**B.Tech Third Year**

**Design & Analysis of Algorithms Lab**

**Lab Assignment #2**

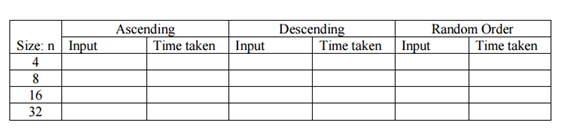
**Assignment 2:**

**Part-a**

1. Sort a given set of elements using the Quick sort method. ( using randomized pivot & middle element pivot chosen as the pivot element)
2. Determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be generated using the random number generator**.**
3. Also display the partial pass-wise sorting done. Sort algorithm with different array sizes:

**Note:**

* Program to be executed for various sizes of input.
* Fill the given table. Obtaining a constant value in the column “time taken” would prove: Is the complexity of quick sort same in all cases.

****

**Part-b:**

**Challenge**   
In practice, how much faster is Quicksort (in-place) than Insertion Sort? Compare the running time of the two algorithms by counting how many swaps or shifts each one takes to sort an array, and output the difference. You can modify your previous sorting code to keep track of the swaps. The number of swaps required by Quicksort to sort any given input have to be calculated. Keep in mind that the (using randomized pivot is chosen as the pivot, and that the array is sorted in-place as demonstrated in the explanation below.

Any time a number is lower than the partition, it should be "swapped", even if it doesn't actually move to a different location. Also ensure that you count the swap when the pivot is moved into place.