```
#include<iostream>
#include<omp.h>
#include<bits/stdc++.h>

using namespace std;

void minimum(vector<int> array){
   int min = INT_MAX;
   double start = omp_get_wtime();
   for(auto i = array.begin(); i != array.end();i++){
      if(*i < min){
         min = *i;
      }
   }
   double end = omp_get_wtime();
   cout << "Minimum Element: " << min << endl;
   cout << "Time Taken: " << (end-start) << endl;
   int min_ele = INT_MAX;
   start = omp_get_wtime();
   #pragma omp parallel for reduction(min: min_ele)
      for(auto i = array.begin(); i != array.end();i++){
         if(*i < min_ele){
            min_ele = *i;
         }
      }
   end = omp_get_wtime();
   cout << "Minimum Element(Parallel Reduction): " << min_ele << endl;
   cout << "Time Taken: " << (end-start) << endl;

}

void maximum(vector<int> array){
   int max = INT_MIN;
   double start = omp_get_wtime();
   for(auto i = array.begin(); i != array.end();i++){
      if(*i > max){
         max = *i;
      }
   }
   double end = omp_get_wtime();
   cout << "Maximum Element: " << max << endl;
   cout << "Time Taken: " << (end-start) << endl;
   int max_ele = INT_MIN;
   start = omp_get_wtime();
   #pragma omp parallel for reduction(max: max_ele)
      for(auto i = array.begin(); i != array.end();i++){
         if(*i > max_ele){
            max_ele = *i;
         }
      }
   end = omp_get_wtime();
   cout << "Maximum Element(Parallel Reduction): " << max_ele << endl;
   cout << "Time Taken: " << (end-start) << endl;

}

void sum(vector<int> array){
   int sum = 0;
   double start = omp_get_wtime();
   for(auto i = array.begin(); i != array.end();i++){
```

```
        sum += *i;
    }
    double end = omp_get_wtime();
    cout << "Summation: " << sum << endl;
    cout << "Time Taken: " << (end-start) << endl;
    sum = 0;
    start = omp_get_wtime();
    #pragma omp parallel for reduction(+: sum)
        for(auto i = array.begin(); i != array.end();i++){
            sum += *i;
        }
    end = omp_get_wtime();
    cout << "Summation(Parallel Reduction): " << sum << endl;
    cout << "Time Taken: " << (end-start) << endl;

}
void average(vector<int> array){
    float avg = 0;
    double start = omp_get_wtime();
    for(auto i = array.begin(); i != array.end();i++){
        avg += *i;
    }
    double end = omp_get_wtime();
    cout << "Average: " << avg / array.size() << endl;
    cout << "Time Taken: " << (end-start) << endl;
    avg = 0;
    start = omp_get_wtime();
    #pragma omp parallel for reduction(+: avg)
        for(auto i = array.begin(); i != array.end();i++){
            avg += *i;
        }
    end = omp_get_wtime();
    cout << "Average(Parallel Reduction): " << avg / array.size() << endl;
    cout << "Time Taken: " << (end-start) << endl;

}

int main(){
    cout << "Enter number of elements in array: ";
    int N;
    int MAX = 1000;
    cin >> N;
    vector<int> array;
    for(int i = 0; i < N; i++){
        array.push_back(rand() % MAX);
    }
    minimum(array);
    maximum(array);
    sum(array);
    average(array);
    return 0;
}
```