

1.Parallel Reduction (Min,Max,Sum,Average)

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

```
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;
}

int main() {
    int n;
    cout << "Enter the number of elements: ";
    cin >> n;

    vector<int> arr(n);
    cout << "Enter the 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;
}

```

Output:

```

Enter the number of elements: 10
Enter the elements: 10
20
30
40
50
60
70
80
90
100

```

```

Minimum value: 10
Maximum value: 100
Sum: 550
Average: 55

```