MIT-WPU

॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥

Dr. Vishwanath Karad

MIT WORLD PEACE UNIVERSITY | PUNE

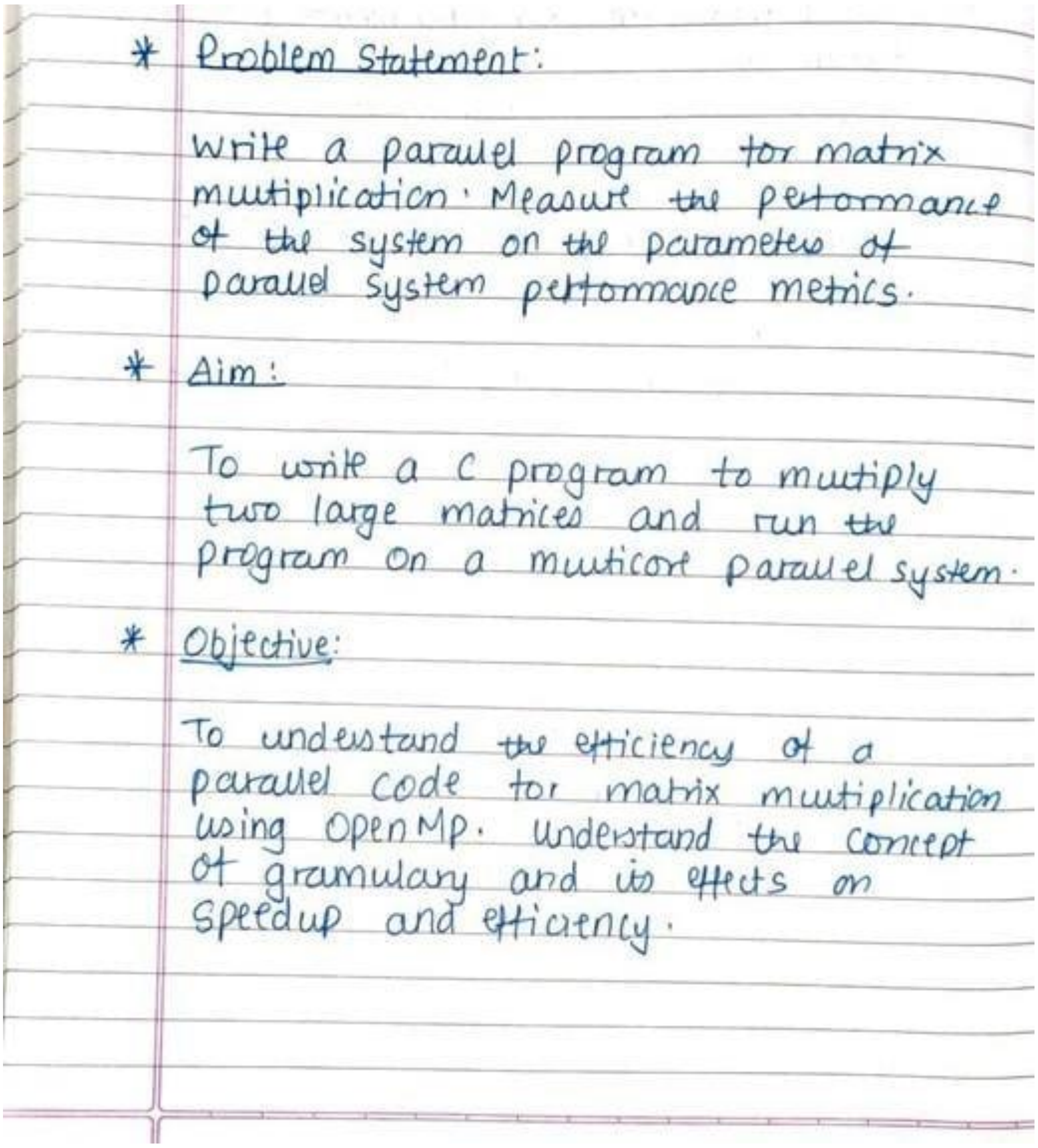
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

PP LAB-1

Name: Raghav Maheshwari

Roll: PA53

Panel: A



* Problem Statement:

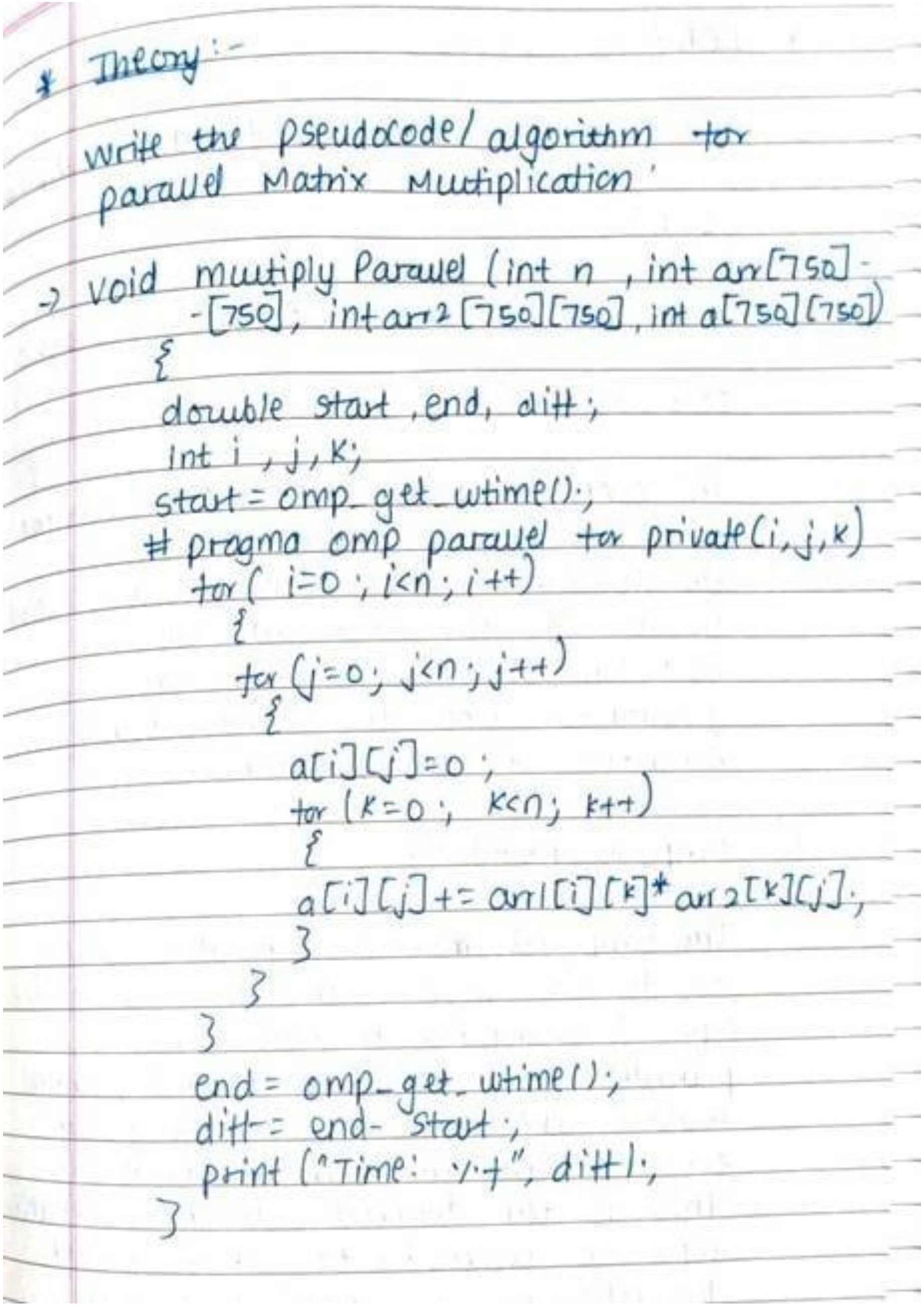
Write a parallel program for matrix multiplication. Measure the performance of the system on the parameters of parallel system performance metrics.

* Aim:

To write a C program to multiply two large matrices and run the program on a multicore parallel system.

* Objective:

To understand the efficiency of a parallel code for matrix multiplication using OpenMP. Understand the concept of granulary and its effects on speedup and efficiency.



* Theory :-

Write the pseudocode/ algorithm for parallel Matrix Multiplication'

```
-> void multiply Parallel (int n , int arr[750]-
        -[750], int arr2 [750][750], int a[750][750])
   {
     double start , end, diff ;
     int i , j , k;
     start = omp_get_wtime();
    # pragma omp parallel for private(i, j, k)
      for ( i=0 ; i<n ; i++)
        {
         for (j=0; j<n ; j++)
          {
            a[i][j]=0 ;
            for (k=0 ; k<n; k++)
            {
            a[i][j] += arr1[i][k]* arr2[k][j];
            }
          }
        }
      end = omp_get_wtime();
      diff = end - start ;
      print ("Time: %f", diff);
   }
```

* FAQ's:-

Q1) What is the complexity of Matrix Mutiplica

-> $O(n^3)$

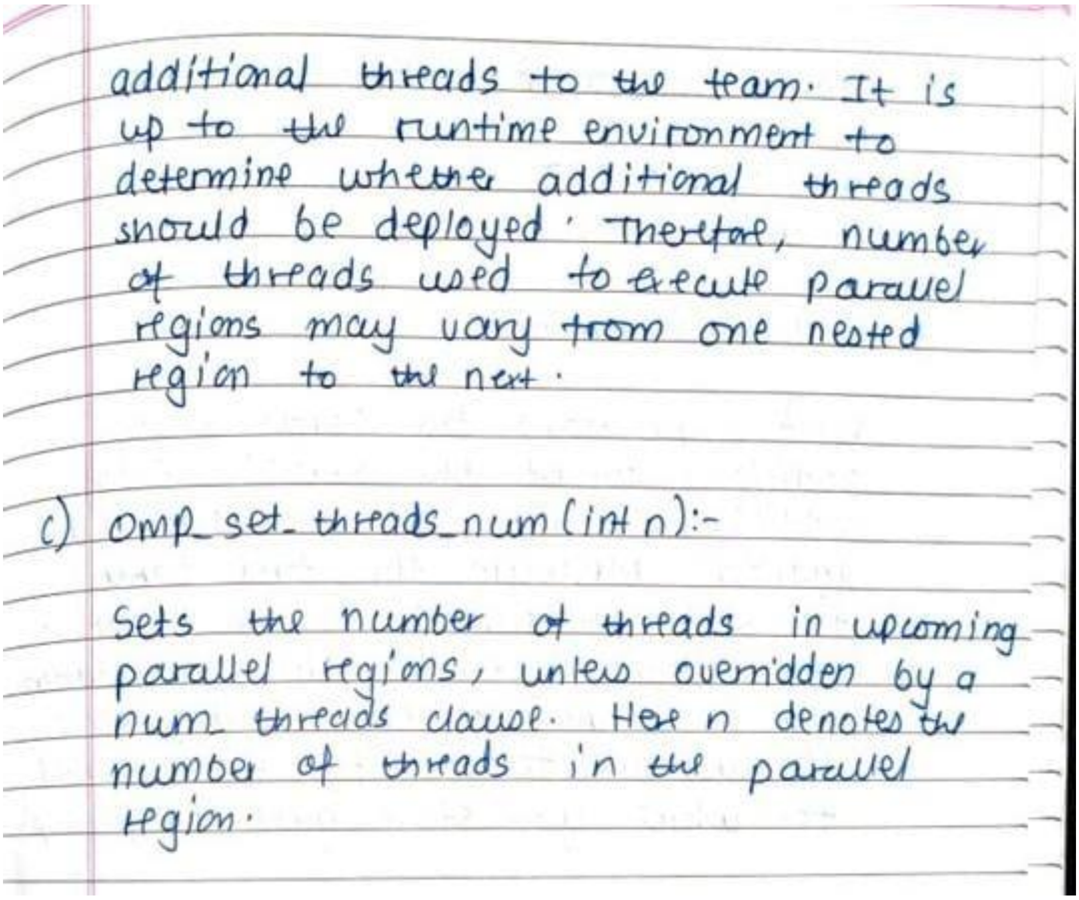
Q2) Give uses of the following functions:-

a) omp_get_wtime():-

The omp_get_wtime() function returns a double precision value equal to the number of seconds since the initial value of the operating system real-time clock. The initial value is guaranted not to change during execution of the program.

b) omp_set_nested(1):-

The omp_set_nested Subroutine enables or disables nested parallelism. If enable expr is evaluated to False, nested parallelism is disabled. Nested parallel regions are serolised, and they are executed by the current thread. This is the default setting. If enable expr is evaluated to TRUE, nested parallelism is enabled. Parallel regions that are nested can deploy



additional threads to the team. It is up to the runtime environment to determine whether additional threads should be deployed. Therefore, number of threads used to execute parallel regions may vary from one nested region to the next.

c) omp_set_threads_num(int n):-

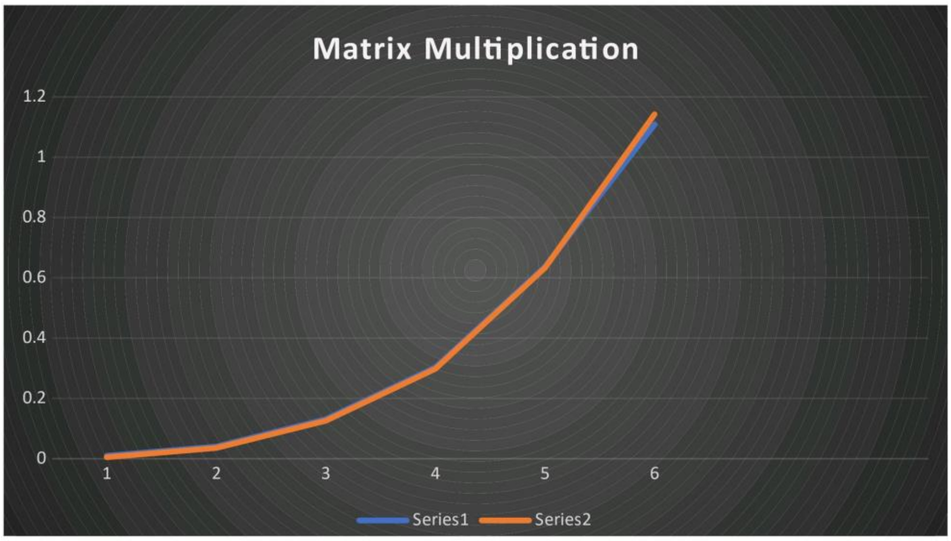
Sets the number of threads in upcoming parallel regions, unless overridden by a num_threads clause. Here n denotes the number of threads in the parallel region.

CODE

```c
#include <stdio.h>
#include<stdlib.h>
#include<omp.h>

int main(void)
{
    int i,j,k,n;
    printf("\nEnter number of rows and columns: ");
    scanf("%d", &n);

    int arr[n][n];
    int brr[n][n];

    for( i = 0; i<n; i++)
    {
        for( j = 0; j<n; j++)
        {
            arr[i][j] = rand() % 10;
            brr[i][j] = rand() % 10;
        }
    }
    int crr[n][n];
    double t_start,t_end,temp,temp1;
    t_start = omp_get_wtime();
    for( i=0; i<n; i++)
    {
        for( j=0; j<n; j++)
        {
            crr[i][j] = 0;
    for( k=0; k<n; k++)
    {
      crr[i][j] += arr[i][k] * brr[k][j];
    }
  }
}

t_end = omp_get_wtime();
temp = t_end - t_start;
printf("\nTime required for series multiplication: %f", temp);
t_start = omp_get_wtime();
#pragma omp parallel for private(i,j,k)
for( i=0; i<n; i++)
{
  for( j=0; j<n; j++)
  {
    crr[i][j] = 0;
    for( k=0; k<n; k++)
    {
      crr[i][j] += arr[i][k] * brr[k][j];
    }
  }
}

t_end = omp_get_wtime();
temp1 = t_end - t_start;

printf("\nTime required for parallel multiplication: %f", temp1);
```

Output:

Size of matrix	Series Time	Parallel Time
100 x 100	0.00945963	0.00479173
200 x 200	0.0399801	0.0357836
300 x 300	0.13124	0.125959
400 x 400	0.303301	0.298397
500 x 500	0.63737	0.633556
700 x 700	1.10831	1.14227



Input: Unsorted array of data points/values.

Output: Sorted Array of data points/values.

Platform: Ubuntu (give latest version) or Windows

Conclusion: Thus, successfully studied, analyzed serial to parallel conversion.