**Lab 9**

1. (a) Write a C or a C++ program to implement Quicksort.

(b) Write a C or a C++ program to implement Randomized Quicksort.

( Randomize the Quicksort algorithm by using a random number generator.

Instead of always using A[p] (the first element of the subarray) as the

pivot, choose randomly element from the subarray A[p .. r]. You can do it

by exchanging element A[p] with an element chosen at random from A[p ...r] )

(c) Write a C or a C++ program to implement Randomized Quicksort by removing tail

recursion from Quicksort.

(Implement Randomized Quicksort algorithm by avoiding the second recursive call

to the Quicksort function).

{Note: Test the above three codes for a large number of inputs. Use array sizes of 100,

200, . . . , 500 integers. For each size, run the same code say 10,00 times and estimate the time taken by the program for one run. Compare the time taken by three codes}

2. Write a C or a C++ program to implement Heap sort.

3. Write a C or a C++ program to implement selection sort.

4. Write a C or a C++ program to implement Merge sort.