#include<iostream>

#include<cstdlib>

#include<omp.h>

#include<time.h>

#define MAX 900000

using namespace std;

void merge(int array[],int low1, int high1,int low2,int high2)

{

int temp[MAX];

int i=low1,j=low2,k=0;

while(i<=high1 && j<=high2)

{

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

temp[k++]=array[i++];

else

temp[k++]=array[j++];

}

while(i<=high1)

temp[k++]=array[i++];

while(j<=high2)

temp[k++]=array[j++];

for(i=low1,j=0;i<=high2;i++,j++)

array[i]=temp[j];

}

void mergesort(int array[], int low, int high)

{

if(low<high)

{

int mid=(low+high)/2;

#pragma omp parallel sections

{

#pragma omp section

{

mergesort(array,low,mid);

}

#pragma omp section

{

mergesort(array,mid+1,high);

}

}/\*

mergesort(array,low,mid);

mergesort(array,mid+1,high);

merge(array,low,mid,mid+1,high);\*/

}

}

void display(int array[MAX], int n)

{

cout<<"\nArray : ";

for(int i=00;i<n;i++)

cout<<array[i]<<"\t";

}

int main()

{

int array[MAX],n;

cout<<"\nEnter the number of elements : ";

cin>>n;

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

{

array[i]=rand()%32;

}

display(array,n);

clock\_t start=clock();

mergesort(array,0,n-1);

clock\_t stop=clock();

display(array,n);

cout<<"\nTime required : "<<(double)(stop-start)\*1000/CLOCKS\_PER\_SEC<<" ms";

return 0;

}

/\*

gautami@Asus:~/HPC$ g++ par\_msort.cpp -fopenmp

gautami@Asus:~/HPC$ time ./a.out

Enter the number of elements : 50

Array : 7 6 9 19 17 31 10 12 9 13 26 11 18 27 3 6 28 2 20 24 27 8 7 13 22 26 14 3 19 31 9 26 6 18 13 23 17 24 3 26 5 29 5 23 24 9 30 20 11 18

Array : 7 6 9 19 17 31 10 12 9 13 26 11 18 27 3 6 28 2 20 24 27 8 7 13 22 26 14 3 19 31 9 26 6 18 13 23 17 24 3 26 5 29 5 23 24 9 30 20 11 18

Time required : 7.732 ms

real 0m2.610s

user 0m0.012s

sys 0m0.000s

\*/