

NIKHIL N WAKODE

2022BCS0201

PARALLEL AND DISTRIBUTED COMPUTING

LAB-6 SET-6

---

Explore the following omp constructs:

1. parallel (different ways of creating threads)
2. single
3. master
4. worksharing using for
5. critical
6. barrier (explicit)

## Code:

```
threads.cpp > main()
1  #include <omp.h>
2  #include <iostream>
3  #include <iomanip>
4
5  using namespace std;
6
7  constexpr int N = 1000;
8
9  int main() {
10     int arr[N];
11     int tid = omp_get_thread_num();
12
13
14     cout << "Parallel Region:" << endl;
15     #pragma omp parallel
16     {
17         int tid = omp_get_thread_num();
18         #pragma omp critical
19         {
20
21             cout << "Hello from thread " << tid << endl;
22         }
23     }
24     cout << endl;
25
26     "
```

```
threads.cpp > main()
9  int main() {
27
28     #pragma omp parallel for
29     for (int i = 0; i < N; i++) {
30         arr[i] = i * 2;
31     }
32     cout << endl;
33     #pragma omp master
34     {
35         cout << "Master thread (ID: " << tid << endl;
36     }
37
38     cout << "Sections Region" << endl;
39     #pragma omp parallel sections
40     {
41         #pragma omp section
42         {
43             #pragma omp critical
44             {
45
46                 cout << "Section 1 running on thread " << omp_get_thread_num() << endl;
47             }
48         }
49         #pragma omp section
50         {
51             #pragma omp critical
52             {
53                 cout << "Section 2 running on thread " << omp_get_thread_num() << endl;
54             }
55         }
56     }
57     cout << endl;
58 }
```

```

58
59     cout << "Tasks " << endl;
60     #pragma omp parallel
61     {
62         #pragma omp single
63         {
64             #pragma omp task
65             {
66                 #pragma omp critical
67                 {
68                     cout << "Task 1 running on thread " << omp_get_thread_num() << endl;
69                 }
70             }
71             #pragma omp task
72             {
73                 #pragma omp critical
74                 {
75                     cout << "Task 2 running on thread " << omp_get_thread_num() << endl;
76                 }
77             }
78         }
79     }
80     cout << endl;
81
82
83
84
85     return 0;
86 }

```

Output:

```

• PS C:\Users\hp\Desktop\OpenMplab6> g++ -fopenmp threads.cpp -o thread
• PS C:\Users\hp\Desktop\OpenMplab6> ./thread
Parallel Region:
Hello from thread 1
Hello from thread 2
Hello from thread 3
Hello from thread 5
Hello from thread 4
Hello from thread 6
Hello from thread 0
Hello from thread 7

Master thread ID: 0
Sections Region
Section 2 running on thread 1
Section 1 running on thread 4

Tasks
Task 1 running on thread 1
Task 2 running on thread 6
• PS C:\Users\hp\Desktop\OpenMplab6> 

```

## Codes:

```
codes.cpp > main()
1  #include <omp.h>
2  #include <iostream>
3  #include <cstdlib>
4  #include <ctime>
5  using namespace std;
6
7  const int N = 1000;
8
9  int main() {
10     int data[N];
11     int sum = 0;
12
13
14     srand(time(nullptr));
15
16     #pragma omp parallel
17     {
18         int thread_id = omp_get_thread_num();
19         int num_threads = omp_get_num_threads();
20
21
22         #pragma omp single
23         {
24             cout << "Initializing data with " << num_threads << " threads.\n";
25             for (int i = 0; i < N; ++i) {
26                 data[i] = rand() % 100;
27             }
28         }
29 }
```

```
codes.cpp > main()
9  int main() {
28
29
30     cout << " Critical section to prevent race conditions when updating sum" << endl;
31     #pragma omp for
32     for (int i = 0; i < N; ++i) {
33
34         #pragma omp critical
35         {
36             sum += data[i];
37         }
38     }
39
40     cout << "Explicit barrier to synchronize all threads before continuing" << endl;
41     #pragma omp barrier
42
43
44
45     cout << " only one thread prints the final result" << endl;
46     #pragma omp single
47     {
48         cout << sum << "\n";
49     }
50 }
51
52 return 0;
53 }
54 }
```

## Output:

```
PS C:\Users\hp\Desktop\OpenMpLab6> ./code
Initializing data with 8 threads.
48255
PS C:\Users\hp\Desktop\OpenMpLab6> █
```

