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

**Year:** 2024-25 **Semester:** 1

**Course:** High Performance Computing Lab

**Practical No. 3**

**Exam Seat No:**

**Title of practical:**

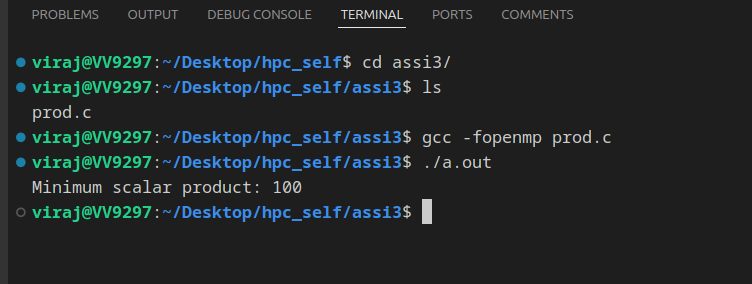
Study and Implementation of schedule, nowait, reduction, ordered and collapse clauses

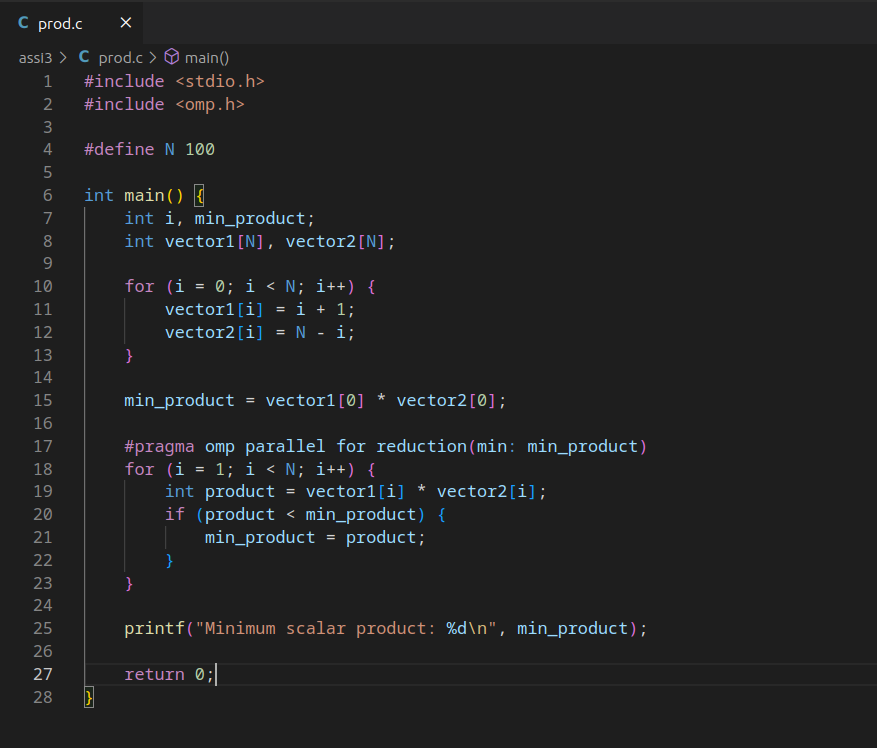
**Problem Statement 1:**

Analyse and implement a Parallel code for below program using OpenMP.

// C Program to find the minimum scalar product of two vectors (dot product)

**Screenshots:**

****

****

**Information and analysis:**

**This C program initializes two vectors and calculates their minimum scalar product using OpenMP for parallel processing. The vectors are initialized such that vector1 contains values from 1 to 100 and vector2 contains values from 100 to 1. The #pragma omp parallel for reduction(min: min\_product) directive ensures that the minimum product is computed in parallel, improving performance on multi-core systems.**

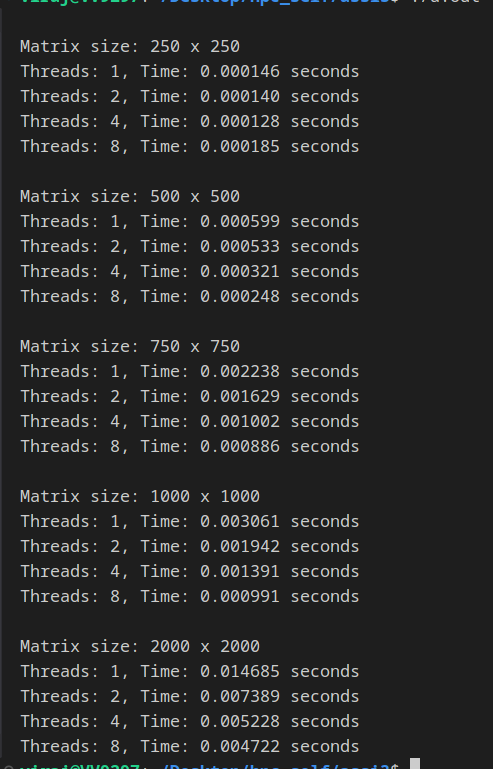
**Problem Statement 2:**

Write OpenMP code for two 2D Matrix addition, vary the size of your matrices from 250, 500, 750, 1000, and 2000 and measure the runtime with one thread (Use functions in C in calculate the execution time or use GPROF)

i. For each matrix size, change the number of threads from 2,4,8., and plot the speedup versus the number of threads.

ii. Explain whether or not the scaling behaviour is as expected.

**Screenshots:**

****

**Information and analysis:**

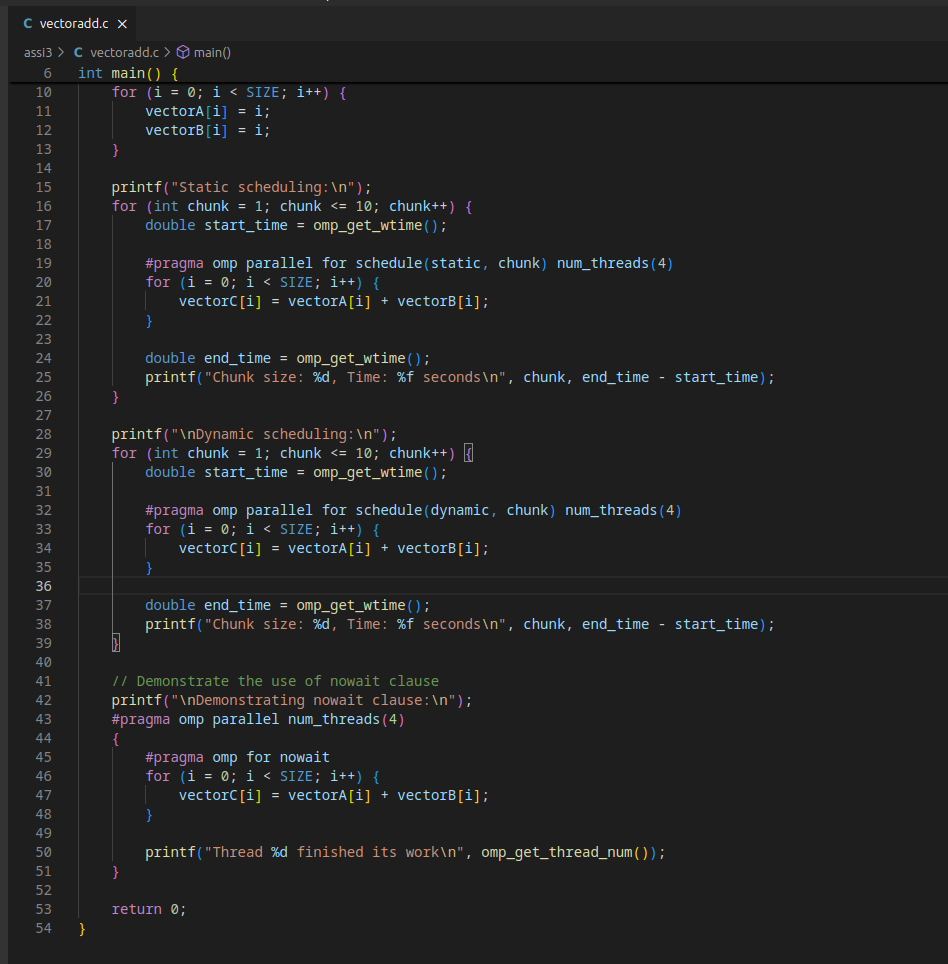
**the speedup increases with the number of threads doing the job.**

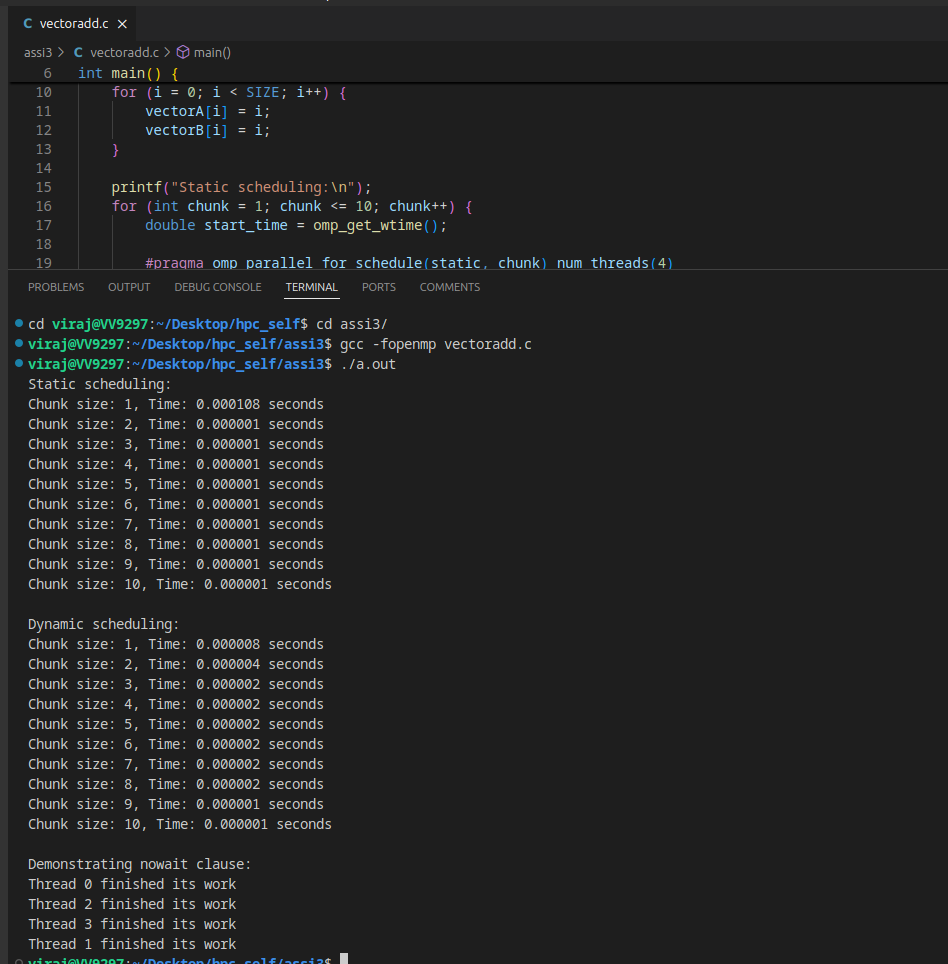
**The time required to complete the execution decreases but in case of the 1st one the time for 8 threads in more than 4 threads because of excessive overhead of interprocess communication**

**Problem Statement 3:**

For 1D Vector (size=200) and scalar addition, Write a OpenMP code with the following: i. Use STATIC schedule and set the loop iteration chunk size to various sizes when changing the size of your matrix. Analyze the speedup. ii. Use DYNAMIC schedule and set the loop iteration chunk size to various sizes when changing the size of your matrix. Analyze the speedup. iii. Demonstrate the use of nowait clause.

**Screenshots:**

****

****

**Information and analysis:**

**This C program uses OpenMP to parallelize the addition of two vectors, vectorA and vectorB, storing the result in vectorC. It demonstrates both static and dynamic scheduling with varying chunk sizes, measuring and printing the execution time for each configuration. The #pragma omp parallel for directive is used to parallelize the loop, and the nowait clause is introduced to show its effect on synchronization. The program aims to compare the performance impact of different scheduling strategies and chunk sizes on parallel vector addition.**

**Github Link:**