Practical 3 : Parallel Reduction

```
#include <iostream>
#include <vector>
#include <omp.h>
#include <climits>
#include <cstdlib>

using namespace std;

void min_reduction(vector<int>& arr) {
  int min_value = INT_MAX;
  #pragma omp parallel for reduction(min: min_value)
  for (int i = 0; i < arr.size(); i++) {
    if (arr[i] < min_value) {
      min_value = arr[i];
    }
  }
  cout << "Minimum value: " << min_value << endl;
}

void max_reduction(vector<int>& arr) {
  int max_value = INT_MIN;
  #pragma omp parallel for reduction(max: max_value)
  for (int i = 0; i < arr.size(); i++) {
    if (arr[i] > max_value) {
      max_value = arr[i];
    }
  }
  cout << "Maximum value: " << max_value << endl;
}

void sum_reduction(vector<int>& arr) {
  int sum = 0;
   #pragma omp parallel for reduction(+: sum)
   for (int i = 0; i < arr.size(); i++) {
    sum += arr[i];
  }
  cout << "Sum: " << sum << endl;
}

void average_reduction(vector<int>& arr) {
  int sum = 0;
  #pragma omp parallel for reduction(+: sum)
  for (int i = 0; i < arr.size(); i++) {
    sum += arr[i];
  }
  cout << "Average: " << (double)sum / arr.size() << endl;
}

void print_arr(vector<int>&arr){
```

```
  if(arr.size() > 20 ){
    for(int i=0;i<20;i++){
       std::cout<<arr[i]<<" ";
    }
  }else{
    for(int i=0;i<arr.size();i++){
      std::cout<<arr[i]<<" ";
    }
  }

}

int main() {
  std::cout<<"This is Atharva Pingale's code";
  std::cout<<"\nPractical 3 : Parallel Reduction\n";
  vector<int> arr;
  int n;
  cout<<"Enter the size of the vector : ";
  cin>>n;
  for(int i=0;i<n;i++){
    int random_value = rand() % 99999;
    arr.push_back(random_value);
  }
  std::cout<<"Printing first 20 elements of Vector : \n";
  print_arr(arr);
  std::cout<<"\n";

#pragma omp parallel
{
   #pragma omp single
   {
      min_reduction(arr);
      max_reduction(arr);
      sum_reduction(arr);
      average_reduction(arr);
   }
}

return 0;

}
```

Output :

```
athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ g++ -fopenmp parallel_reduction.cpp -o parallel_reduction

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ ./parallel_reduction
This is Atharva Pingale's code
Practical 3 : Parallel Reduction
Enter the size of the vector : 15000
Printing first 20 elements of Vector :
41 18467 6334 26500 19169 15724 11478 29358 26962 24464 5705 28145 23281 16827 9961 491 2995 11942 4827 5436
Minimum value: 1
Maximum value: 32765
Sum: 247532296
Average: 16502.2
```

```
athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ g++ -fopenmp parallel_reduction.cpp -o parallel_reduction

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ ./parallel_reduction
This is Atharva Pingale's code
Practical 3 : Parallel Reduction
Enter the size of the vector : 25000
Printing first 20 elements of Vector :
41 18467 6334 26500 19169 15724 11478 29358 26962 24464 5705 28145 23281 16827 9961 491 2995 11942 4827 5436
Minimum value: 1
Maximum value: 32765
Sum: 412785710
Average: 16511.4
```

```
athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ g++ -fopenmp parallel_reduction.cpp -o parallel_reduction

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ ./parallel_reduction
This is Atharva Pingale's code
Practical 3 : Parallel Reduction
Enter the size of the vector : 32000
Printing first 20 elements of Vector :
41 18467 6334 26500 19169 15724 11478 29358 26962 24464 5705 28145 23281 16827 9961 491 2995 11942 4827 5436
Minimum value: 0
Maximum value: 32765
Sum: 528049601
Average: 16501.6
```