Course: High Performance Computing Lab

Practical No 1

PRN: 22510107

Name: Akshay Wale

Batch: B4

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,

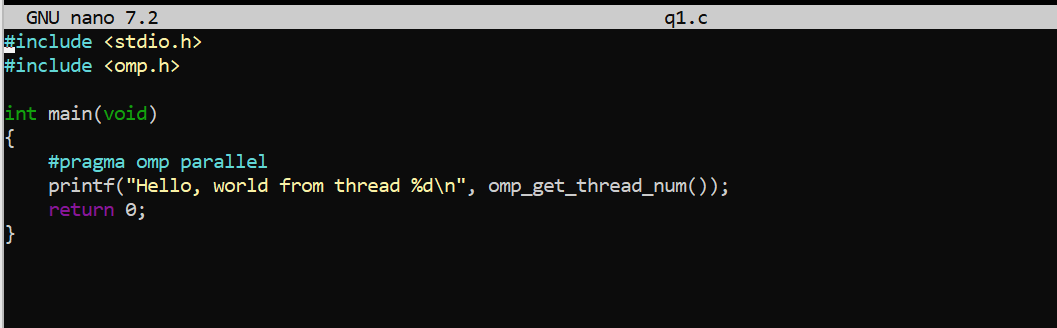
We need to first install C, C++ compiler if not already done. This is possible through the MinGW Installer.  
Reference: Article on GCC and G++ installer ([Link](https://www.scaler.com/topics/c/c-compiler-for-windows/))

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

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

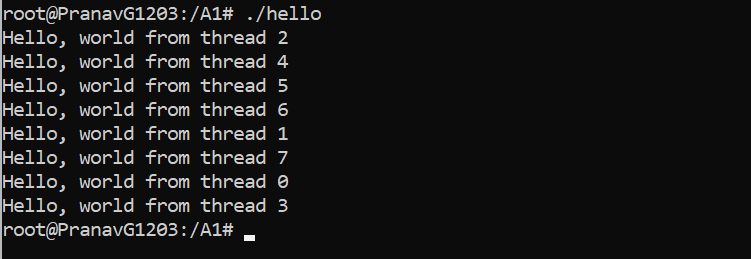
Example:

To run a basic Hello World,



gcc -fopenmp q1.c -o hello

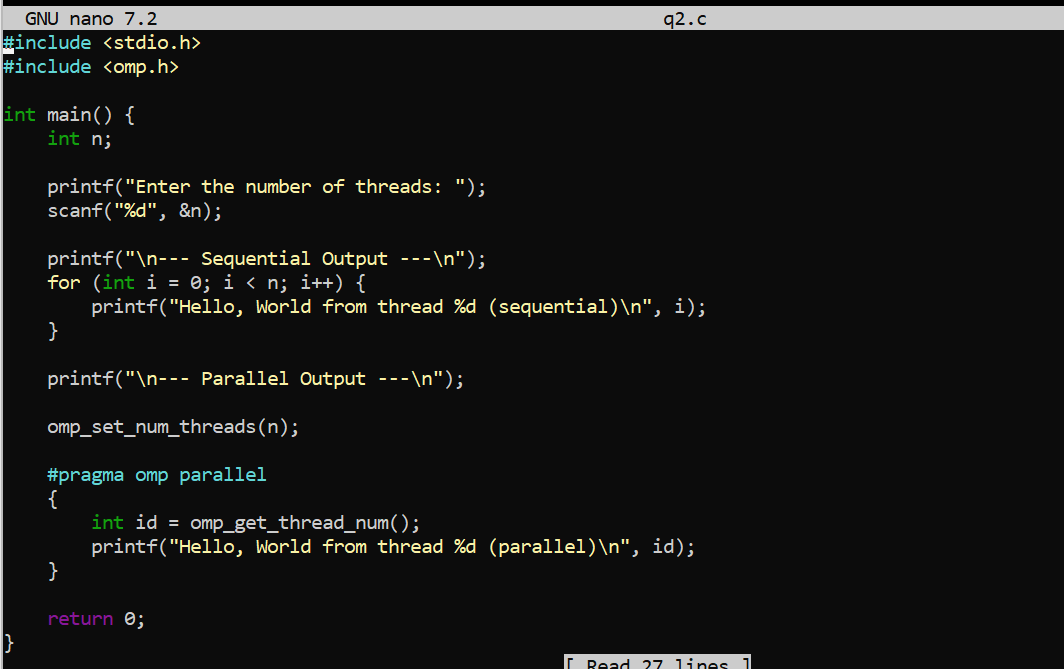
.\hello



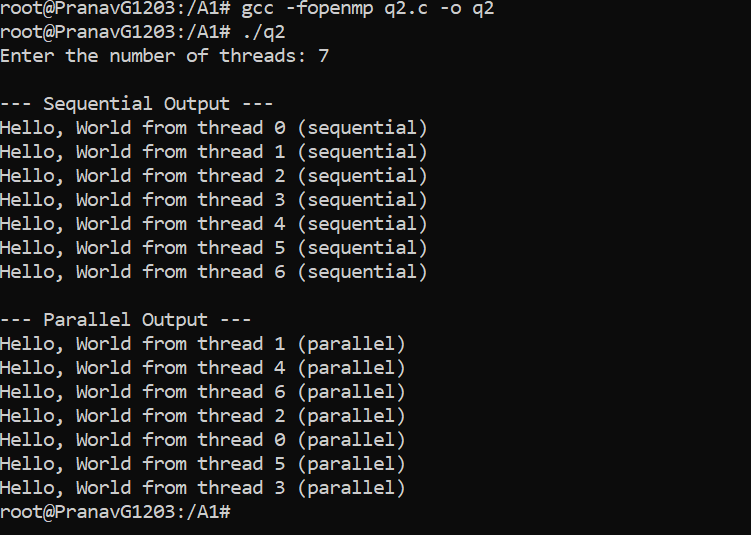
Problem Statement 2 – Print ‘Hello, World’ in Sequential and Parallel in OpenMP

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:

* Sequential output executes in the main thread, one after the other.
* Parallel output is executed by multiple threads concurrently.
* The number of threads can be set at runtime using omp\_set\_num\_threads(n).

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

Link: https:

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

**Definition:**

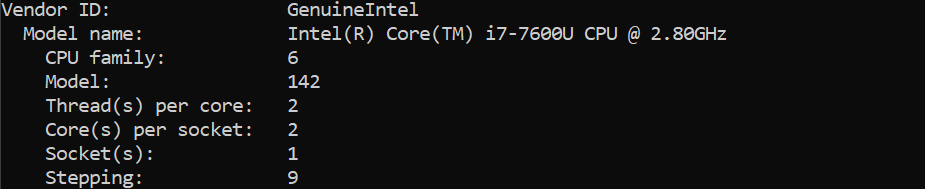
**FLOPS (Floating Point Operations Per Second)** is a measure of computational performance — especially in tasks involving scientific or parallel computation. This helps estimate how much raw number-crunching your system is capable of.

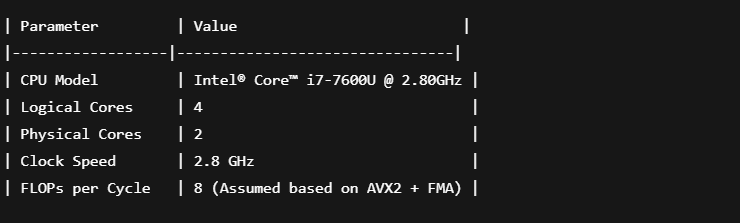
**Theoretical FLOPS Formula**

Theoretical FLOPS= Number of Cores **×** Clock Speed (Hz) **×** FLOPs Per Cycle

Where:

* **Cores** = Number of physical or logical CPU cores.
* **Clock Speed** = CPU frequency (in Hz).
* **FLOPs per cycle** = Depends on architecture — for modern CPUs, it can be **4 to 16** due to vectorization (SSE, AVX, etc.).





Assuming:

* Cores = **2 physical cores**
* Clock speed = **2.8 GHz = 2.8 × 10⁹ Hz**
* FLOPs per cycle = **8**

FLOPS = 2 × 2.8 × 109 × 8 = **44.8 GFLOPS**

Elaborate the parameters and show calculation.