Write a program to implement **Parallel Bubble Sort** and Merge sort using OpenMP. Use existing algorithms and measure the performance of sequential and parallel algorithms

#include<omp.h>

#include<iostream>

#include<time.h>

using namespace std;

int main()

{

// freopen("input.txt","r",stdin);

// freopen("output.txt","w",stdout);

int n;

cout<<"Enter the number of elements : ";

cin>>n;

cout<<endl;

int array[n] = {0};

for(int i=0;i<n;i++) array[i]=rand()%32;

cout<<"Original Array: ";

for(int i=0; i<n; i++) cout<<array[i]<<" ";

cout<<endl;

clock\_t start=clock();

int var = 0;

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

{

#pragma omp parallel for

for(int j=var; j<n-1; j+=2)

{

if(array[j] > array[j+1])

{

int temp = array[j];

array[j] = array[j+1];

array[j+1] = temp;

}

}

if(var == 0) var = 1;

else var = 0;

}

clock\_t stop=clock();

cout<<"\nFinal Array: ";

for(int i=0; i<n; i++) cout<<array[i]<<" ";

cout<<endl;

cout<<"\nTime required : "<<(double)(stop-start)<<" ms"<<endl;

return 0;

}

/\*

PS D:\C++> g++ -fopenmp parallel\_bub.cpp

PS D:\C++> ./a out

Enter the number of elements : 15

Original Array: 9 3 30 4 1 12 22 14 18 16 9 17 17 27 9

Final Array: 1 3 4 9 9 9 12 14 16 17 17 18 22 27 30

Time required : 0 ms

\*/