National Institute of Technology Karnataka Surathkal,
Mangalore - 575025

DEPARTMENT OF INFORMATION TECHNOLOGY

LAB ASSIGNMENT 4

Submitted for Parallel Computing (IT301) By

Gaurav Chaurasia 181CV155

To

Dr. Geetha V *Dept of IT, NITK Surathkal*

Code link
github

problem - 1

```c
#include <omp.h>
#include <stdio.h>
int main()
{
    int nthreads, tid;
#pragma omp parallel private(tid)
    {
        tid = omp_get_thread_num();
        printf("Hello world from thread = %d\n", tid);
        if (tid == 0)
        {
            nthreads = omp_get_num_threads();
            printf("Number of threads = %d\n", nthreads);
        }
    }

    return 0;
}
```

Output

```
❯ g++ -fopenmp 01.c
❯ export OMP_NUM_THREADS=2
❯ ./a.out
Hello world from thread = 1
Hello world from thread = 0
Number of threads = 2
```

PB 2 Check the output of following program:

```
/*ifparallel.c*/
#include <omp.h>
#include <stdio.h>
int main()
{
    int val;
    printf("Enter 0: for serial 1: for parallel\n");
    scanf("%d", &val);
#pragma omp parallel if (val)
    {
        if (omp_in_parallel())
            printf("Parallel val = %d id = %d\n", val,
omp_get_thread_num());
        else
            printf("Serial val = % d id = %d\n", val,
omp_get_thread_num());
    }
}
```

Output

```
❯ g++ -fopenmp 02.c
❯ ./a.out
Enter 0: for serial 1: for parallel
0
Serial val =  0 id = 0
❯ ./a.out
Enter 0: for serial 1: for parallel
1
Parallel val = 1 id = 0
Parallel val = 1 id = 1
```

PB 3 .Observe and record the output of following program

```
/*num_threads.c*/
#include <omp.h>
#include <stdio.h>
int main()
{
#pragma omp parallel num_threads(4)
    {
        int tid = omp_get_thread_num();
        printf("Hello world from thread = %d\n", tid);
    }
}
```

Output

```
❯ g++ -fopenmp 03.c
❯ ./a.out
Hello world from thread = 1
Hello world from thread = 0
Hello world from thread = 2
Hello world from thread = 3
```

PB 4 Write a C/C++ parallel program for adding corresponding elements of two arrays.

```
/*addarray.c*/
#include <omp.h>
#include <stdio.h>
int main()
{
    int i, n, chunk;
    int a[20], b[20], c[20];
    5 n = 20;
    chunk = 2;
    /*initializing array*/
    for (i = 0; i < n; i++)
    {
        a[i] = i * 2;
        b[i] = i * 3;
    }
#pragma omp parallel for default(shared) private(i)
schedule(static, chunk)
    {
        for (i = 0; i < n; i++)
        {
            c[i] = a[i] + b[i];
            printf("Thread id = %d i = %d, c[%d] = %d\n",
omp_get_thread_num(), i, i, c[i]);
        }
    }
}
```

number of threads = 4 and chunk = 2

Output

```
❯ g++ -fopenmp 03.c
❯ ./a.out
Thread id = 0 i = 0, c[0] = 0
Thread id = 0 i = 1, c[0] = 5
Thread id = 0 i = 8, c[0] = 40
Thread id = 0 i = 9, c[0] = 45
Thread id = 0 i = 16, c[0] = 80
Thread id = 0 i = 17, c[0] = 85
Thread id = 3 i = 6, c[0] = 30
```

```
Thread id = 3 i = 7, c[0] = 35
Thread id = 3 i = 14, c[0] = 70
Thread id = 3 i = 15, c[0] = 75
Thread id = 1 i = 2, c[0] = 10
Thread id = 1 i = 3, c[0] = 15
Thread id = 1 i = 10, c[0] = 50
Thread id = 1 i = 11, c[0] = 55
Thread id = 1 i = 18, c[0] = 90
Thread id = 1 i = 19, c[0] = 95
Thread id = 2 i = 4, c[0] = 20
Thread id = 2 i = 5, c[0] = 25
Thread id = 2 i = 12, c[0] = 60
Thread id = 2 i = 13, c[0] = 65
```

number of threads = 4 and chunk = 3

Output

```
❯ g++ -fopenmp 03.c
❯ ./a.out
Thread id = 0 i = 0, c[0] = 0
Thread id = 0 i = 1, c[0] = 5
Thread id = 0 i = 2, c[0] = 10
Thread id = 0 i = 12, c[0] = 65
Thread id = 0 i = 13, c[0] = 65
Thread id = 0 i = 14, c[0] = 70
Thread id = 1 i = 3, c[0] = 15
Thread id = 1 i = 4, c[0] = 20
Thread id = 1 i = 5, c[0] = 25
Thread id = 1 i = 15, c[0] = 75
Thread id = 1 i = 16, c[0] = 80
Thread id = 1 i = 17, c[0] = 85
Thread id = 2 i = 6, c[0] = 30
Thread id = 2 i = 7, c[0] = 35
Thread id = 2 i = 8, c[0] = 40
Thread id = 2 i = 18, c[0] = 90
Thread id = 2 i = 19, c[0] = 95
Thread id = 3 i = 9, c[0] = 45
Thread id = 3 i = 10, c[0] = 50
Thread id = 3 i = 11, c[0] = 55
```

Thank you