**Competitive Programming**

1. Sorting Algorithms Set 2
   1. Merge Sort

**Source code:**

#include <iostream>

using namespace std;

void mergearray(int a[], int l, int m, int u)

{

int i,j,k;

int ll = m-l+1;

int ul = u-m;

int L[ll],R[ul];

for (i=0;i<ll;i++)

L[i]=a[l+i];

for (j=0;j<ul;j++)

R[j] = a[m+1+j];

i = 0;

j = 0;

k = l;

while (i<ll&&j<ul)

{

if (L[i]<=R[j])

{

a[k] = L[i];

i++;

}

else

{

a[k] = R[j];

j++;

}

k++;

}

while (i < ll)

{

a[k] = L[i];

i++;

k++;

}

while (j < ul)

{

a[k] = R[j];

j++;

k++;

}

}

void divide(int a[], int l, int u)

{

if (l < u)

{

int m = l+(u-l)/2;

divide(a,l,m);

divide(a, m+1,u);

mergearray(a,l,m,u);

}

}

int main()

{

int i,n;

cout<<"MERGE SORT \n";

cout<<"Enter the array size: ";

cin>>n;

int a[n];

cout<<"Enter array elements: ";

for(i=0;i<n;i++)

{

cin>>a[i];

}

divide(a,0,n-1);

cout<<"sorted array is: ";

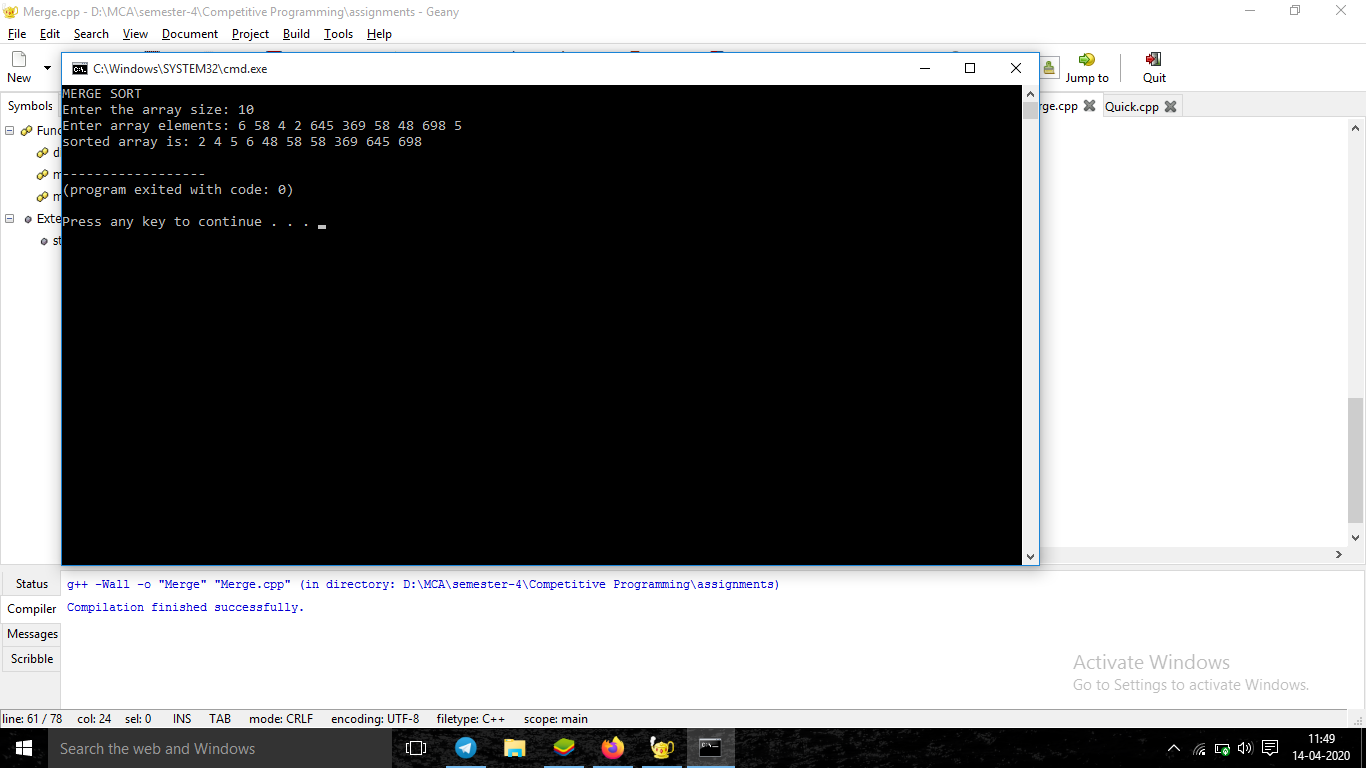
for(i=0;i<n;i++)

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

return 0;

}

**Output:**



* 1. Quick Sort

**Source code:**

#include<iostream>

using namespace std;

void swap(int\* a, int\* b)

{

int t = \*a;

\*a = \*b;

\*b = t;

}

int divide (int a[], int l, int h)

{

int p = a[h];

int i = (l - 1);

for (int j = l; j <= h - 1; j++)

{

if (a[j] < p)

{

i++;

swap(&a[i], &a[j]);

}

}

swap(&a[i + 1], &a[h]);

return (i + 1);

}

void quickSort(int a[], int l, int h)

{

if (l < h)

{

int pindex = divide(a, l, h);

quickSort(a, l, pindex - 1);

quickSort(a, pindex + 1, h);

}

}

int main()

{

int i,n;

cout<<"QUICK SORT \n";

cout<<"Enter the array size: ";

cin>>n;

int a[n];

cout<<"Enter array elements: ";

for(i=0;i<n;i++)

{

cin>>a[i];

}

quickSort(a,0,n-1);

cout<<"sorted array is: ";

for(i=0;i<n;i++)

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

return 0;

}

**Output:**

