Experiment No:8

Title: Implementation of Quick sort in C.

Problem Statement: Implementation of Quick sort in C.

Algorithm:

1. Choose the highest index value has pivot
2. Take two variables to point left and right of the list excluding pivot
3. left points to the low index
4. right points to the high
5. while value at left is less than pivot move right
6. while value at right is greater than pivot move left
7. if both step 5 and step 6 does not match swap left and right
8. if left ≥ right, the point where they met is new pivot

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);

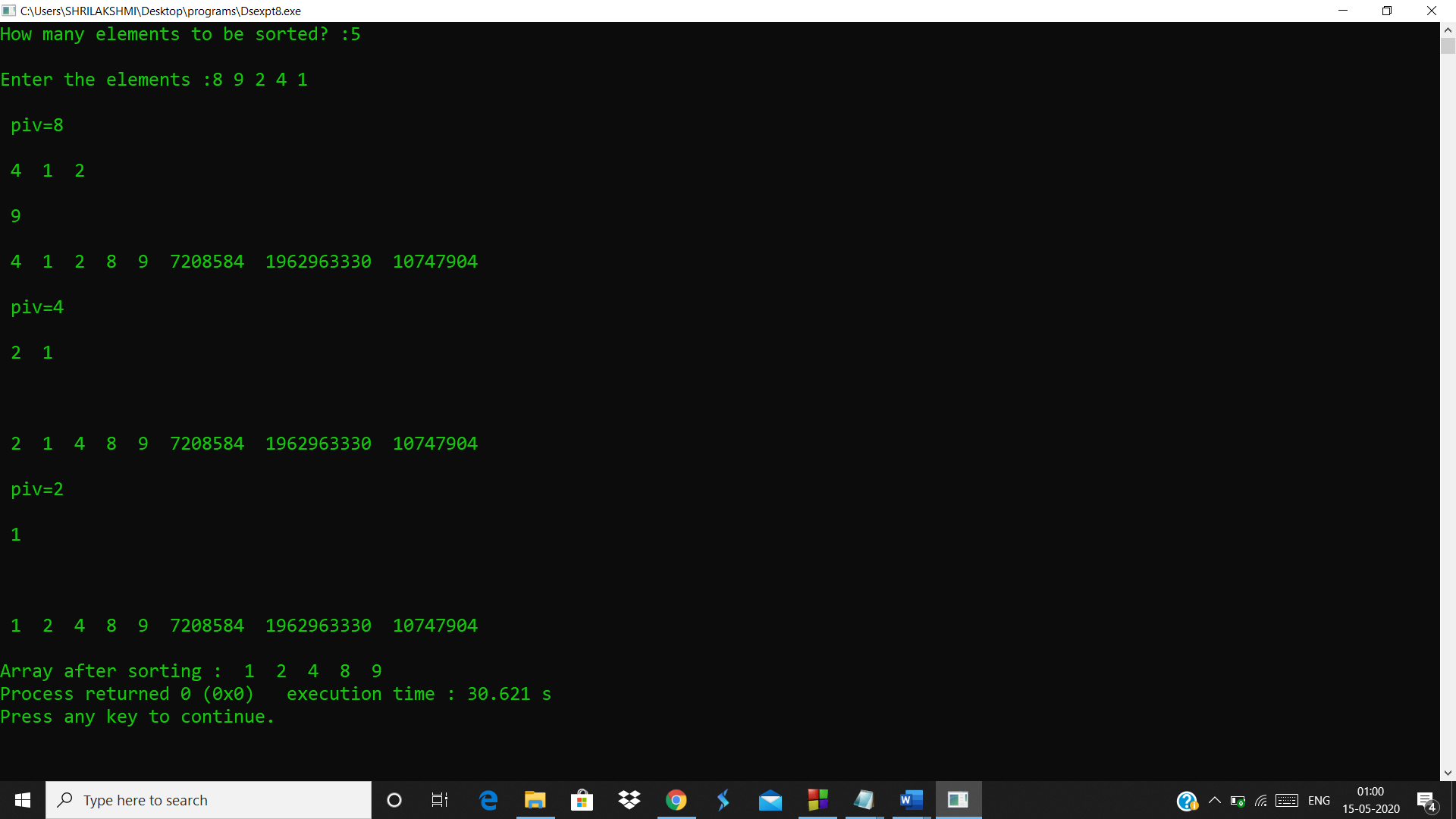
a[low]=a[j];

a[j]=piv;

return(j);

}

Output:



Analysis(Limitations):

* The major disadvantage of quick sort is that its worst-case performance is similar to average performances of the bubble, insertion or selections sorts. In general, the quick sort produces the most effective and widely used method of sorting a list of any item size.