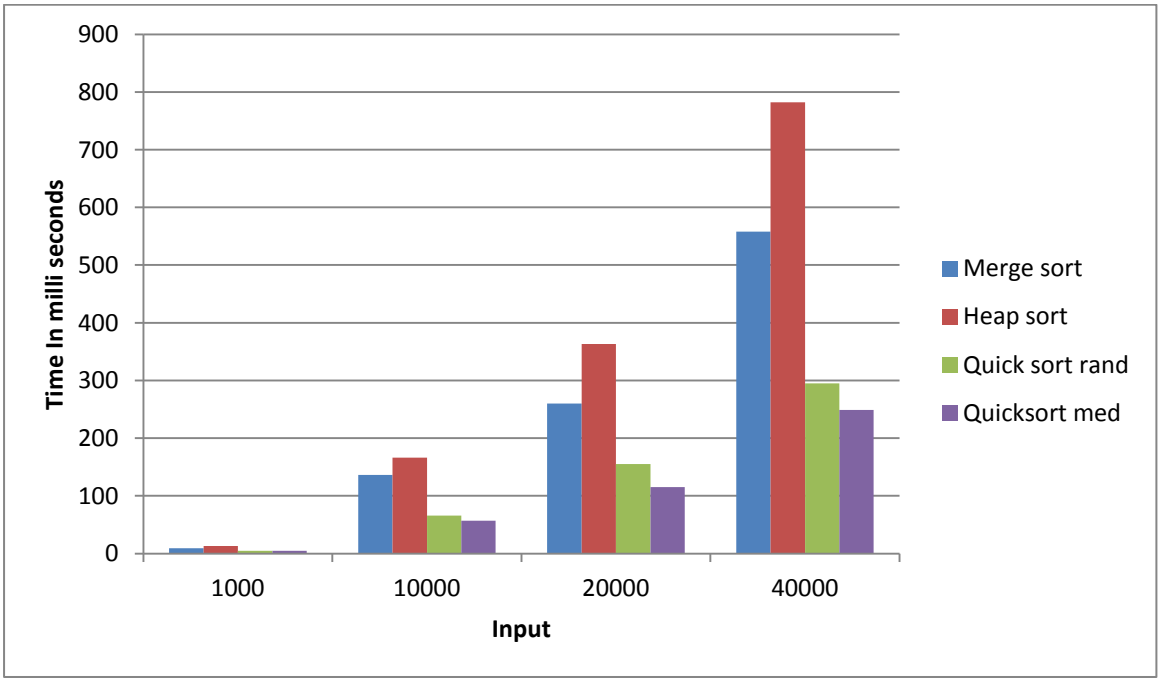
Worst case: O(n^2)

Graph showing execution time of Quick sort- Median as pivot in 3 cases:

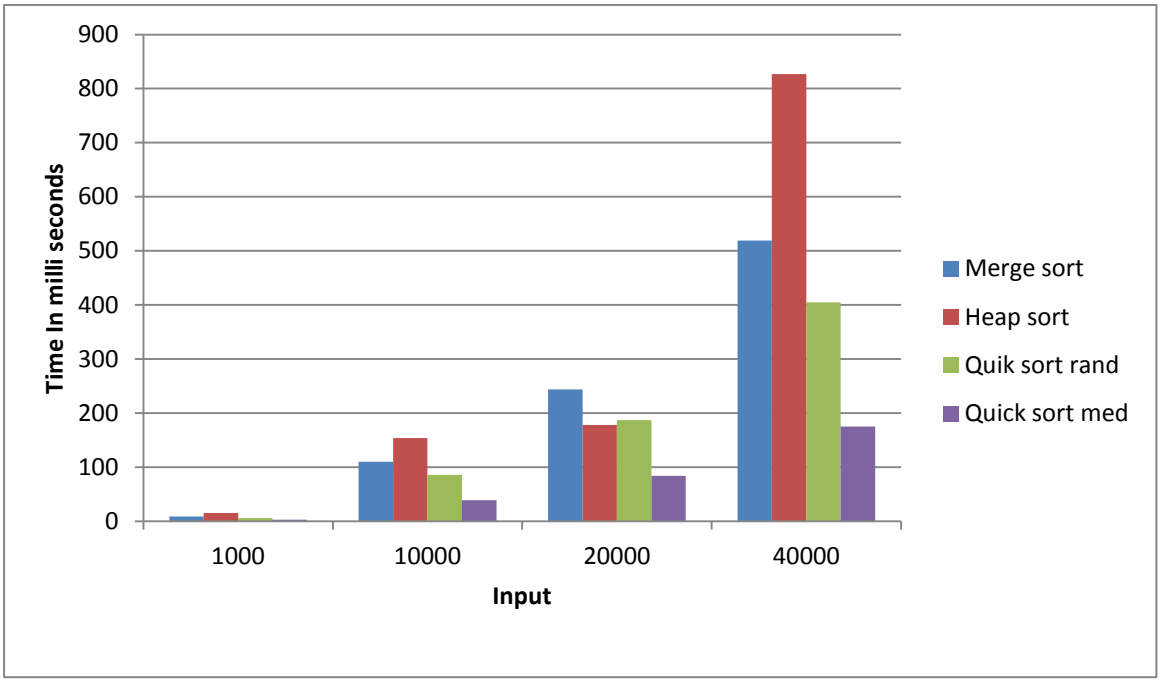



Here are graphs which compares algorithms in three different scenarios such as unsorted, sorted and reversely sorted.

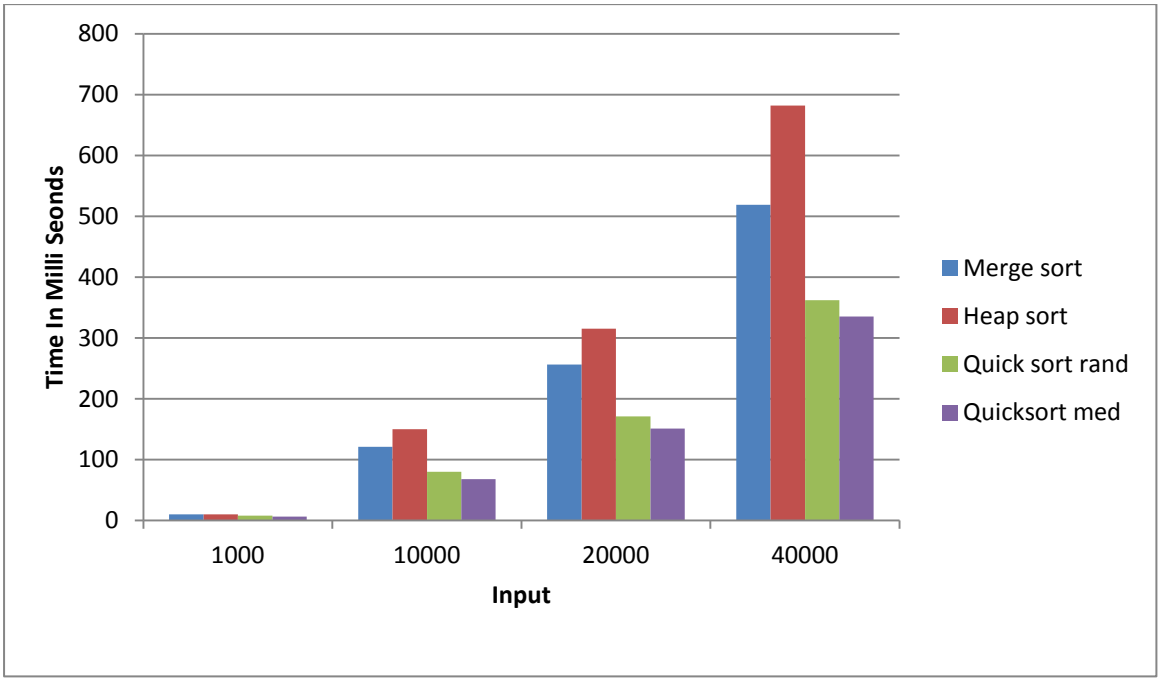
Graph showing comparison of algorithms for unsorted input :



Graph showing execution time of sorted input:



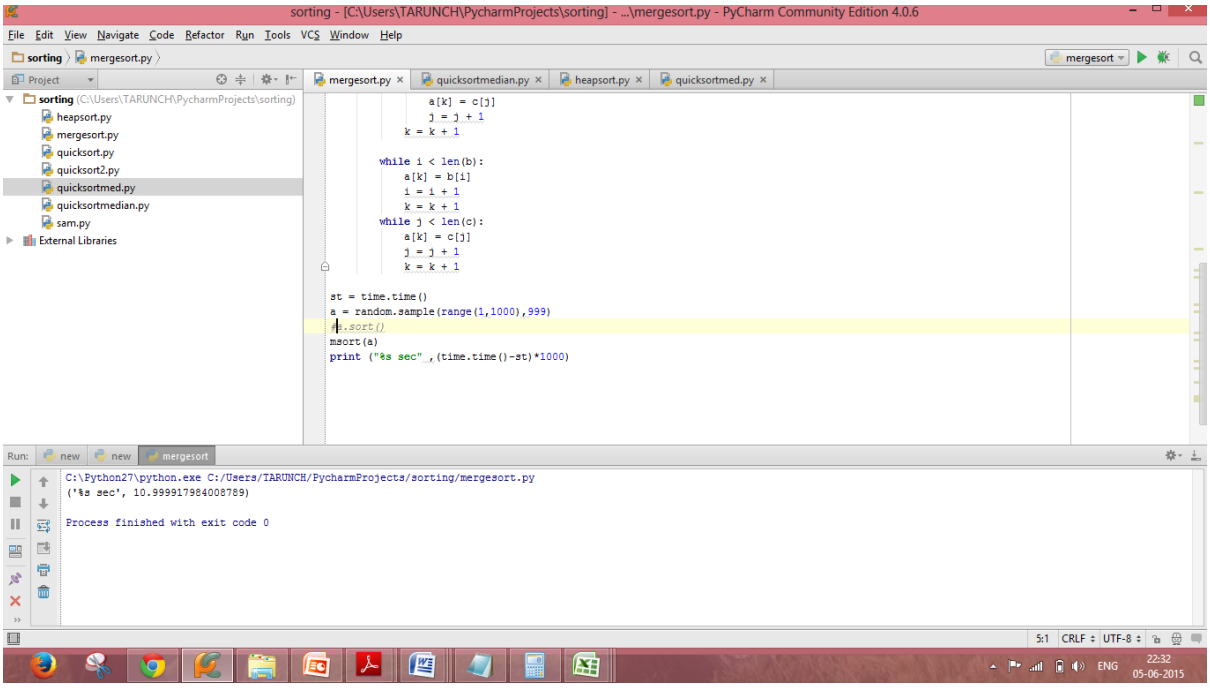

Graph showing execution time for reversely sorted input:



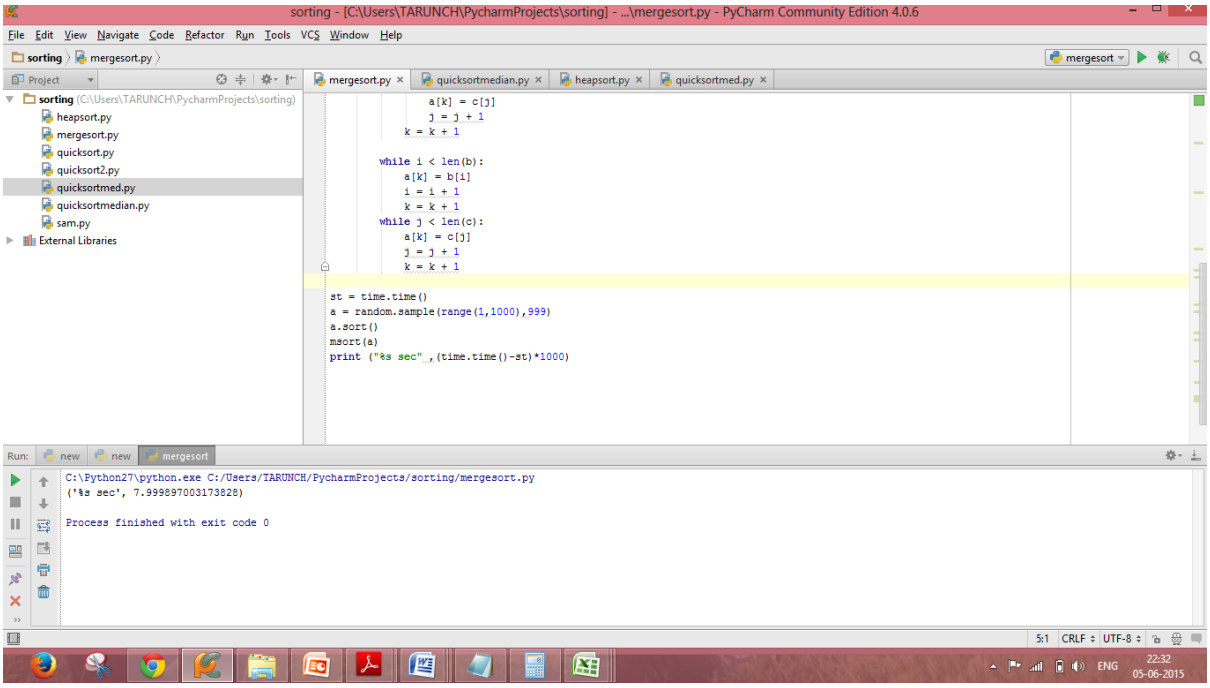
Analysis:

- Upon observing the entire cases heap sorting algorithm consuming more time than merge sort as it takes time to heapify. Big oh notation for both the algorithms is same nlog(n) but while coming to implementation heap sort it taking more time than merge algorithm.
- Quick sort with pivot as median executed faster than quick sort in which pivot chosen randomly.
- In inplace quick sort(pivot chosen randomly) sorted array takes more time than reversely sorted array.
- Execution time of heap sort increases very much corresponding to increase of input size.

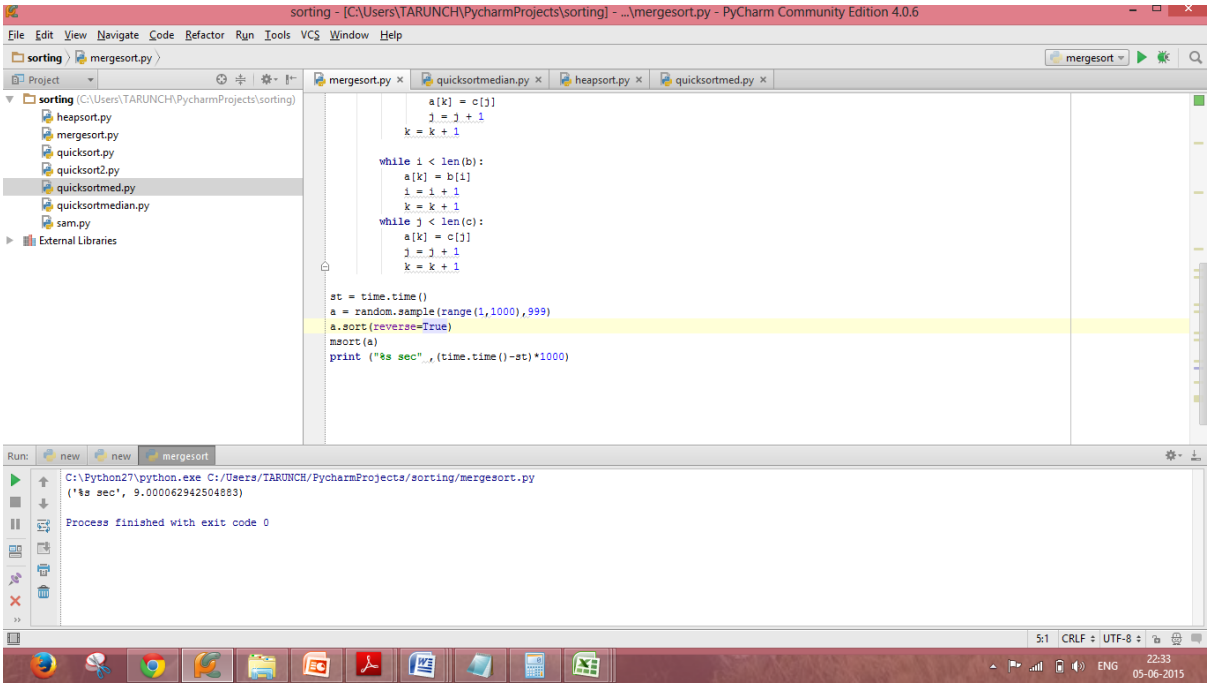
Merge sort execution: (Unsorted case)



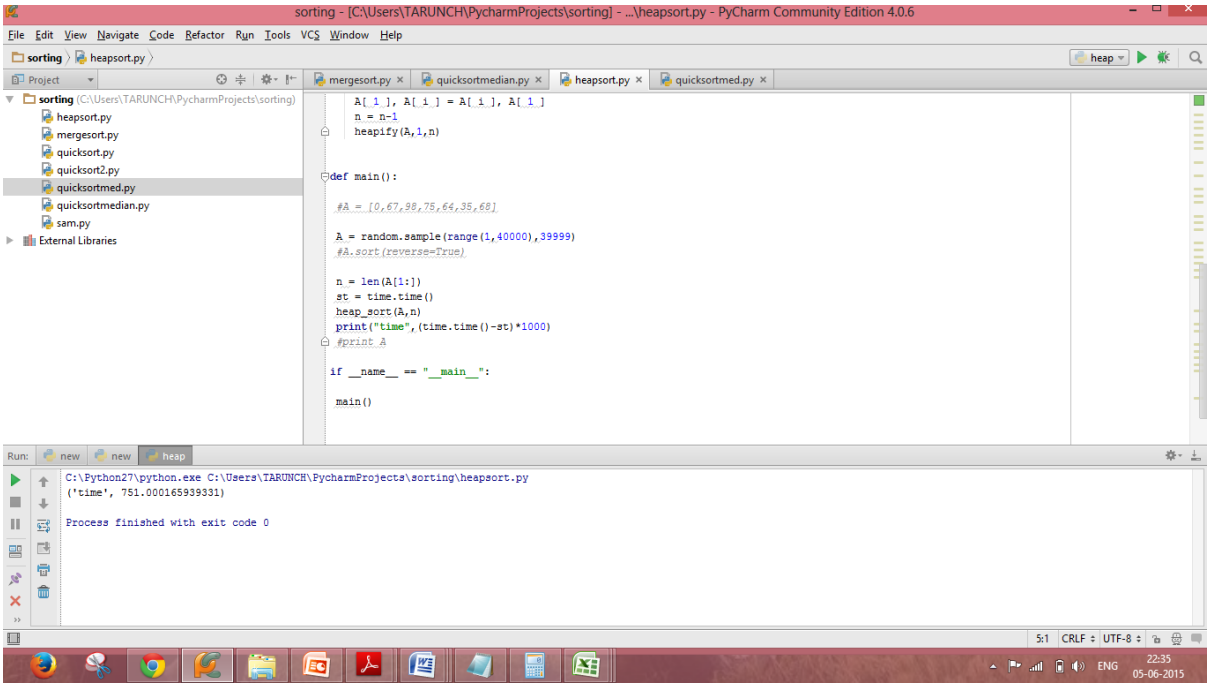

Merge sort execution: (Sorted case)



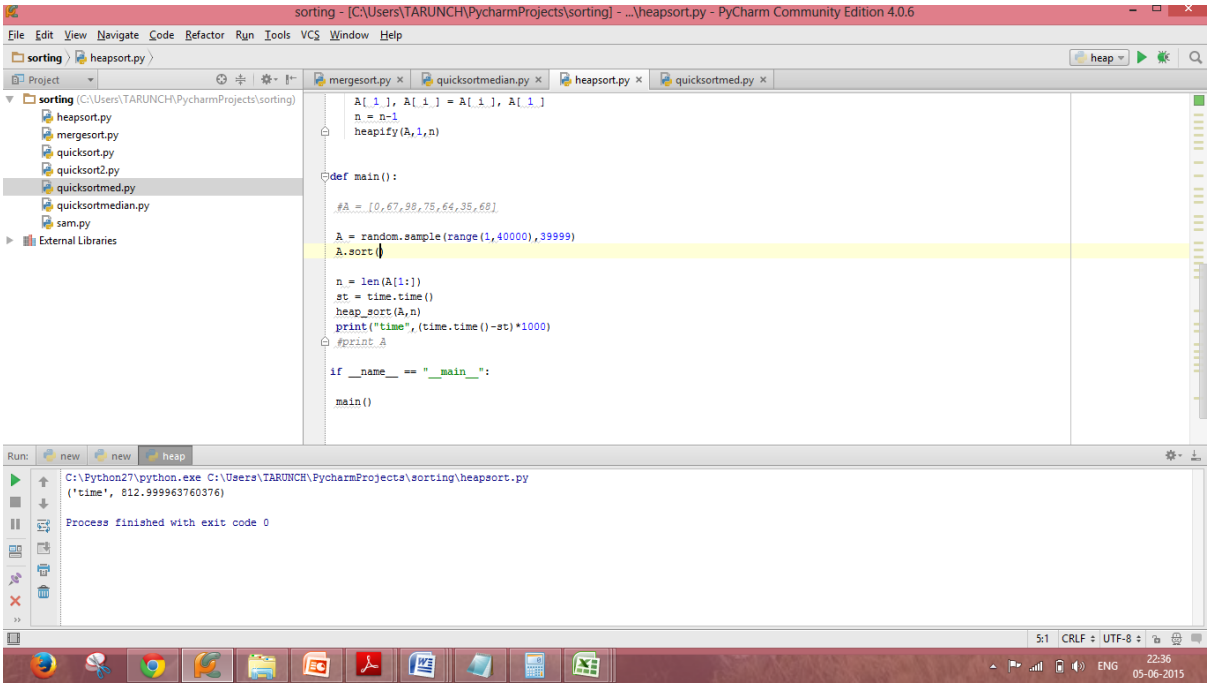
Merge sort: (Reversely sorted)



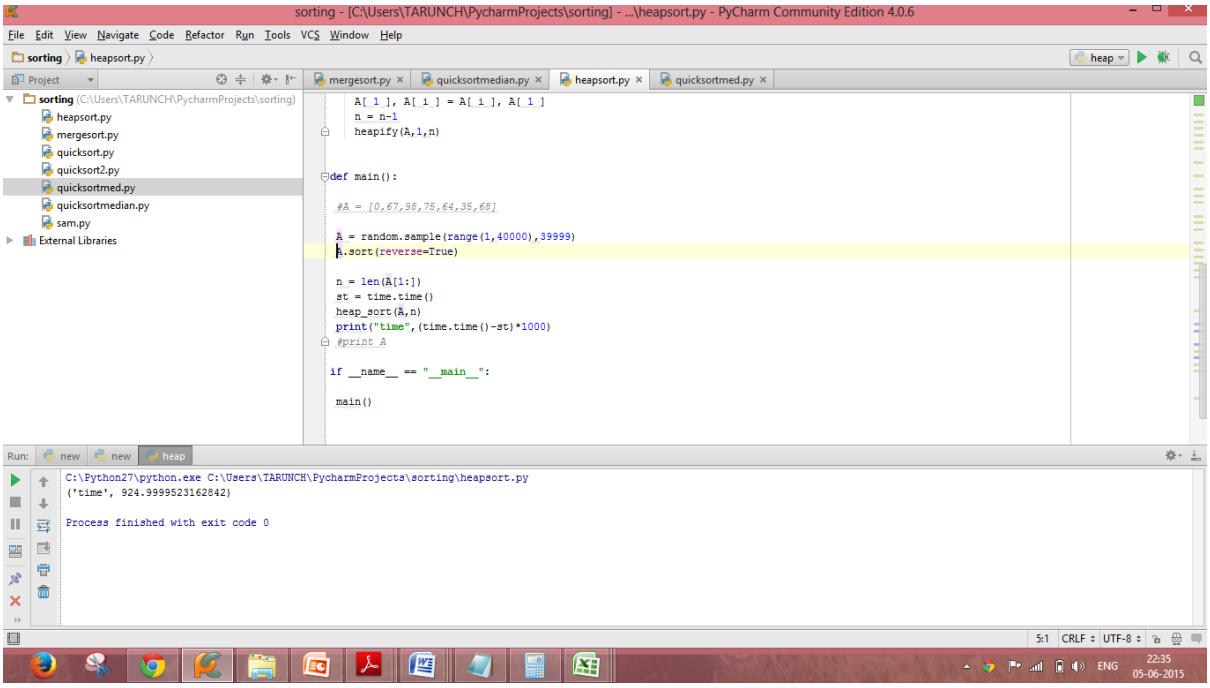

Heap sort: (Unsorted)



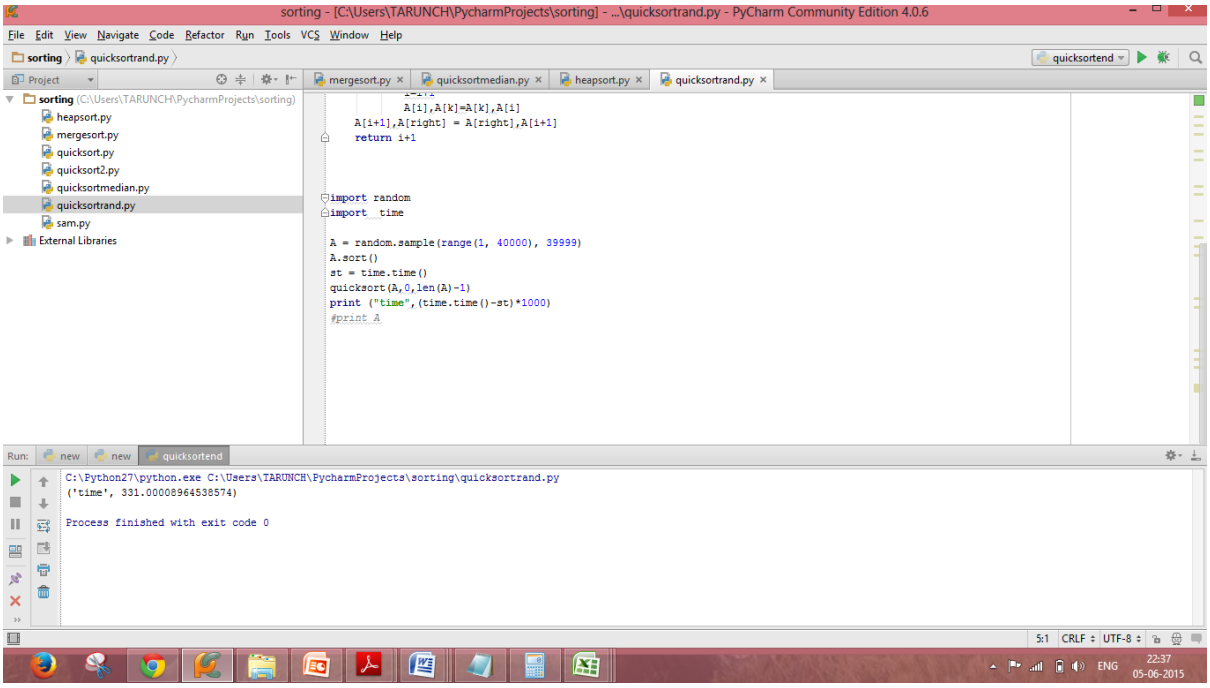
Heap sort(sorted):



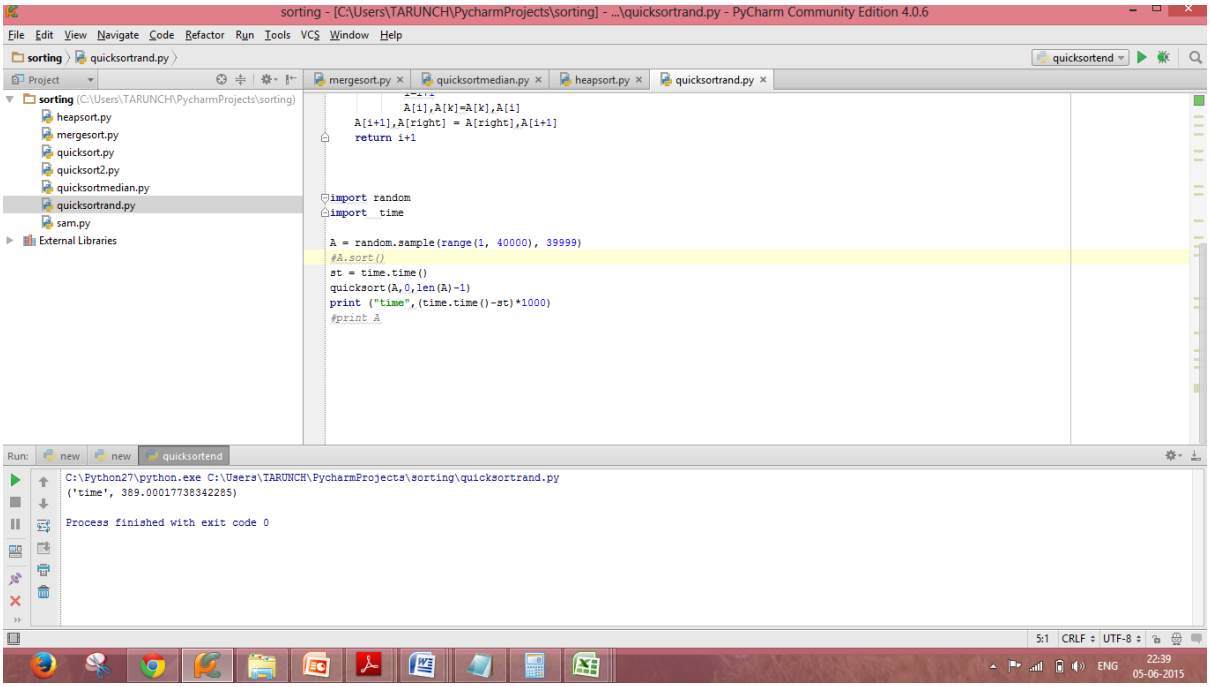

Heap sort(Reversely sorted):



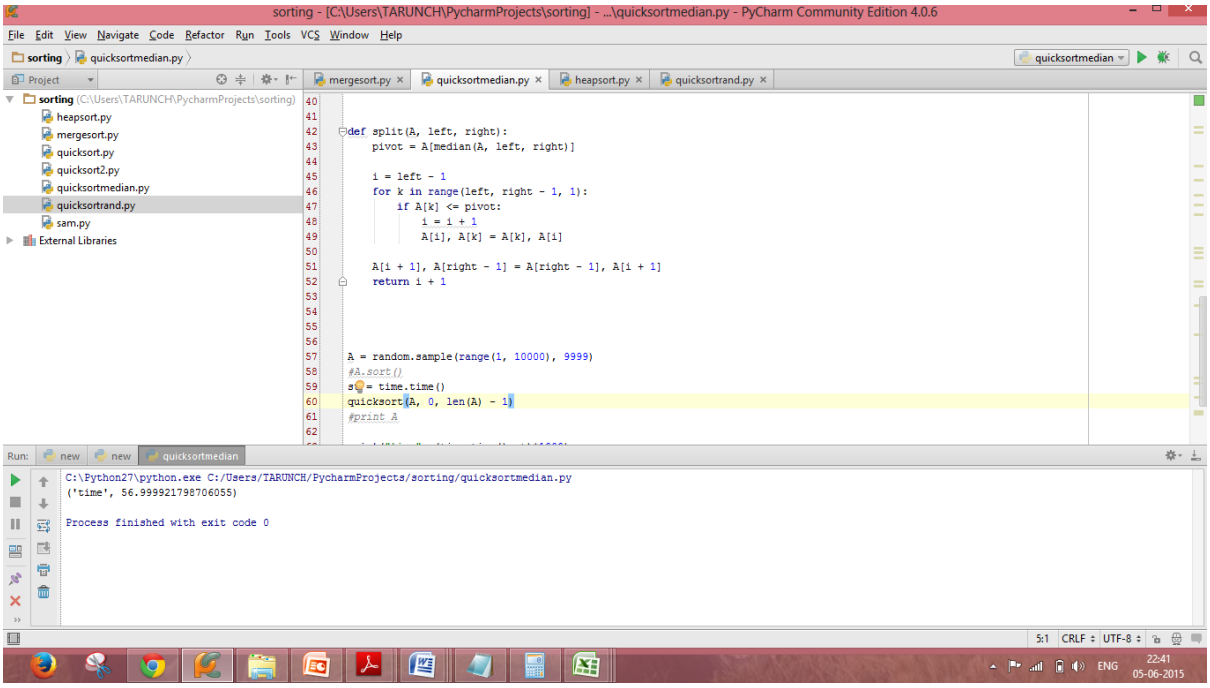
Quick sort(sorted):



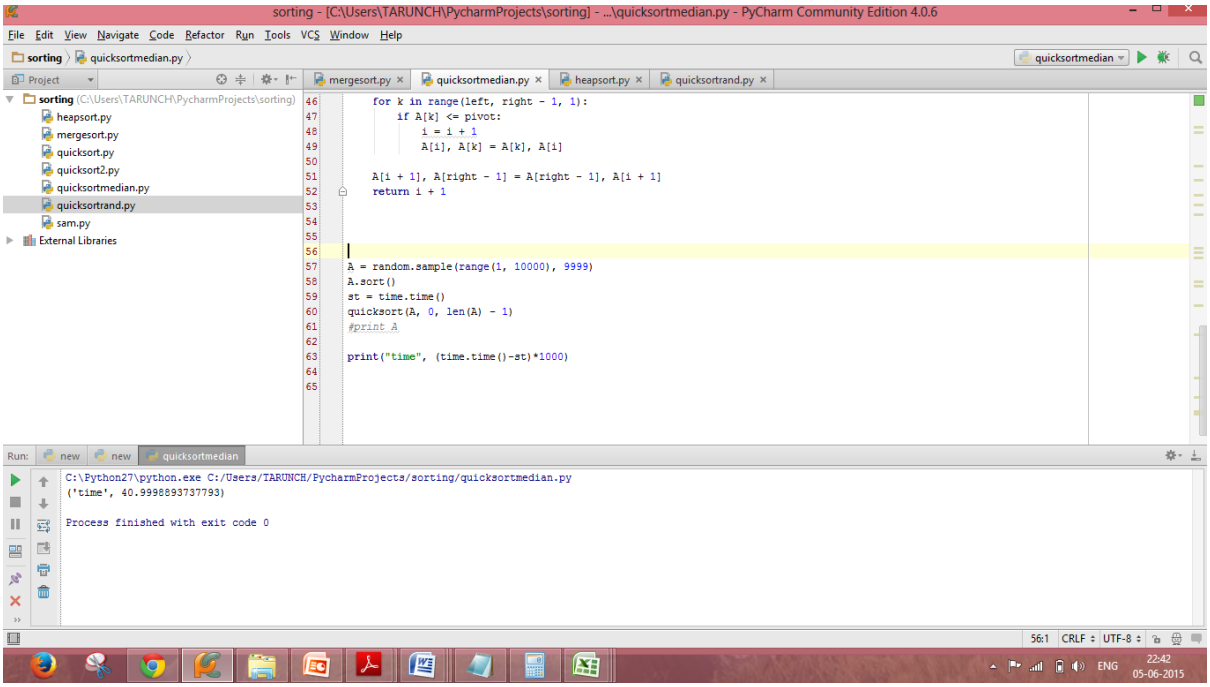

Quick sort (Unsorted):



Quick sort(Unsorted) median as pivot:




Quick sort(Sorted) median as pivot:



Quick sort(Reversely sorted) median as pivot: