

Walchand College of Engineering, Sangli Department of Computer Science and Engineering

Class: Final Year (Computer Science and Engineering)

Year: 2021-22 Semester: 1

Course: High Performance Computing Lab

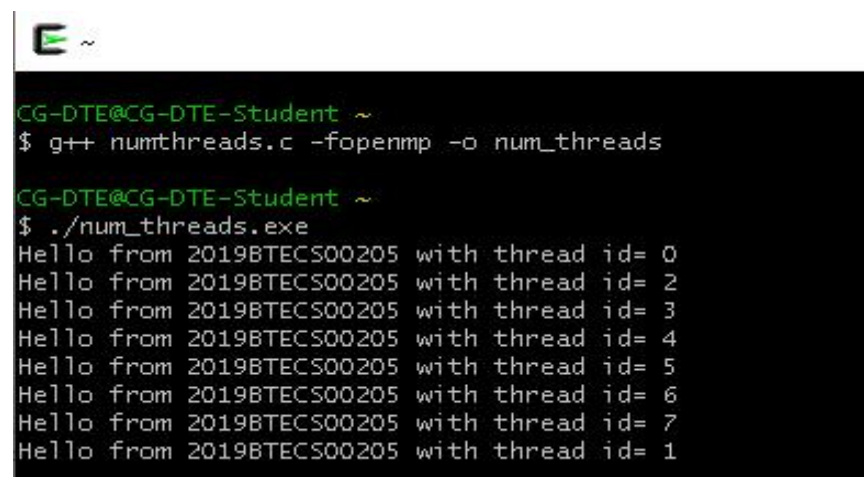
Practical No. 1

Exam Seat No : 2019BTECS00205

Name: Shweta Nandkumar Arbune

Problem Statement 1: How to create parallel program to print Hello from multiple threads.

Screenshot 1:



```
CG-DTE@CG-DTE-Student ~  
$ g++ numthreads.c -fopenmp -o num_threads  
  
CG-DTE@CG-DTE-Student ~  
$ ./num_threads.exe  
Hello from 2019BTECS00205 with thread id= 0  
Hello from 2019BTECS00205 with thread id= 2  
Hello from 2019BTECS00205 with thread id= 3  
Hello from 2019BTECS00205 with thread id= 4  
Hello from 2019BTECS00205 with thread id= 5  
Hello from 2019BTECS00205 with thread id= 6  
Hello from 2019BTECS00205 with thread id= 7  
Hello from 2019BTECS00205 with thread id= 1
```

Information :

```
#include <omp.h>  
#include <stdio.h>  
#include <stdlib.h>  
int main()  
{  
    omp_set_num_threads(10);  
  
    #pragma omp parallel  
    {  
        printf("Hello from 2019BTECS00205 with id= %d\n",omp_get_thread_num());  
    }  
}
```

Explanation:

1. We first have to include openmp header file i.e. `#include<omp.h>` .
2. Then, we have to specify parallel region by `#pragma omp parallel` directive.
3. The `pragma omp parallel` is used to fork additional threads to carry out the work enclosed in the parallel. The original thread having thread id 0 and it is the master thread .
4. We can set number of threads by using external variable or by using `omp_set_num_threads ()` function.
5. To compile the code we need to run the following command:
6. `gcc -o numthreads.c -fopenmp -o num_threads`
7. The output is then saved as `num_threads.exe`
8. Then to run we have to use the command : `./num_threads.exe`
9. Once the parallel region ended, all threads will get merged into the master thread.

Github Link: <https://github.com/shwetaarbune/HPC-LAB>