Page No.
Date

Experiment-5

Aim - To implement quicksort and analyse it's time complexity

Software used :- Turbo C++

Algorithm and Analysis :-

```
quicksort (arr[], low, high)
{ if (low < high)
    { pi = partition (arr, low, hight)
      quicksort (arr, low, pi-1).
      quicksort ( arr, pi+1 , high)
    }
}
partition (arr[], low, high)
    pivot = arr[high]
     i = (low-1)
    for ( j=low to j<=high-1  j++)
        if (arr[j] <= pivot)
            i++
            swap arr[i] and arr[j]
    else
    swap arr[i+1] and arr [high]
    return (i+1)
```

Analysis.

Worst Case :- The worst case occurs when partition process analyse picks greatest or smallest element as pivot.

Recurrence relation.

$$T(n) = T(0) + T(n-1) + \Theta(n)$$

$$T(n) = T(n-1) + \Theta(n)$$

using masters method $a=1$, $b=1$

$$f(n) = n^1 \Rightarrow c=1$$

using 2nd case

$$T(n) = O(n^{k+1}) = O(^{1+1}) = O(n^2)$$

Best case :- The best case occurance when partition process always picks middle element as pivot. Recurrence relation is :-

$$T(n) = 2T(n/2) + \Theta(n)$$

using masters theorem $\Rightarrow$ $a=2$, $b=2$ $f(n) = cn$

$$n^{\log_b a} = n\log_2^1 = n$$

As $n^{\log_b a} = f(n)$

By 2nd case

$$T(n) = f(n) \log n$$

$$T(n) = O(n \log n)$$

The average case is also given by $O(n \log n)$.

Result - Successfully implemented quicksort and analysed it's time complexity.