

```
#include<iostream>
#include<stdlib.h>
#include<omp.h>
using namespace std;

void mergesort(int a[],int i,int j);
void merge(int a[],int i1,int j1,int i2,int j2);

void mergesort(int a[],int i,int j)
{
    int mid;
    if(i<j)
    {
        mid=(i+j)/2;

        #pragma omp parallel sections
        {

            #pragma omp section
            {
                mergesort(a,i,mid);
            }

            #pragma omp section
            {
                mergesort(a,mid+1,j);
            }
        }

        merge(a,i,mid,mid+1,j);
    }
}

void merge(int a[],int i1,int j1,int i2,int j2)
```

```

{
    int temp[1000];
    int i,j,k;
    i=i1;
    j=i2;
    k=0;

    while(i<=j1 && j<=j2)
    {
        if(a[i]<a[j])
        {
            temp[k++]=a[i++];
        }
        else
        {
            temp[k++]=a[j++];
        }
    }

    while(i<=j1)
    {
        temp[k++]=a[i++];
    }

    while(j<=j2)
    {
        temp[k++]=a[j++];
    }

    for(i=i1,j=0;i<=j2;i++,j++)
    {
        a[i]=temp[j];
    }
}

```

```

int main()
{
    int *a,n,i;
    cout<<"\n enter total no of elements=>";
}

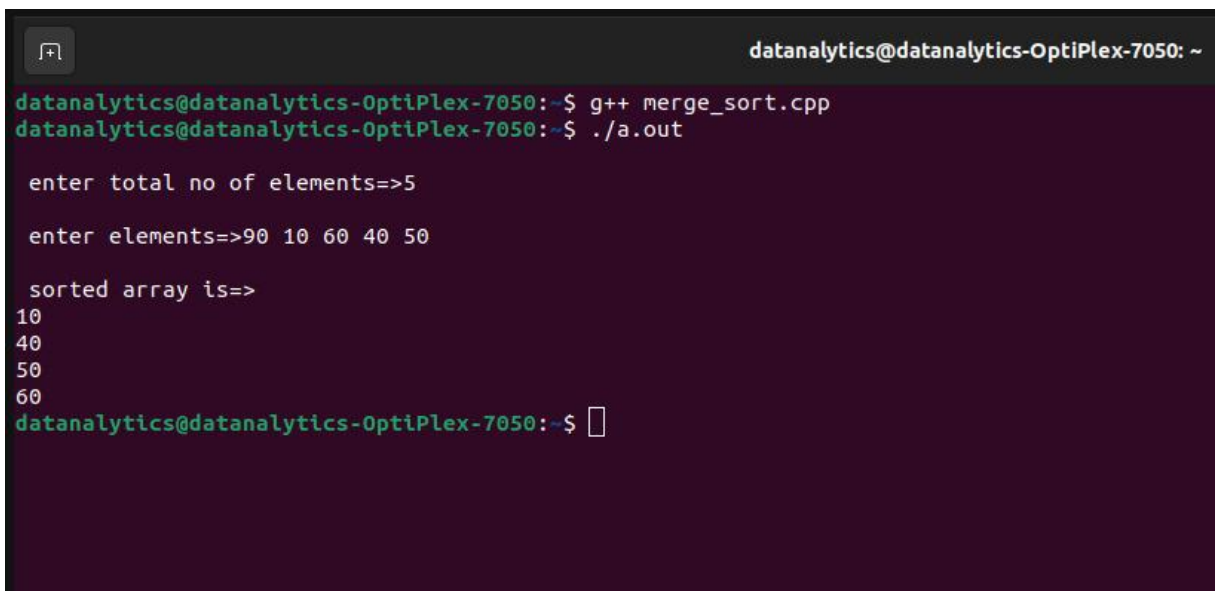
```

```

    cin>>n;
    a= new int[n];

    cout<<"\n enter elements=>";
    for(i=0;i<n;i++)
    {
        cin>>a[i];
    }
    // start=.....
    // #pragma omp....
    mergesort(a, 0, n-1);
    // stop.....
    cout<<"\n sorted array is=>";
    for(i=0;i<n;i++)
    {
        cout<<"\n"<<a[i];
    }
    // Cout<<Stop-Start
    return 0;
}

```



```

datanalytics@datanalytics-OptiPlex-7050: ~
datanalytics@datanalytics-OptiPlex-7050:~$ g++ merge_sort.cpp
datanalytics@datanalytics-OptiPlex-7050:~$ ./a.out

enter total no of elements=>5

enter elements=>90 10 60 40 50

sorted array is=>
10
40
50
60
datanalytics@datanalytics-OptiPlex-7050:~$ 

```