**Class:** Final Year (Computer Science and Engineering)

**Year:** 2022-23 **Semester:**1

**Course:** High Performance Computing Lab

**Practical No. 1**

**Exam Seat No:**

2019BTECS00005 –Ashish Sutar

**Title of practical:**

Study and implementation of parallel for and reduction clause

**Problem Statement 1:**

Implement a parallel code for Hello World

**Screenshot 1:**

#include <stdio.h>

#include <omp.h>

int main()

{

    double start, end;

    start = omp\_get\_wtime();

#pragma omp parallel num\_threads(5)

    {

        printf("Hello World by Thread %d\n", omp\_get\_thread\_num());

    }

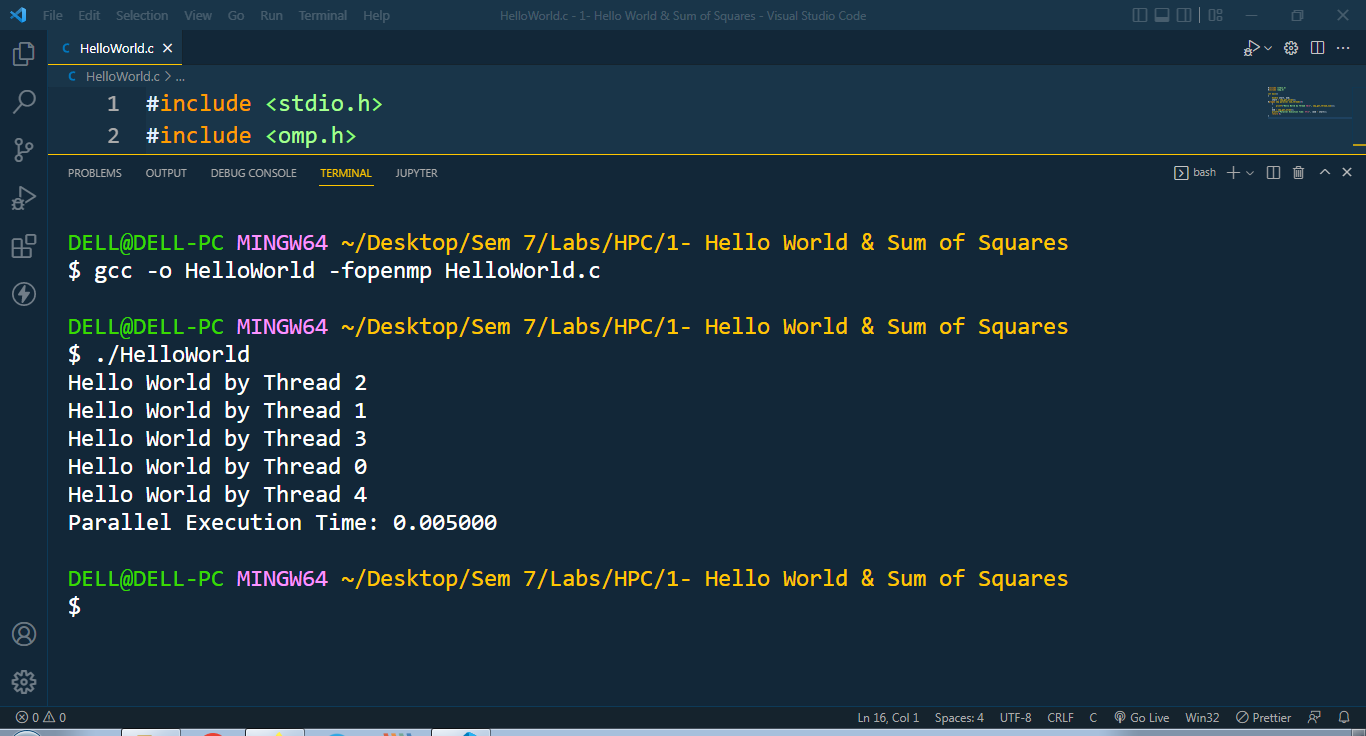
    end = omp\_get\_wtime();

    printf("Parallel Execution Time: %f\n", (end - start));

    return 0;

}

**Screenshot 2:**

****

**Information 1:**

OpenMP programs execute serially until they encounter the parallel directive. This directive is responsible for creating a group of threads. The exact number of threads can be specified in the directive, set using an environment variable, or at runtime using OpenMP functions. The main thread that encounters the parallel directive becomes the master of this group of threads and is assigned the thread id 0 within the group.

**Problem Statement 2:**

Implement a parallel code for Sum of Squares till 100

**Screenshot 3:**

#include <stdio.h>

#include <omp.h>

int main()

{

    double start, end;

    int sum = 0;

    start = omp\_get\_wtime();

#pragma omp parallel for reduction(+ \

                                   : sum) num\_threads(5)

    for (int i = 0; i < 100; i++)

    {

        sum += (i \* i);

    }

    end = omp\_get\_wtime();

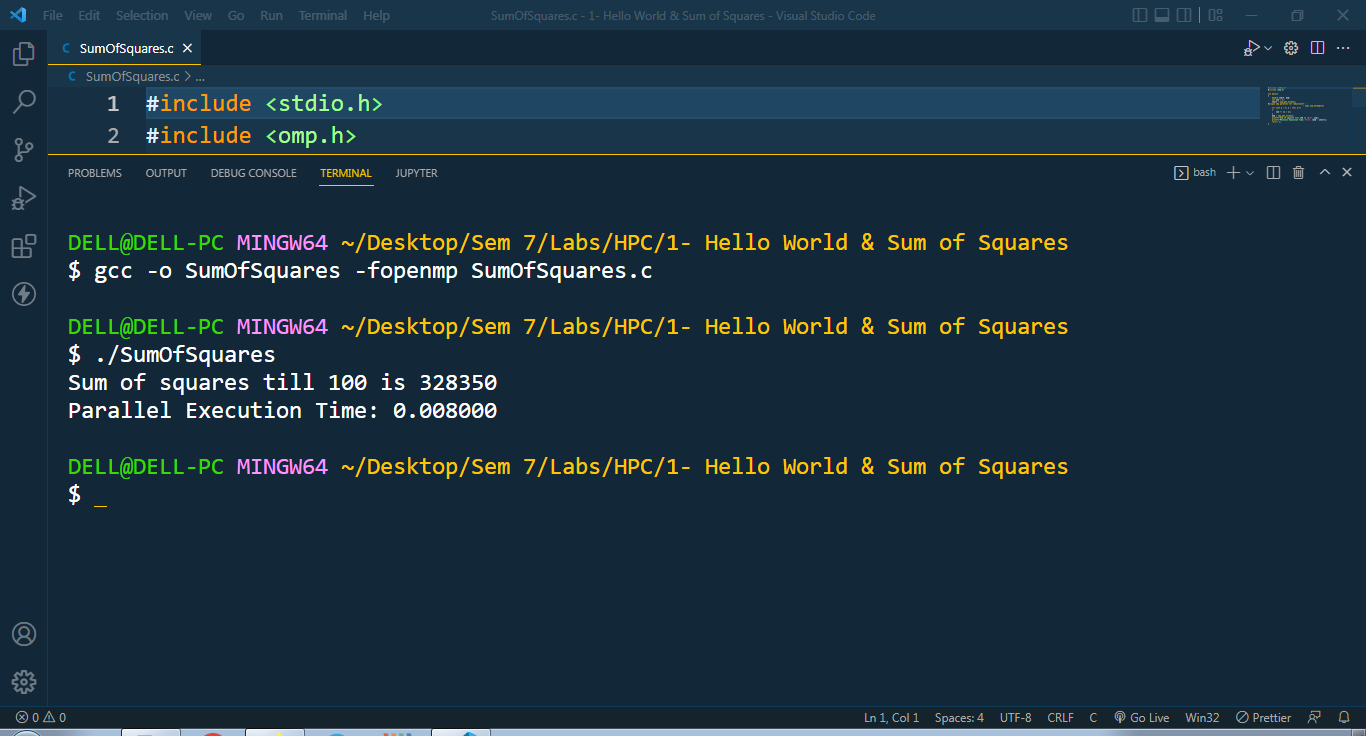
    printf("Sum of squares till 100 is %d\n", sum);

    printf("Parallel Execution Time: %f\n", (end - start));

    return 0;

}

**Screenshot 4:**

****

**Information 2:**

There reduction clause specifies how multiple local copies of a variable at different threads are combined into a single copy at the master when threads exit. The usage of the reduction clause is reduction (operator: variable list). This clause performs a reduction on the scalar variables specified in the list using the operator. The variables in the list are implicitly specified as being private to threads. The operator can be one of +, \*, -, &, |, ^, &&, and ||.

**Github Link:**