

ARYAMAN MISHRA

19BCE1027

LAB EXERCISE 1

Q.1. Write a Program to implement a “Hello World” Program using OpenMP.

```
#include<omp.h>

#include <stdio.h>

#include <stdlib.h>

int main()

{

#pragma omp parallel

{

printf("Hello World from thread=%d\n",omp_get_thread_num());

}

}
```

A screenshot of a Linux desktop environment. The top panel shows the date and time as 'Aug 10 01:06'. Below the panel is a dock with icons for Firefox, Files, and the Application Store. The main window is a 'Text Editor' titled 'omp_hello.c' located at '~/Desktop'. The code in the editor is the same as shown in the previous block, with line numbers 1 through 10. The line 'printf("Hello World from thread=%d\n",omp_get_thread_num());' is highlighted in light blue. The window has standard Linux window controls (Save, Close, etc.) in the title bar.

```
aryaman@aryaman-VirtualBox: ~/Desktop/19BCE1027PDC
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ cd 19BCE1027PDC
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ gcc -o omp_hello -fopenmp omp_hello.c
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ export OMP_NUM_THREADS=5
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ ./omp_hello
Hello World from thread=1
Hello World from thread=0
Hello World from thread=4
Hello World from thread=3
Hello World from thread=2
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$
```

Q.2. Write a Program to implement a “Hello World” with thread id and allow master thread to print the total number of threads.

```
#include<omp.h>

#include <stdio.h>

#include <stdlib.h>

int main()

{

int nthreads,tid;

/*Fork a team of threads with each thread having a private tid variable*/

#pragma omp parallel private(tid)

{
```

```
/*Obtain and print thread id*/

tid=omp_get_thread_num();

printf("Hello World from thread=%d\n",tid);

/*Only master thread does this*/

if(tid==0)

{

nthreads=omp_get_num_threads();

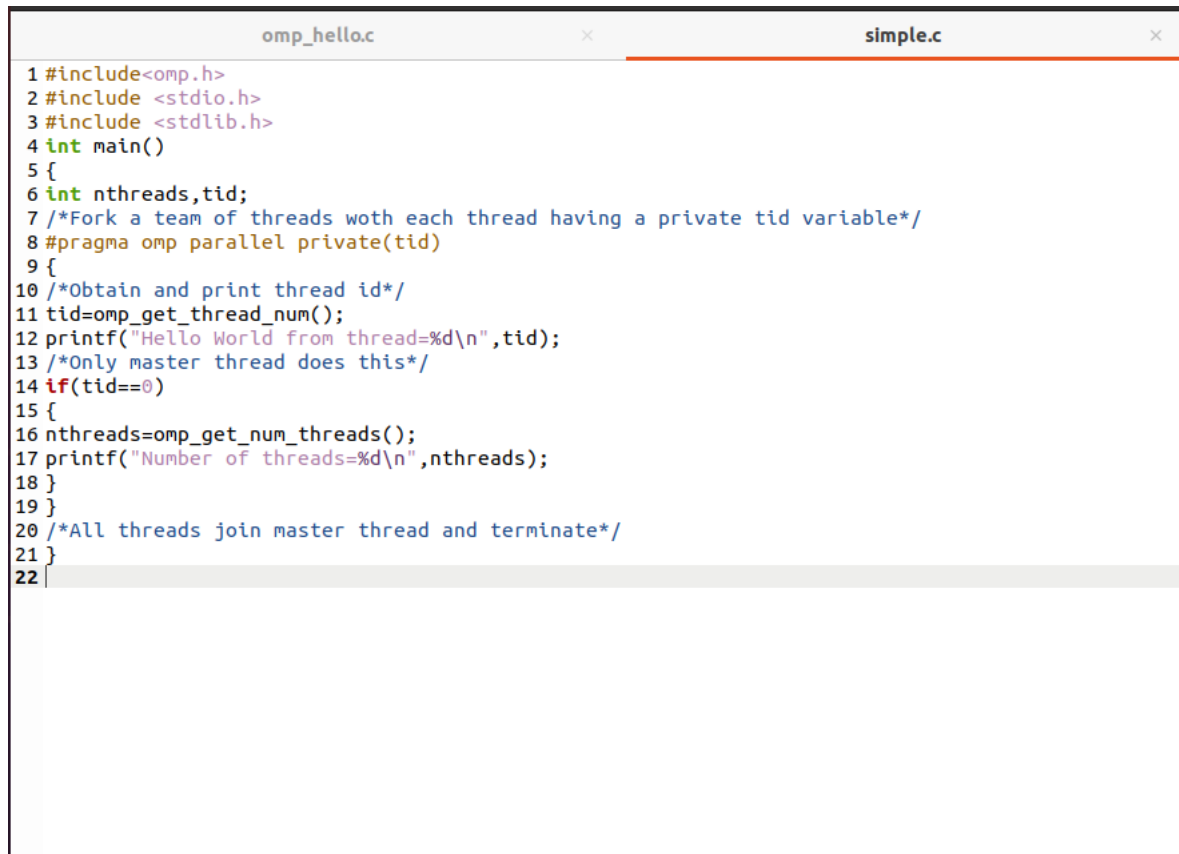
printf("Number of threads=%d\n",nthreads);

}

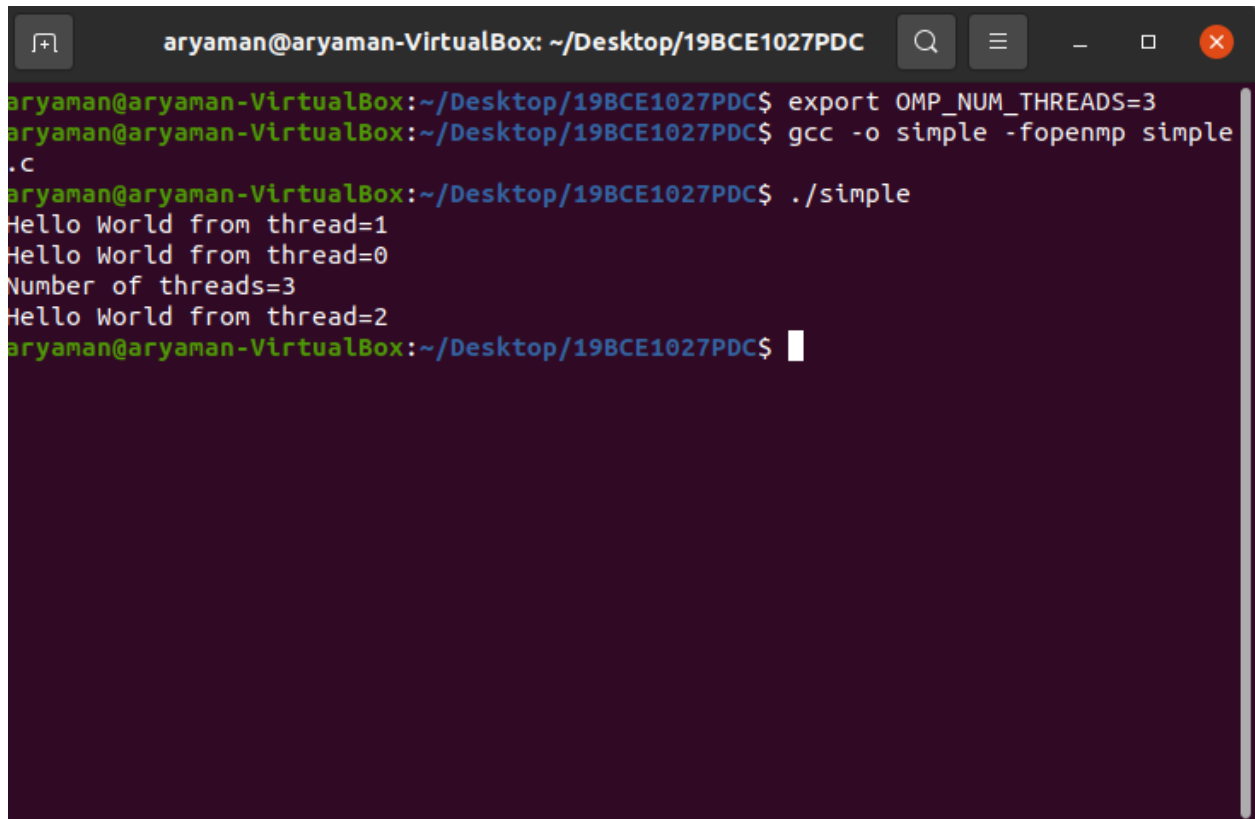
}

/*All threads join master thread and terminate*/

}
```



```
omp_hello.c × simple.c ×
1 #include<omp.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4 int main()
5 {
6 int nthreads,tid;
7 /*Fork a team of threads with each thread having a private tid variable*/
8 #pragma omp parallel private(tid)
9 {
10 /*Obtain and print thread id*/
11 tid=omp_get_thread_num();
12 printf("Hello World from thread=%d\n",tid);
13 /*Only master thread does this*/
14 if(tid==0)
15 {
16 nthreads=omp_get_num_threads();
17 printf("Number of threads=%d\n",nthreads);
18 }
19 }
20 /*All threads join master thread and terminate*/
21 }
22 |
```

A terminal window titled 'aryaman@aryaman-VirtualBox: ~/Desktop/19BCE1027PDC'. The window shows the execution of an OpenMP program. The user sets the environment variable 'OMP_NUM_THREADS=3', compiles a file named 'simple.c' into 'simple' using 'gcc -o simple -fopenmp simple.c', and then runs './simple'. The output shows three threads printing 'Hello World from thread=X' where X is 1, 0, and 2 respectively, followed by 'Number of threads=3'.

```
aryaman@aryaman-VirtualBox: ~/Desktop/19BCE1027PDC
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ export OMP_NUM_THREADS=3
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ gcc -o simple -fopenmp simple.c
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ ./simple
Hello World from thread=1
Hello World from thread=0
Number of threads=3
Hello World from thread=2
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$
```

Q.3. Write a Program to print name and registration number.

```
#include<omp.h>

#include <stdio.h>

#include <stdlib.h>

int main()

{

int nthreads,tid;

/*Fork a team of threads with each thread having a private tid variable*/
```

```
#pragma omp parallel private(tid)

{

/*Obtain and print thread id*/

tid=omp_get_thread_num();

printf("Aryaman Mishra\n");

/*Only master thread does this*/

if(tid==0)

{

nthreads=omp_get_num_threads();

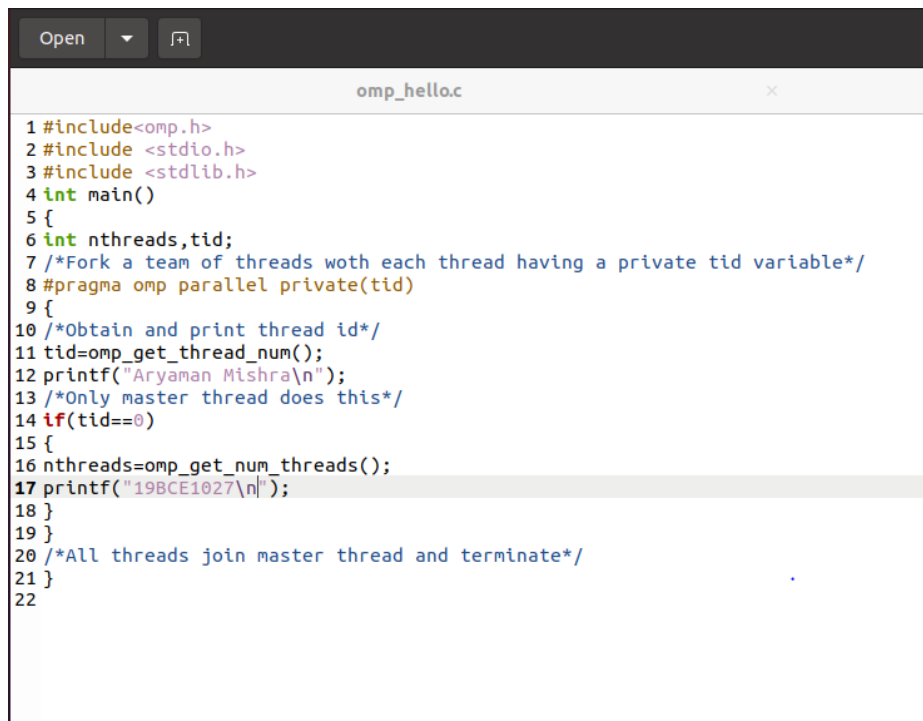
printf("19BCE1027\n");

}

}

/*All threads join master thread and terminate*/

}
```



The screenshot shows a code editor window titled "omp_hello.c" with a dark theme. The code is written in C and uses OpenMP for parallelism. It includes headers for `omp.h`, `stdio.h`, and `stdlib.h`. The `main` function declares `int nthreads, tid;` and forks a team of threads with `#pragma omp parallel private(tid)`. Inside the parallel region, each thread obtains its ID with `tid=omp_get_thread_num();` and prints "Aryaman Mishra\n". The master thread (tid==0) also prints "19BCE1027\n" and gets the total number of threads with `nthreads=omp_get_num_threads();`. The parallel region ends with `#pragma omp parallel private(tid)` and the program terminates with `return 0;`.

```
1 #include<omp.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4 int main()
5 {
6     int nthreads, tid;
7     /*Fork a team of threads with each thread having a private tid variable*/
8     #pragma omp parallel private(tid)
9     {
10    /*Obtain and print thread id*/
11    tid=omp_get_thread_num();
12    printf("Aryaman Mishra\n");
13    /*Only master thread does this*/
14    if(tid==0)
15    {
16        nthreads=omp_get_num_threads();
17        printf("19BCE1027\n");
18    }
19    }
20    /*All threads join master thread and terminate*/
21 }
22
```

```
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ export OMP_NUM_THREADS=10
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ gcc -o simple2 -fopenmp simple2.c
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$ ./simple2
Aryaman Mishra
Aryaman Mishra
19BCE1027
Aryaman Mishra
Aryaman Mishra
Aryaman Mishra
Aryaman Mishra
Aryaman Mishra
Aryaman Mishra
Aryaman Mishra
Aryaman Mishra
aryaman@aryaman-VirtualBox:~/Desktop/19BCE1027PDC$
```

RESULT:ALL 3 PROGRAMS HAVE BEEN SUCCESFULLY EXECUTED.