Walchand College of Engineering, Sangli

Department of Computer Science and Engineering

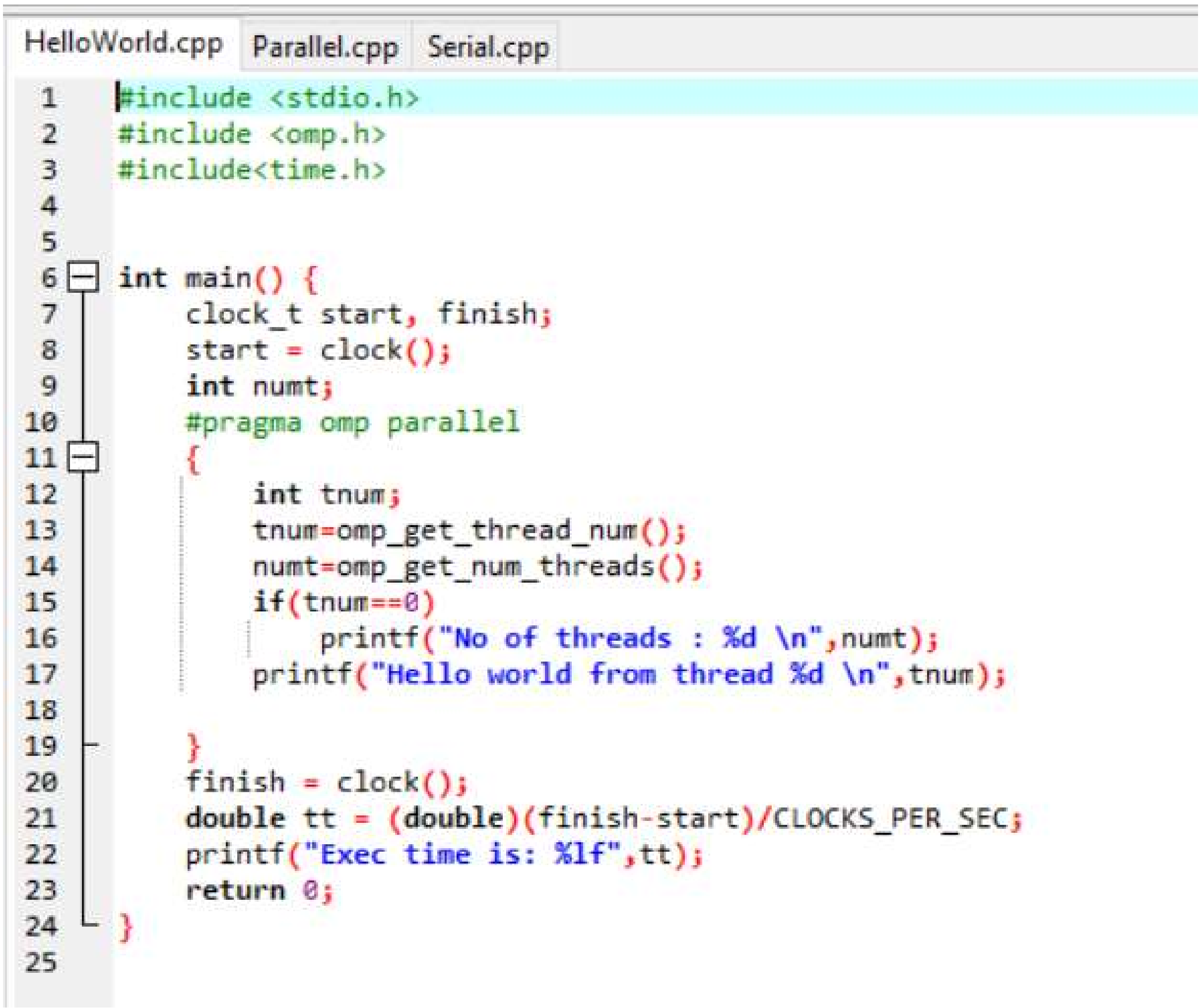
Name : Kajal Jitendra Pawar

PRN : 2019BTECS00010

Batch : B1

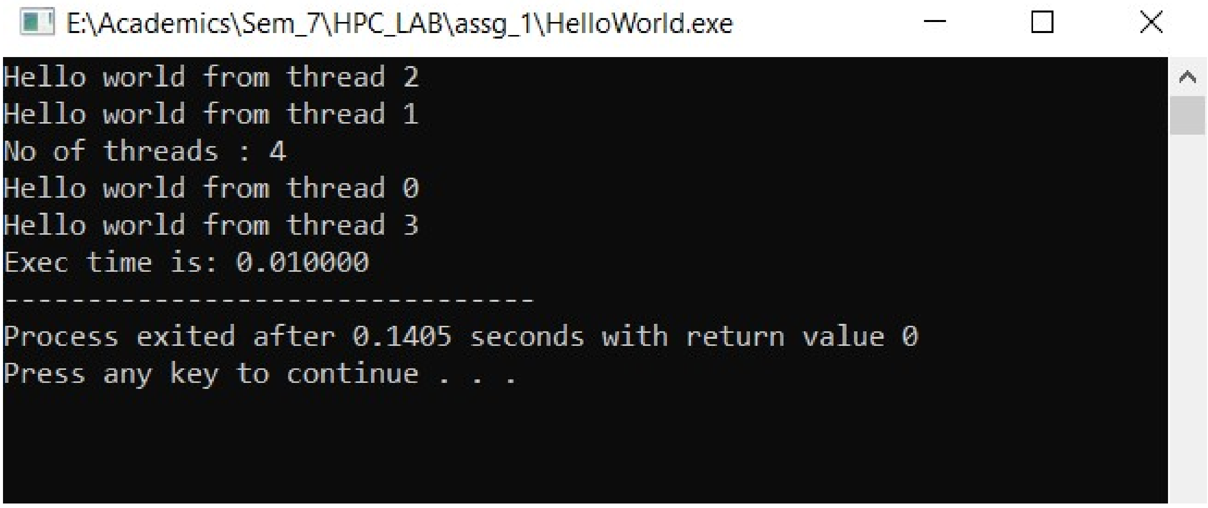
Practical No. 1

Q1.Hello World Program:

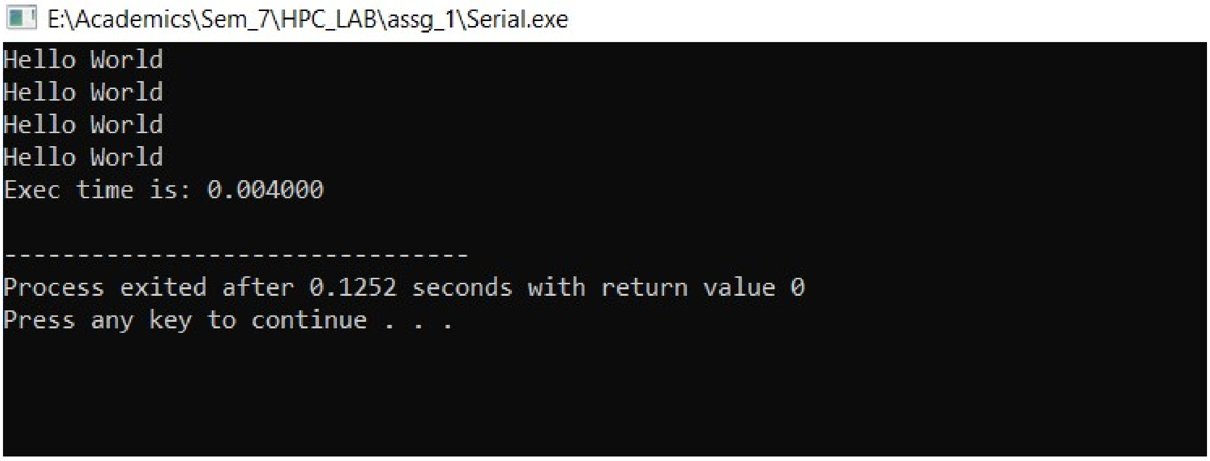


```
#include <stdio.h>
#include <omp.h>
#include<time.h>


int main() {
    clock_t start, finish;
    start = clock();
    int numt;
    #pragma omp parallel
    {
        int tnum;
        tnum=omp_get_thread_num();
        numt=omp_get_num_threads();
        if(tnum==0)
            printf("No of threads : %d \n",numt);
        printf("Hello world from thread %d \n",tnum);

    }
    finish = clock();
    double tt = (double)(finish-start)/CLOCKS_PER_SEC;
    printf("Exec time is: %lf",tt);
    return 0;
}

```

Output:



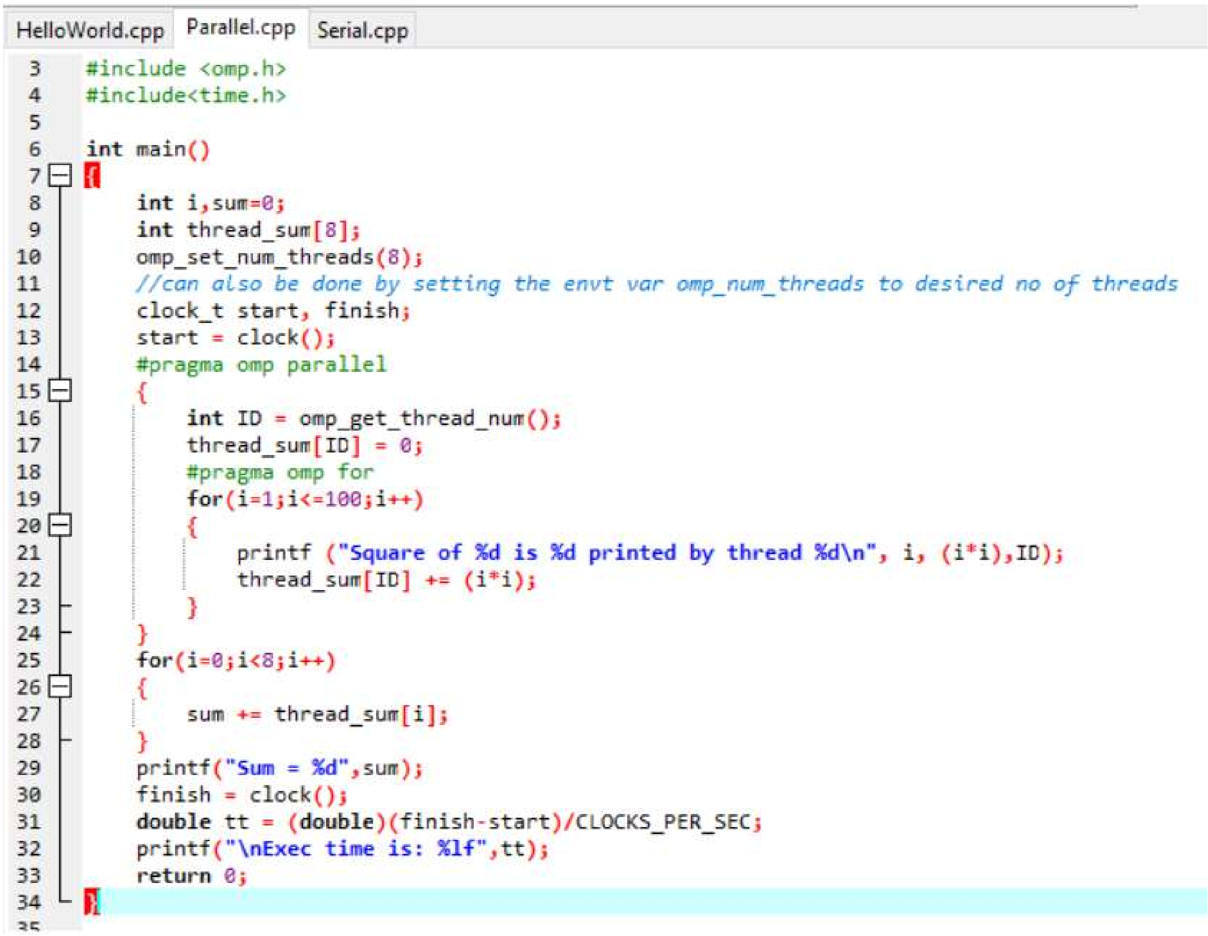
```
Hello world from thread 2
Hello world from thread 1
No of threads : 4
Hello world from thread 0
Hello world from thread 3
Exec time is: 0.010000
--------------------------------
Process exited after 0.1405 seconds with return value 0
Press any key to continue . . .
```



```
Hello World
Hello World
Hello World
Hello World
Exec time is: 0.004000

--------------------------------
Process exited after 0.1252 seconds with return value 0
Press any key to continue . . .
```

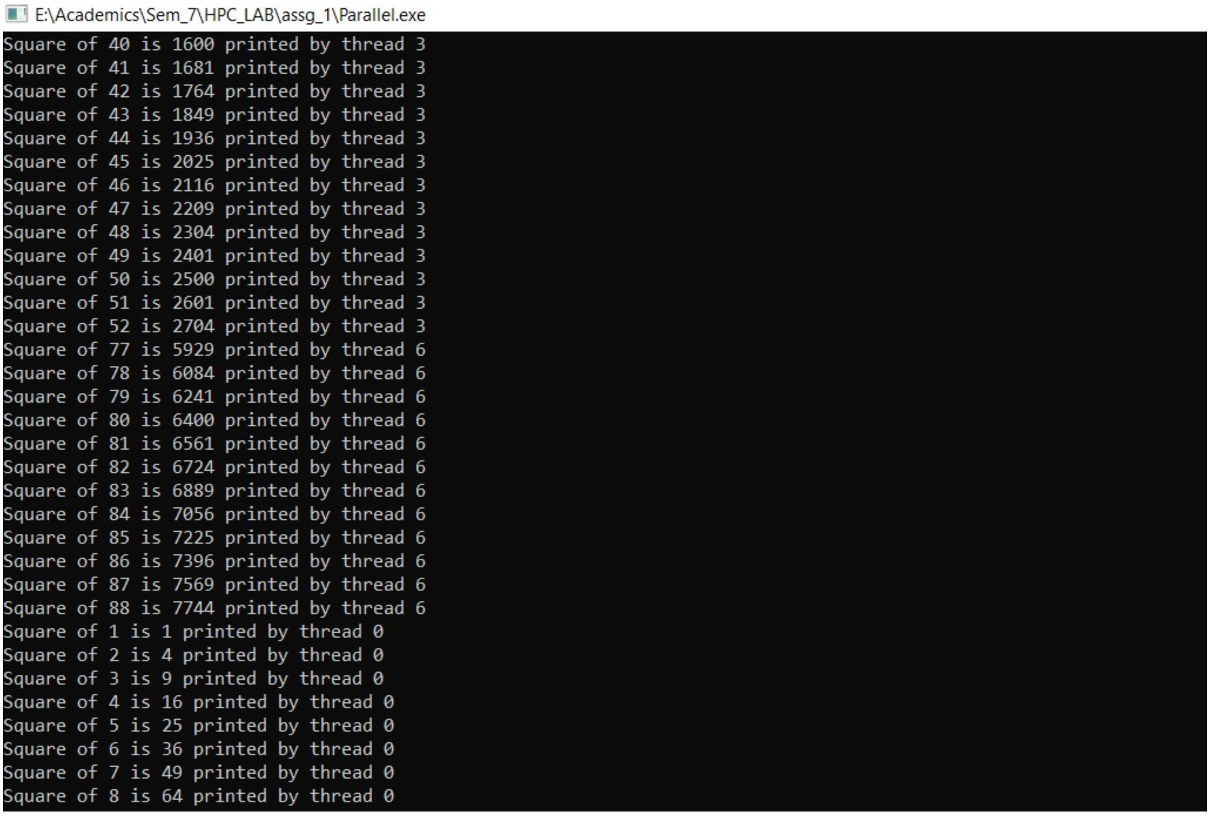
Execution Time : 0.010000 for Parallel
Execution Time : 0.004000 for Serial
This is due to fork and join

Q2. To add squares of numbers Program:



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

int main()
{
    int i,sum=0;
    int thread_sum[8];
    omp_set_num_threads(8);
    //can also be done by setting the envt var omp_num_threads to desired no of threads
    clock_t start, finish;
    start = clock();
    #pragma omp parallel
    {
        int ID = omp_get_thread_num();
        thread_sum[ID] = 0;
        #pragma omp for
        for(i=1;i<=100;i++)
        {
            printf ("Square of %d is %d printed by thread %d\n", i, (i*i),ID);
            thread_sum[ID] += (i*i);
        }
    }
    for(i=0;i<8;i++)
    {
        sum += thread_sum[i];
    }
    printf("Sum = %d",sum);
    finish = clock();
    double tt = (double)(finish-start)/CLOCKS_PER_SEC;
    printf("\nExec time is: %lf",tt);
    return 0;
}
```

Output:



E:\Academics\Sem_7\HPC_LAB\assg_1\Parallel.exe

```
Square of 40 is 1600 printed by thread 3
Square of 41 is 1681 printed by thread 3
Square of 42 is 1764 printed by thread 3
Square of 43 is 1849 printed by thread 3
Square of 44 is 1936 printed by thread 3
Square of 45 is 2025 printed by thread 3
Square of 46 is 2116 printed by thread 3
Square of 47 is 2209 printed by thread 3
Square of 48 is 2304 printed by thread 3
Square of 49 is 2401 printed by thread 3
Square of 50 is 2500 printed by thread 3
Square of 51 is 2601 printed by thread 3
Square of 52 is 2704 printed by thread 3
Square of 77 is 5929 printed by thread 6
Square of 78 is 6084 printed by thread 6
Square of 79 is 6241 printed by thread 6
Square of 80 is 6400 printed by thread 6
Square of 81 is 6561 printed by thread 6
Square of 82 is 6724 printed by thread 6
Square of 83 is 6889 printed by thread 6
Square of 84 is 7056 printed by thread 6
Square of 85 is 7225 printed by thread 6
Square of 86 is 7396 printed by thread 6
Square of 87 is 7569 printed by thread 6
Square of 88 is 7744 printed by thread 6
Square of 1 is 1 printed by thread 0
Square of 2 is 4 printed by thread 0
Square of 3 is 9 printed by thread 0
Square of 4 is 16 printed by thread 0
Square of 5 is 25 printed by thread 0
Square of 6 is 36 printed by thread 0
Square of 7 is 49 printed by thread 0
Square of 8 is 64 printed by thread 0
```

Github Link : https://github.com/kajalp23/HPC_Lab