Name - Shantanu Deshpande
Class - SYCSE
Rollno. - 7
Batch - S1
Sub. - D.S

Practical No. 8

Aim: Implementation of quick sort

Theory:

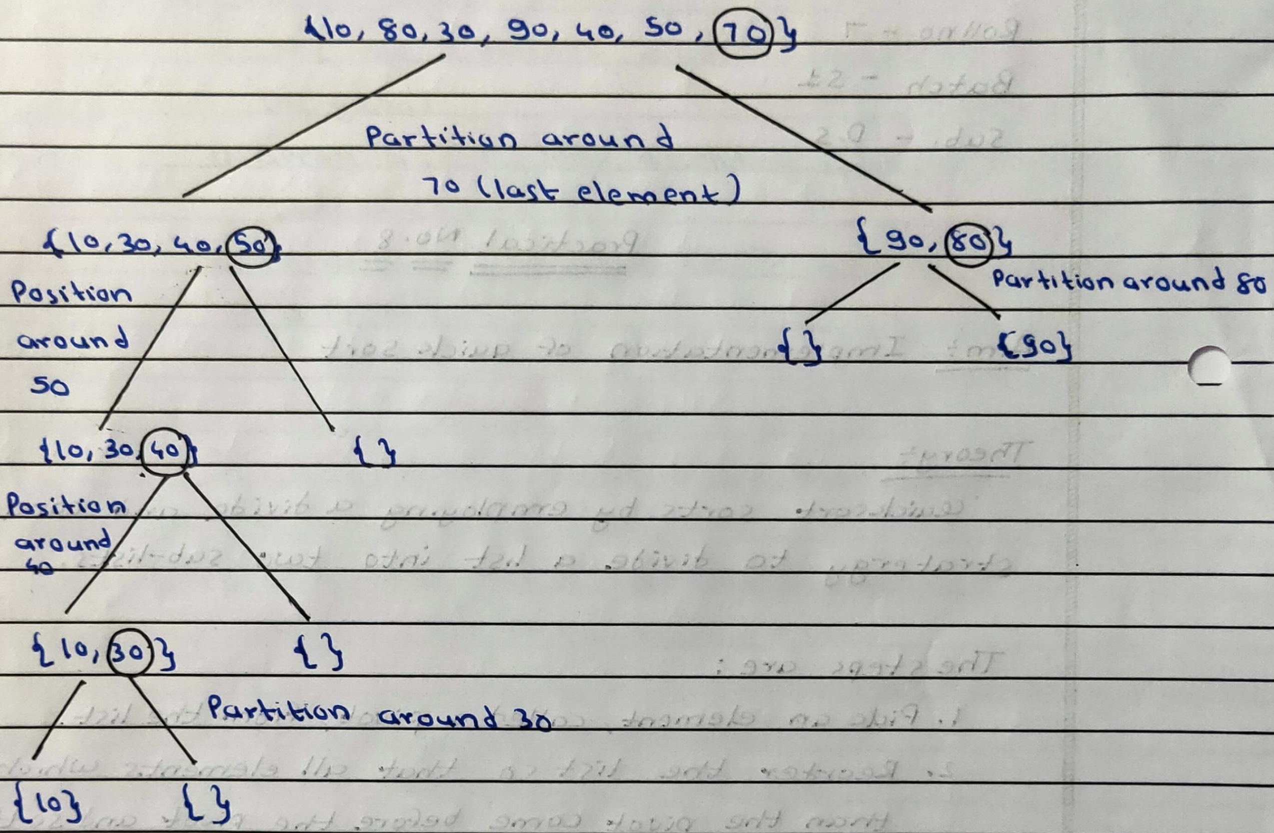
Quicksort sorts by employing a divide and conquer stratergy to divide a list into two sub-lists.

The steps are:

1. Pick an element, called a pivot, from the list.
2. Reorder the list so that all elements which are less than the pivot come before the pivot and so that all elements greater than pivot come after it (equal values can go either way). After this partitioning the pivot is in its final position. This is called the partition operation.
3. Recursively sort the sub-list of lesser elements and the sub-list of greater elements

The base case of the recursion are lists of size zero or one, which are always sorted.

Example :


{10, 80, 30, 90, 40, 50, 70}
Partition around
70 (last element)
{10, 30, 40, 50}
{90, 80}
Partition around 80
{}
{90}
Position
around
50
{10, 30, 40}
{}
Position
around
40
{10, 30}
{}
Partition around 30
{10}
{}

Algorithm :-

```
Algorithm Quick sort (p, q)
  // sorts the elements a[p], --- a[q] which reside in
     the global
  // array a[1:n] into ascending order; a [n+1] is
      considered to
  // be defined and must be >= all the elements in
      a [1:n].

{
   if (p<q) then // If there are more than one element
   {
      // divide P into two subproblems.
           j: = Partition (a, p, q+1);
                // j is the position of the partioning element
      // solve the subproblems.
           Quicksort (p, j-1);
           Quicksort (j+1, q);
      // There is no need for combining solutions.
   }
}
```

Algorithm partition (a,m,p)
{
  v := a[m]; i := m; j := p;
  repeat
  {
    repeat
      i := i+1;
    until (a[i] ≥ v);

    repeat
      j := j-1;
    until (a[j] ≤ v);

    if (i<j) then Interchange (a,i,j);
  } until (i≥j);

  a[m] := a[j]; a[j] := v; return j;
}

Conclusion :- Thus we have implemented Quick sort.

```c
#include<stdio.h>
int a[20] ;
int partition(int m,int p)
{
int v= a[m],i=m,j =p, temp;
do
{
do
{
i++;
}while(a[i] < v);
do
{
j--;
}while(a[j]> v);
if(i<j)
{
temp =a[i];
a[i] =a[j];
a[j]= temp;
}
} while(i<j);
a[m]=a[j];
a[j] = v;
return j;
}

void quicksort(int p, int q)
{
int j;
if (p <q)

{
j = partition (p,q+1);
quicksort(p,j-1);
quicksort(j+1 ,q);
}
}

void main()
{
int n,i, inf=32000;
printf("Enter The total no of elements");
scanf("%d", &n);
printf("enter Elements:");
for(i=1;i<=n ;i++)
scanf( "%d",&a[i]);
a[n+1]= inf;
quicksort(1,n);
printf("Sorted elements are:\n");
for(i=1;i<=n ;i++)
printf( "%d ",a[i]);
}
```

```
Enter The total no of elements5
enter Elements:45
75
90
1125
56
Sorted elements are:
45 56 75 90 1125

...Program finished with exit code 0
Press ENTER to exit console.
```