**Assignment 2**

**Parallel Merge Sort**

#include<iostream>

#include<omp.h>

using namespace std;

void merge(int \*,int,int,int);

void merge\_sort(int \*arr, int low, int high)

{

int mid;

if(low<high)

{

mid=(low+high)/2;

#pragma omp parallel sections

{

#pragma omp section

{

merge\_sort(arr,low,mid);

}

#pragma omp section

{

merge\_sort(arr,mid+1,high);

}

}

merge(arr,low,high,mid);

}

}

void merge(int \*arr,int low,int high,int mid)

{

int i,j,k,c[50];

i=low;

k=low;

j=mid+1;

while(i<=mid && j<=high)

{

if(arr[i]<arr[j])

{

c[k]=arr[i];

k++;

i++;

}

else

{

c[k]=arr[j];

k++;

j++;

}

}

while(i<=mid)

{

c[k]=arr[i];

k++;

i++;

}

while(j<=high)

{

c[k]=arr[j];

k++;

j++;

}

for(i=low;i<k;i++)

{

arr[i]=c[i];

}

}

int main()

{

omp\_set\_num\_threads(4);

int myarray[30],num;

cout<<"\nEnter number of elements to be sorted : ";

cin>>num;

cout<<"\nEnter elements : ";

for(int i=0;i<num;i++)

{

cin>>myarray[i];

}

merge\_sort(myarray,0,num-1);

cout<<"\nSorted array :"<<" ";

for(int i=0;i<num;i++)

{

cout<<myarray[i]<<" ";

}

}

**Output –**

Enter number of elements to be sorted : 5

Enter elements : 5 4 3 2 1

Sorted array : 1 2 3 4 5