AY 2025-26 Odd Sem

Course: Cutting Edge Technologies Lab

Course Code: 7CS352

Practical No 1

PRN:23510030

Name:Harshvardhan Mane

Batch:T1

Title: Introduction to parallel programming, OpenMP installation, simple “Hello World” programs.

Problem Statement 1: Illustrate different types of Parallel Programming Models(Shared memory, distributed memory, accelerated computing)

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 hello

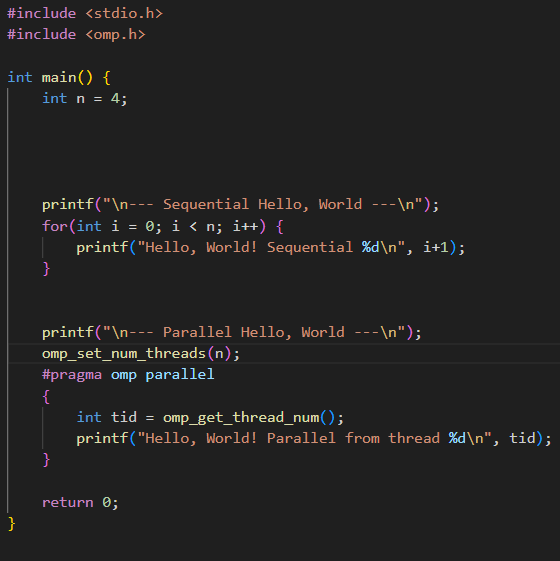
.\hello.exe



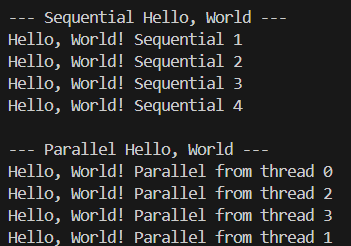
Problem Statement 2 – Print ‘Hello, World’ in Sequential and Parallel in OpenMP

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: Executes in single thread, one print at a time.

Parallel: Multiple threads execute simultaneously, each thread prints independently.

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.

FLOPS=Cores×Clock Speed×Vector Width×FLOPs/cycle

Elaborate the parameters and show calculation.

**Cores** = 10

**Clock speed** = 1.3 GHz = 1.3 × 10⁹ cycles/sec

**Vector width** = supports **AVX2 (256-bit registers)**

* 256 bits = 8 single-precision floats (32-bit) OR 4 double-precision (64-bit)

**FLOPs per cycle** = 2 (because Intel CPUs support FMA: fused multiply + add = 2 operations per cycle)

**Calculation:**

1. For **single precision (32-bit float):**

=10×(1.3×109)×8×2=208GFLOPS

1. For **double precision (64-bit float):**

=10×(1.3×109)×4×2=104GFLOPS