HPC Lab Exp No.3

#include <omp.h>

#include <iostream>

#include <chrono>

using namespace std;

using namespace chrono;

void displayArray(int nums[], int length) {

    cout << "Nums: [";

    for (int i = 0; i < length; i++)

        cout << nums[i] << (i != length - 1 ? ", " : "");

    cout << "]\n";

}

void minOperation(int nums[], int length) {

    int minValue = nums[0];

#pragma omp parallel for reduction(min:minValue)

    for (int i = 0; i < length; i++)

        if (nums[i] < minValue) minValue = nums[i];

    cout << "Min value: " << minValue << endl;

}

void maxOperation(int nums[], int length) {

    int maxValue = nums[0];

#pragma omp parallel for reduction(max:maxValue)

    for (int i = 0; i < length; i++)

        if (nums[i] > maxValue) maxValue = nums[i];

    cout << "Max value: " << maxValue << endl;

}

void sumOperation(int nums[], int length) {

    int sum = 0;

#pragma omp parallel for reduction(+:sum)

    for (int i = 0; i < length; i++)

        sum += nums[i];

    cout << "Sum: " << sum << endl;

}

void avgOperation(int nums[], int length) {

    float sum = 0;

#pragma omp parallel for reduction(+:sum)

    for (int i = 0; i < length; i++)

        sum += nums[i];

    cout << "Average: " << (sum / length) << endl;

}

int main() {

    int length;

    cout << "Enter number of elements: ";

    cin >> length;

    int\* nums = new int[length];

    cout << "Enter " << length << " integers:\n";

    for (int i = 0; i < length; i++)

        cin >> nums[i];

    auto start = high\_resolution\_clock::now();

    displayArray(nums, length);

    minOperation(nums, length);

    maxOperation(nums, length);

    sumOperation(nums, length);

    avgOperation(nums, length);

    auto stop = high\_resolution\_clock::now();

    cout << "\nExecution time: " << duration\_cast<milliseconds>(stop - start).count() << " ms\n";

    delete[] nums;

    return 0;

}

OUTPUT:

