Name: Mayur Goraksha Gaikwad

Roll No.: 19121028

Subject: High Performance Computing



**Experiment No:4**

Title : Implement Min, Max, Sum, and Average operations using Parallel Reduction

Code :

#include <iostream>

#include <omp.h>

using namespace std;

void parallel\_min(int \*arr, int n) {

int min\_val = arr[0];

#pragma omp parallel for reduction(min:min\_val)

for (int i = 1; i < n; i++) {

if (arr[i] < min\_val) {

min\_val = arr[i];

}

}

cout << "Minimum value in the array: " << min\_val << endl;

}

void parallel\_max(int \*arr, int n) {

int max\_val = arr[0];

#pragma omp parallel for reduction(max:max\_val)

for (int i = 1; i < n; i++) {

if (arr[i] > max\_val) {

max\_val = arr[i];

}

}

cout << "Maximum value in the array: " << max\_val << endl;

}

void parallel\_sum(int \*arr, int n) {

int sum = 0;

#pragma omp parallel for reduction(+:sum)

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

sum += arr[i];

}

cout << "Sum of the array elements: " << sum << endl;

}

void parallel\_avg(int \*arr, int n) {

int sum = 0;

#pragma omp parallel for reduction(+:sum)

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

sum += arr[i];

}

double avg = static\_cast<double>(sum) / static\_cast<double>(n);

cout << "Average value of the array elements: " << avg << endl;

}

int main() {

int arr[] = {5, 2, 9, 1, 5, 6};

int n = sizeof(arr) / sizeof(arr[0]);

parallel\_min(arr, n);

parallel\_max(arr, n);

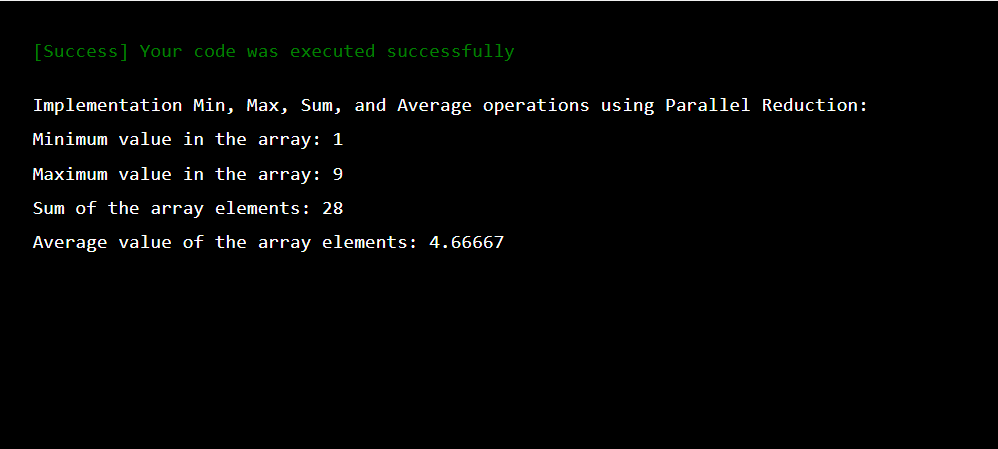
parallel\_sum(arr, n);

parallel\_avg(arr, n);

return 0;

}

Output:

****