Report of the Project

Code Design:

Functions used:

Main(): To create randomized unsorted dynamic array, For calling the functions for quick sorting and displaying the array

Quicksort_last(): Recursive function call for version 1: last element as pivot

Quicksort_random(): Recursive function call for version2: random element as pivot

Quicksort_median(): Recursive function call for version3: median of medians as pivot

Select(): Function created by using select algorithm to find median of median

Partition(): Function used to partition and fix position of pivot as kth smallest element

R_partition(): Function used to create random index for pivot and swapping it to last element of array for partitioning

Swap(): Function used to swap number with pivot during partitioning or medium selection

Display(): Function used to display time taken to sort and number of comparisons in sorting

Data Structures used:

Dynamic arrays are used for four cases of number of inputs.

Static arrays and pointers are used as primitive data types while sorting.

Experimental Results:

Total time taken for first three cases is approx. 35 seconds.

Time taken for quick sort with n = 100000

a) QS1 took 0.024 seconds
b) QS2 took 0.017 seconds

Time taken for quick sort with n = 1000000

a) QS1 took 0.279 seconds
b) QS2 took 0.270 seconds

Time taken for quick sort with n = 10000000:

a) QS1 took 7.1 seconds
b) QS2 took 15.2 seconds

Time taken for quick sort with n = 100000000:

Experiment showed that time taken for such huge input is consumed more than 15 minutes hence I have omitted this case for user's convenience.

QS1(last element as pivot) performs better asymptotically than other two versions of quicksort.

Note: Experimental results for QS3(select algorithm) are not given this report. Because the quicksort I implemented for QS3 is taking long time compared to QS1 and QS2. I observed that even for n=1000000, QS# is taking time for than 2 minutes.

Hence I excluded the code for QS3 in program. But I request Professor to have look into it. I used select algorithm referred in Algorithm book in 7^{th} chapter.