#include <iostream>

#include <vector>

#include <limits>

#include <climits>

#include <omp.h>

using namespace std;

int main() {

int n, num\_threads;

cout << "Enter number of elements: ";

cin >> n;

vector<int> data(n);

cout << "Enter the elements separated by space:\n";

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

cin >> data[i];

}

cout << "Enter number of threads to use: ";

cin >> num\_threads;

int min\_val = INT\_MAX;

int max\_val = INT\_MIN;

long long sum = 0;

#pragma omp parallel for reduction(min:min\_val) reduction(max:max\_val) reduction(+:sum) num\_threads(num\_threads)

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

if (data[i] < min\_val)

min\_val = data[i];

if (data[i] > max\_val)

max\_val = data[i];

sum += data[i];

}

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

cout << "\n--- Parallel Reduction Results ---\n";

cout << "Minimum: " << min\_val << "\n";

cout << "Maximum: " << max\_val << "\n";

cout << "Sum : " << sum << "\n";

cout << "Average: " << average << "\n";

return 0;

}