**Experiment No.8**

**Title:** Implementation of Quick Sort

**Problem Statement:** Write a C++ program for implementing quick sort.

**Algorithm:**

**Step 1 -**Consider the first element of the list as **pivot** (i.e., Element at first position in the list).

**Step 2 -**Define two variables i and j. Set i and j to first and last elements of the list respectively.

**Step 3 -**Increment i until a[i] > pivot then stop.

**Step 4 -**Decrement j until a[j] < pivot then stop.

**Step 5 -**If i < j then exchange a[i] and a[j].

**Step 6 -**Repeat steps 3,4 & 5 until i > j.

**Step 7 -**Exchange the pivot element with a[j] element.

**Code:**

#include <iostream>

using namespace std;

void quickSort(int[],int,int);

int partition(int[],int,int);

int main()

{

int a[50],n,i;

cout<<"How many elements to be sorted? :";

cin>>n;

cout<<"\nEnter the elements :";

for(i=0;i<n;i++) //To read the array elements

cin>>a[i];

quickSort(a,0,n-1);

cout<<"\nArray after sorting : ";

for(i=0;i<n;i++) //To print the sorted array

cout<<" "<<a[i]<<" ";

return 0;

}

void quickSort(int a[],int low,int up)

{

int j,i;

if(low<up)

{

j=partition(a,low,up);

// The following 3 blocks of code is to observe the partitioning

// and intermediate results

cout<<"\n";

for(i=low;i<=j-1;i++) //To print the array

cout<<" "<<a[i]<<" ";

cout<<"\n";

cout<<"\n";

for(i=j+1;i<=up;i++) //To print the array

cout<<" "<<a[i]<<" ";

cout<<"\n";

cout<<"\n";

for(i=0;i<=7;i++) //To print the array

cout<<" "<<a[i]<<" ";

cout<<"\n";

quickSort(a,low,j-1);

quickSort(a,j+1,up);

}

}

int partition(int a[],int low,int up)

{

int piv,i,j,temp;

piv=a[low]; //piv is the element whose final position is sought

i=low;

j=up+1;

cout<< "\n piv="<<piv<<"\n";

do

{

do

i++; // move up the array

while(a[i]< piv && i<=up);

do

j--; // move dpwn the array

while(piv<a[j]);

if(i<j)

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}while (i<j);

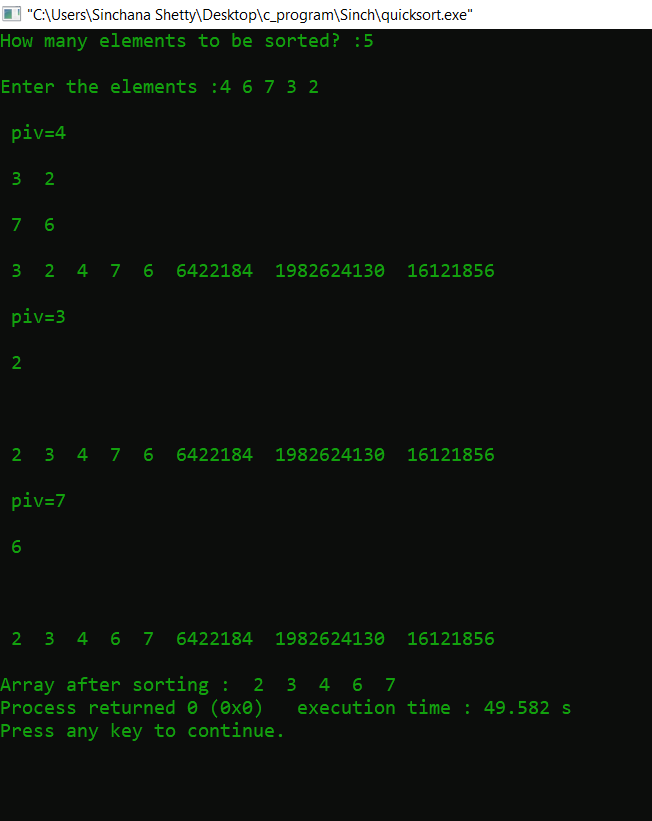
a[low]=a[j];

a[j]=piv;

return(j);

}

**Results:**



**Analysis(Limitations):**

Limitations of this program include choosing pivot, when it will be a last element or midpoint O(n2) is achieved and the difficulty of implementing the partitioning algorithm and the average efficiency for the worst case scenario, which is not offset by the difficult implementation.