Course: High Performance Computing Lab

Practical No 1

PRN: 21510024

Name: Neha Kharat

Batch: B2

Title: Introduction to OpenMP

Problem Statement 1 – Demonstrate Installation and Running of OpenMP code in C

Recommended Linux based System:

Following steps are for windows:

OpenMP – Open Multi-Processing is an API that supports multi-platform shared-memory multiprocessing programming in C, C++ and Fortran on multiple OS. OpenMP uses a portable, scalable model that gives programmers a simple and flexible interface for developing parallel applications for platforms ranging from the standard desktop computer to the supercomputer.

To set up OpenMP,

We need to first install C, C++ compiler if not already done. This is possible through the MinGW Installer.  
Reference: Article on GCC and G++ installer ([Link](https://www.scaler.com/topics/c/c-compiler-for-windows/))

Note: Also install `mingw32-pthreads-w32` package.

Then, to run a program in OpenMP, we have to pass a flag `-fopenmp`.

Example:

To run a basic Hello World,

*#include* <stdio.h>

*#include* <omp.h>

*int* main(*void*)

{

*#pragma* *omp* *parallel*

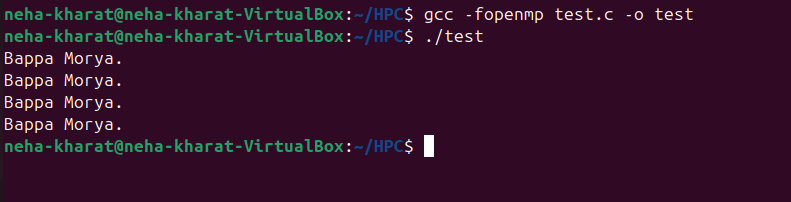
    printf("Hello, world.\n");

*return* 0;

}

gcc -fopenmp test.c -o test

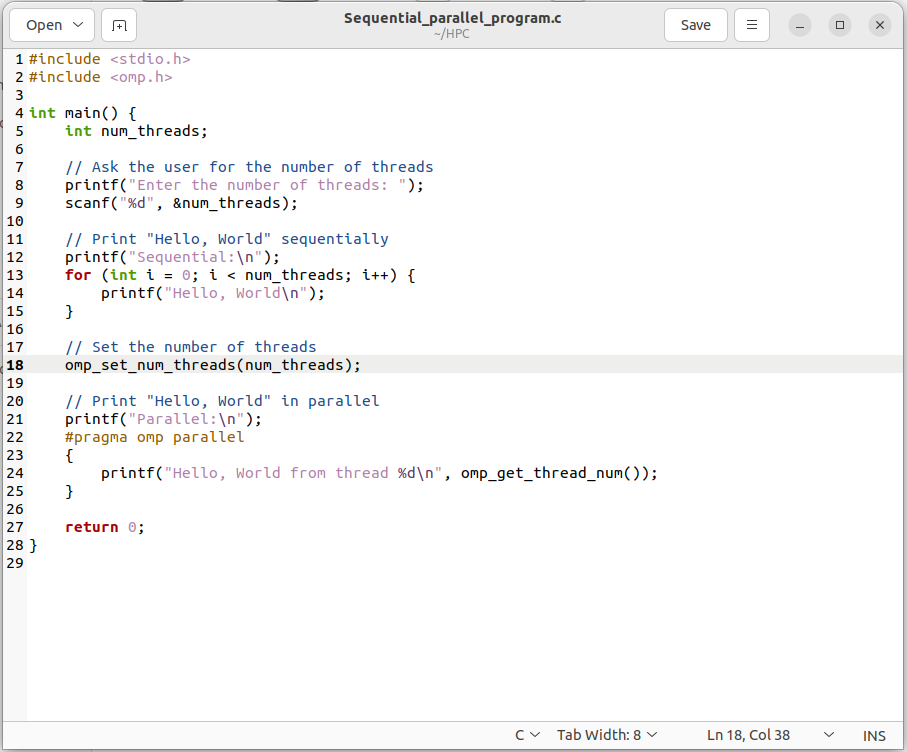
./test



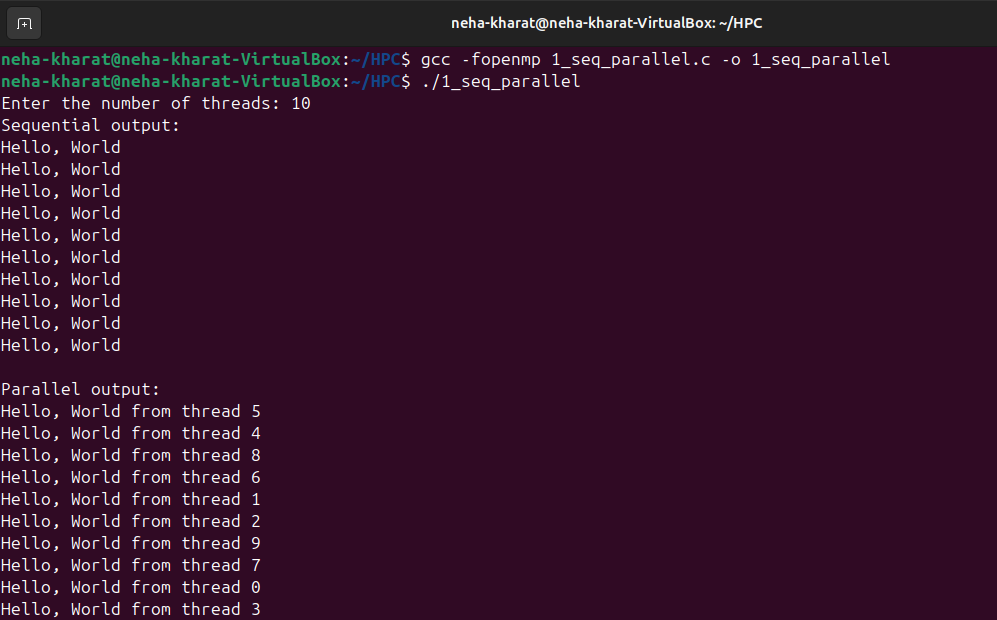
Problem Statement 2 – Print ‘Hello, World’ in Sequential and Parallel in OpenMP

We first ask the user for number of threads – OpenMP allows to set the threads at runtime. Then, we print the Hello, World in sequential – number of times of threads count and then run the code in parallel in each thread.

Code snapshot:



Output snapshot:



Analysis:

GitHub Link: make a public repository upload code of an assignment and paste its link here.

Problem statement 3: Calculate theoretical FLOPS of your system on which you are running the above codes.

Elaborate the parameters and show calculation.

Parameters:

**Base clock speed:** 3.00 GHz

**Number of cores:** 4

**Number of threads per core:** 2 (Hyper-Threading enabled)

**Number of floating-point operations per cycle:** 16 (assuming AVX-512)

Calculation:

Step1: 3.00 GHz = 3.00 × 10^9 Hz

Step2:

Assuming 16 floating-point operations per cycle:

FLOPS per core=Clock speed (Hz)×Floating-point operations per cycle

FLOPS per core=3.00×109Hz×16=48×109FLOPS=48GFLOPS

Step3:

Total FLOPS=Number of cores×Number of threads per core×FLOPS per core

Total FLOPS=4cores×2threads per core×48GFLOPS=384GFLOPS