TASK 2.2

My multithreaded quicksort was much better in terms of time efficiency. Multithreading can significantly improve the application performance of programs.
I decomposed the sorting in different ways. I had a constant MAX, representing the number of values to be sorted, I set the MAX to 1000 and used 4 threads which divided the values in such a way that each thread will be responsible for 250 values. While doing that I noticed that multithreading should be done to a limit which does not affect the performance of the algorithm. Therefore, you have to use right number of threads of the number of values of you have to be sorted. Because if you won't take care of it then you will observe that you are getting better results with the sequential programming. For example, if you have 100 or 200 values you don't need to use 8 threads as it may prove to be too much for these values. If right number of threads are used for the number of values than multithreading gives much better time efficiency as compare to sequential algorithm.