

Parallel and Distributed Computing
CSE4001
Fall Semester 2020-21

Lab Assignment 1

ISHAAN OHRI
18BCE0265

Aim:

Write a simple OpenMP program to demonstrate the parallel loop construct.

1. Use OMP_SET_THREAD_NUM() and OMP_GET_THREAD_NUM() to find the number of processing unit
2. Use function invoke to print 'Hello World'
3. To examine the above scenario, the functions such as omp_get_num_procs(),

omp_set_num_threads(), omp_get_num_threads(), omp_in_parallel(),
omp_get_dynamic() and omp_get_nested() are listed and the explanation is given below to explore the concept practically.

omp_set_num_threads() - takes an integer argument and requests that the Operating System provide that number of threads in subsequent parallel regions.
omp_get_num_threads() (integer function) - returns the actual number of threads in the current team of threads.

omp_get_thread_num() (integer function) - returns the ID of a thread, where the ID ranges from 0 to the number of threads minus 1. The thread with the ID of 0 is the master thread.

omp_get_num_procs() - returns the number of processors that are available when the function is called.

omp_get_dynamic() - returns a value that indicates if the number of threads available in subsequent parallel region can be adjusted by the run time. o

omp_get_nested() returns a value that indicates if nested parallelism is enabled.

`omp_set_num_threads()` and `omp_get_num_threads()`

Source Code:

Execution:

```
[ishanohri@pdc:~/Assignment_1$ gcc -fopenmp hello.c -o h  
[ishanohri@pdc:~/Assignment_1$ ./h  
Inside OpenMp:4  
Inside OpenMp:4  
Inside OpenMp:4  
Inside OpenMp:4  
ishanohri@pdc:~/Assignment_1$ ]
```

`omp_set_num_threads()` and `omp_get_thread_num()`

Source Code:

Execution:

```
[ishaanohri@pdc:~/Assignment_1$ gcc -fopenmp hello2.c -o h  
[ishaanohri@pdc:~/Assignment_1$ ./h  
Inside OpenMp Processor Number:0  
Inside OpenMp Processor Number:1  
Inside OpenMp Processor Number:2  
Inside OpenMp Processor Number:3  
ishaanohri@pdc:~/Assignment_1$
```

Invoking ‘Hello World’

Source Code:



```
#include<omp.h>
#include<stdio.h>

void main(){
    #pragma omp parallel
    {
        printf("Hello World\n");
    }
}

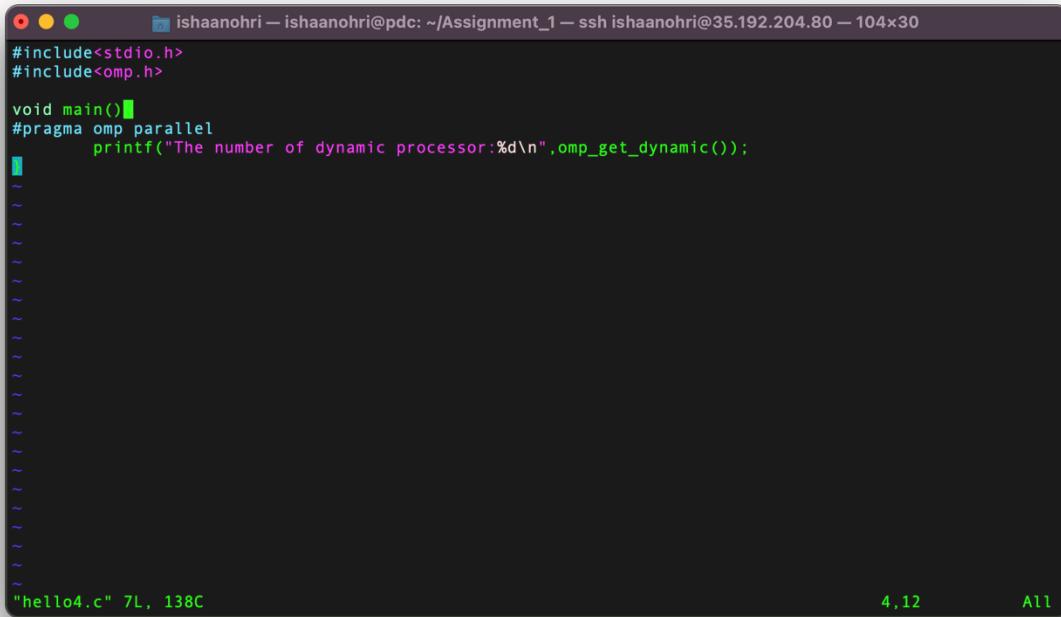
"hello3.c" 9L, 126C
```

Execution:

```
[ishanohri@pdc:~/Assignment_1$ gcc -fopenmp hello3.c -o h  
[ishanohri@pdc:~/Assignment_1$ ./h  
Hello World  
Hello World  
ishanohri@pdc:~/Assignment_1$
```

omp_get_dynamic()

Source Code:



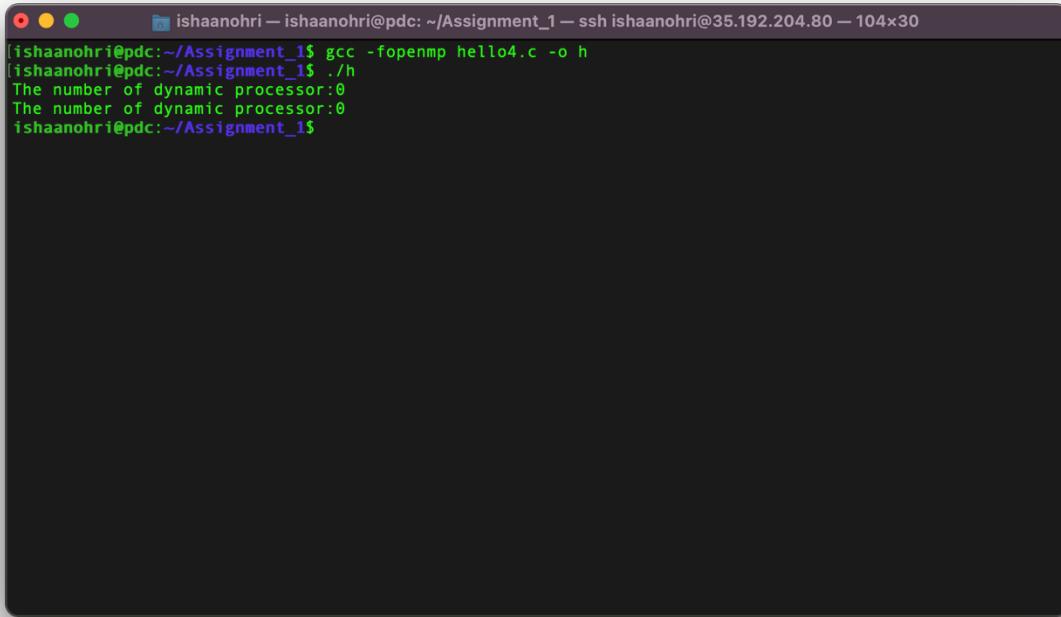
```
#include<stdio.h>
#include<omp.h>

void main()
{
    #pragma omp parallel
        printf("The number of dynamic processor:%d\\n",omp_get_dynamic());
}
```

"hello4.c" 7L, 138C

4,12 All

Execution:



```
[ishaanohri@pdc:~/Assignment_1$ gcc -fopenmp hello4.c -o h
[ishaanohri@pdc:~/Assignment_1$ ./h
The number of dynamic processor:0
The number of dynamic processor:0
ishaanohri@pdc:~/Assignment_1$ ]
```

omp_get_nested()

Source Code:

Execution:

```
[ishaanohri@pdc:~/Assignment_1$ gcc -fopenmp hello5.c -o h  
[ishaanohri@pdc:~/Assignment_1$ ./h  
The number of nested processor in parallel:0  
The number of nested processor in parallel:0  
ishaanohri@pdc:~/Assignment_1$ ]
```

omp_get_num_procs()

Source Code:

Execution:

```
[ishanohri@pdc:~/Assignment_1$ gcc -fopenmp hello6.c -o h  
[ishanohri@pdc:~/Assignment_1$ ./h  
The number of processors in parallel:2  
The number of processors in parallel:2  
ishanohri@pdc:~/Assignment_1$
```

omp_in_parallel()

Source Code:

Execution:

```
[ishanohri@pdc:~/Assignment_1]$ ssh ishanohri@35.192.204.80 - 104x30
[ishanohri@pdc:~/Assignment_1$ gcc -fopenmp hello7.c -o h
[ishanohri@pdc:~/Assignment_1$ ./h
Processes in parallel:1
Processes in parallel:1
ishanohri@pdc:~/Assignment_1$
```

Remarks:

The above experiment was conducted and all results along with the source code have been attached above in the document. The experiment was assisted by Dr Deepak. I thank sir for his assistance.