

## EXPERIMENT 3

PRAKASH P. BISWAS

9947

COMPS-B

### QUICK SORT

```
#include<stdio.h>
void quicksort(int no[25],int first,int last){
    int i, j, pivot, temp;

    if(first<last){
        pivot=first;
        i=first;
        j=last;

        while(i<j){
            while(no[i]<=no[pivot]&& i<last)
                i++;
            while(no[j]>no[pivot])
                j--;
            if(i<j){
                temp=no[i];
                no[i]=no[j];
                no[j]=temp;
            }
        }

        temp=no[pivot];
        no[pivot]=no[j];
        no[j]=temp;
        quicksort(no,first,j-1);
        quicksort(no,j+1,last);

    }
}

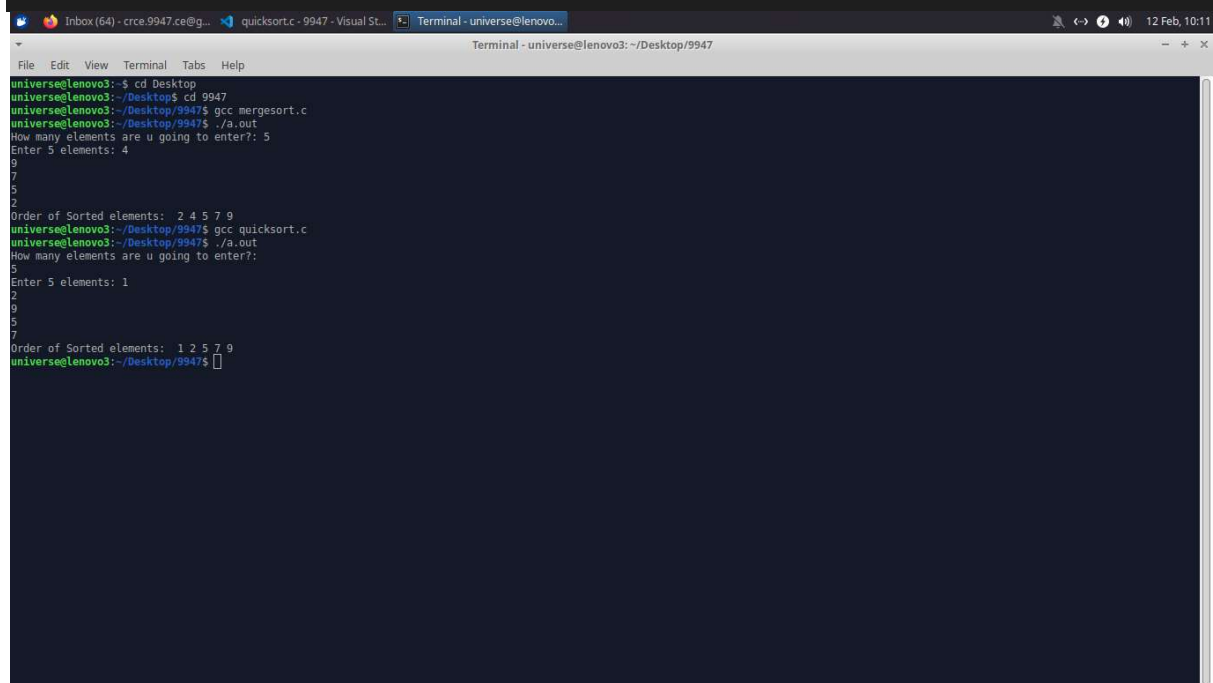
int main(){
    int i, count, no[25];

    printf("How many elements are u going to enter?: ");
    scanf("%d",&count);

    printf("Enter %d elements: ", count);
    for(i=0;i<count;i++)
        scanf("%d",&no[i]);
}
```

```
quicksort(no,0,count-1);

printf("Order of Sorted elements: ");
for(i=0;i<count;i++)
    printf(" %d",no[i]);
printf("\n");
return 0;
}
```



The screenshot shows a terminal window with the following output:

```
universe@lenovo3:~$ cd Desktop
universe@lenovo3:~/Desktop$ cd 9947
universe@lenovo3:~/Desktop/9947$ gcc mergesort.c
universe@lenovo3:~/Desktop/9947$ ./a.out
How many elements are u going to enter?: 5
Enter 5 elements: 4
9
7
5
2
Order of Sorted elements: 2 4 5 7 9
universe@lenovo3:~/Desktop/9947$ gcc quicksort.c
universe@lenovo3:~/Desktop/9947$ ./a.out
How many elements are u going to enter?: 5
Enter 5 elements: 1
2
9
5
7
Order of Sorted elements: 1 2 5 7 9
universe@lenovo3:~/Desktop/9947$
```

## POSTLAB:

9947  
Comps-B

① Space complexity of Quick Sort depends on whether its implemented recursively or iteratively.

i) Recursive Quick sort:

- space complexity is  $O(n)$  in worst case.
- The complexity arises from the recursion stack which grow linearly with size of input array.

ii) Iterative Quicksort:

- space complexity is  $(O(\log n))$  in worst case.
- The complexity is due to the stack used explicitly to manage partitioning with its size of input array.