NAME – GAUTAM CHANDRA SAHA

REG NO – 201900099

DATE – 02/03/2022

LAB5

PROGRAM

/\* AUTHOR : GAUTAM CHANDRA SAHA

DATE & TIME: Wed, March 02,2022 AT 00:52

\*/

#include<iostream>

#include<vector>

#include<omp.h>

using namespace std;

class serial {

public:

void oddEvenSort(vector<int> &arr) {

bool isSorted = false; // Initially array is unsorted

int n=arr.size();

while (!isSorted) {

isSorted = true;

// Perform Bubble sort on odd indexed element

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

if (arr[i] > arr[i+1])

swap(arr[i], arr[i+1]),isSorted = false;

// Perform Bubble sort on even indexed element

for (int i=0; i<=n-2; i=i+2)

if (arr[i] > arr[i+1])

swap(arr[i], arr[i+1]),isSorted = false;

}

}

}serial;

class parallel{

public:

void oddEvenSort(vector<int> &arr) {

bool isSorted = false; // Initially array is unsorted

int n=arr.size();

while (!isSorted) {

#pragma omp parallel sections

{

isSorted = true;

#pragma omp section

{

// Perform Bubble sort on odd indexed element

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

if (arr[i] > arr[i+1])

swap(arr[i], arr[i+1]),isSorted = false;

}

#pragma omp section

{

// Perform Bubble sort on even indexed element

for (int i=0; i<=n-2; i=i+2)

if (arr[i] > arr[i+1])

swap(arr[i], arr[i+1]),isSorted = false;

}

}

}

}

}parallel;

typedef vector<double> vd;

typedef vector<int> vi;

vd calc(int size){

vd ans;

vi arr(size);

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

arr[i]=rand()%size;

vi arr2(arr);//copy the arr

//sort the array in serial

double start\_time = omp\_get\_wtime();

parallel.oddEvenSort(arr);

double end\_time = omp\_get\_wtime();

ans.push\_back(end\_time-start\_time);

//sort the array in parallel

start\_time = omp\_get\_wtime();

serial.oddEvenSort(arr2);

end\_time = omp\_get\_wtime();

ans.push\_back(end\_time-start\_time);

return ans;

}

int main(){

cout<<"BRICK SORT IMPLEMENTATION USING OPEN MP"<<endl<<endl;

auto \_time = calc(500);

printf("%s%32s%32s\n\n","No. of Inputs","Exec time for parallel env","Exec time for serial env");

printf("%d%33lf%32lf\n",500,\_time[0],\_time[1]);

\_time = calc(1000);

printf("%d%32lf%32lf\n",1000,\_time[0],\_time[1]);

\_time = calc(1200);

printf("%d%32lf%32lf\n",1200,\_time[0],\_time[1]);

return 0;

}

OUTPUT

Graphical user interface, text, application

Description automatically generated