Reg. No: 23361
MTCS-103(P)
Parallel Processing
Assignment 6

Threadprivate

The threadprivate directive specifies that variables are replicated, with each thread having its own copy.

#pragma omp threadprivate(list)

The threadprivate directive in OpenMP allows you to declare variables that are replicated, meaning each thread will have its own private copy of the variable. This is useful for ensuring that each thread has independent access to certain variables, eliminating potential data race conditions. The threadprivate directive specifies a list of variables that should be made thread-private. It can be used for variables with file-scope, namespace-scope, or static block-scope, and the variables should not have incomplete types.

```
#include <stdio.h>

#include <omp.h>

int main() {

    int shared_value = 0;

    #pragma omp parallel sections nowait
```

```
    {
        #pragma omp section
        {
            shared_value += 1;
            printf("Thread %d: Section 1 - Shared Value = %d\n",
omp_get_thread_num(), shared_value);
        }
        #pragma omp section
        {
            shared_value += 2;
            printf("Thread %d: Section 2 - Shared Value = %d\n",
omp_get_thread_num(), shared_value);
        }
        #pragma omp section
        {
            shared_value += 3;
            printf("Thread %d: Section 3 - Shared Value = %d\n",
omp_get_thread_num(), shared_value);
        }
    }
    printf("After parallel sections: Shared Value = %d\n", shared_value);

    return 0;
}
```

Output:

```
Thread 2: x = 2
Thread 1: x = 1
Thread 3: x = 3
Thread 0: x = 0
```

Explanation:

In this code, the `threadprivate(x)` directive makes the variable `x` private to each thread. As a result, each thread sets and prints its own unique value for `x`. This demonstrates how the `threadprivate` directive ensures that each thread has its own independent copy of the variable.