#include <iostream>

#include <vector>

#include <omp.h>

#include <climits>

#include <chrono>

using namespace std;

using namespace std::chrono;

void min\_reduction(vector<int>& arr) {

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

int min\_value = INT\_MAX;

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

for (int i = 0; i < arr.size(); i++) {

min\_value = min(min\_value, arr[i]);

}

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

auto duration = duration\_cast<microseconds>(stop - start);

cout << "Minimum value: " << min\_value << " (Time taken: " << duration.count() << " microseconds)" << endl;

}

void max\_reduction(vector<int>& arr) {

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

int max\_value = INT\_MIN;

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

for (int i = 0; i < arr.size(); i++) {

max\_value = max(max\_value, arr[i]);

}

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

auto duration = duration\_cast<microseconds>(stop - start);

cout << "Maximum value: " << max\_value << " (Time taken: " << duration.count() << " microseconds)" << endl;

}

void sum\_reduction(vector<int>& arr) {

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

int sum = 0;

#pragma omp parallel for reduction(+: sum)

for (int i = 0; i < arr.size(); i++) {

sum += arr[i];

}

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

auto duration = duration\_cast<microseconds>(stop - start);

cout << "Sum: " << sum << " (Time taken: " << duration.count() << " microseconds)" << endl;

}

void average\_reduction(vector<int>& arr) {

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

int sum = 0;

#pragma omp parallel for reduction(+: sum)

for (int i = 0; i < arr.size(); i++) {

sum += arr[i];

}

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

auto duration = duration\_cast<microseconds>(stop - start);

cout << "Average: " << (double)sum / arr.size() << " (Time taken: " << duration.count() << " microseconds)" << endl;

}

int main() {

int n;

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

cin >> n;

if (n <= 0) {

cout << "Invalid input. The number of elements must be greater than zero." << endl;

return 1;

}

vector<int> arr(n);

cout << "Enter " << n << " elements: ";

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

cin >> arr[i];

}

min\_reduction(arr);

max\_reduction(arr);

sum\_reduction(arr);

average\_reduction(arr);

return 0;

}