```cpp
// Implement Min, Max, Sum and Average operations using Parallel Reduction.

#include <iostream>
#include <vector>
#include <cstdint>
#include <omp.h>
using namespace std;

class ParallelReducer {
private:
    vector<int> arr;

public:
    ParallelReducer(const vector<int>& input) : arr(input) {}

    int parallelMin() {
        int min_val = INT32_MAX;
        #pragma omp parallel for reduction(min:min_val)
        for (int i = 0; i < arr.size(); i++) {
            if (arr[i] < min_val)
                min_val = arr[i];
        }
        return min_val;
    }

    int parallelMax() {
        int max_val = INT32_MIN;
        #pragma omp parallel for reduction(max:max_val)
        for (int i = 0; i < arr.size(); i++) {
            if (arr[i] > max_val)
                max_val = arr[i];
        }
        return max_val;
    }

    int parallelSum() {
        int sum = 0;
        #pragma omp parallel for reduction(+:sum)
        for (int i = 0; i < arr.size(); i++) {
            sum += arr[i];
        }
        return sum;
    }

    double parallelAverage() {
        double total = parallelSum();
        return total / arr.size();
    }
};

int main() {
    int size;
    cout << "Enter size of the array: ";
    cin >> size;
    vector<int> data(size);
    cout << "Enter " << size << " elements:\n";
    for (int i = 0; i < size; i++)
        cin >> data[i];

    ParallelReducer reducer(data);

    cout << "\nParallel Minimum: " << reducer.parallelMin() << endl;
    cout << "Parallel Maximum: " << reducer.parallelMax() << endl;
    cout << "Parallel Sum: " << reducer.parallelSum() << endl;
    cout << "Parallel Average: " << reducer.parallelAverage() << endl;

    return 0;
}

/*g++ -fopenmp filename.cpp -o filename.exe
Filename.exe
gcc -v takaycha command prompt la
https://www.winlibs.com/*/
```