Type your text

Name: Anshul Agrawal

USN: 1RV17CS021

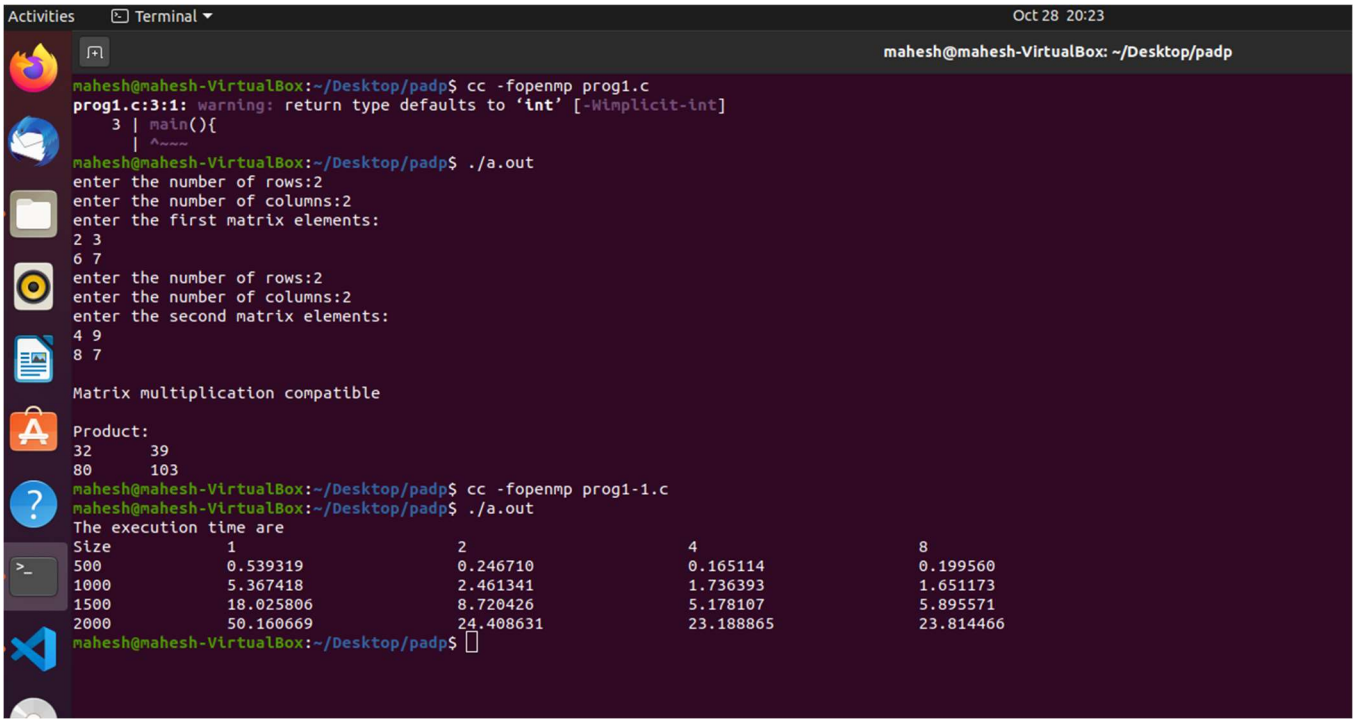
PADP lab Program 1

Code:

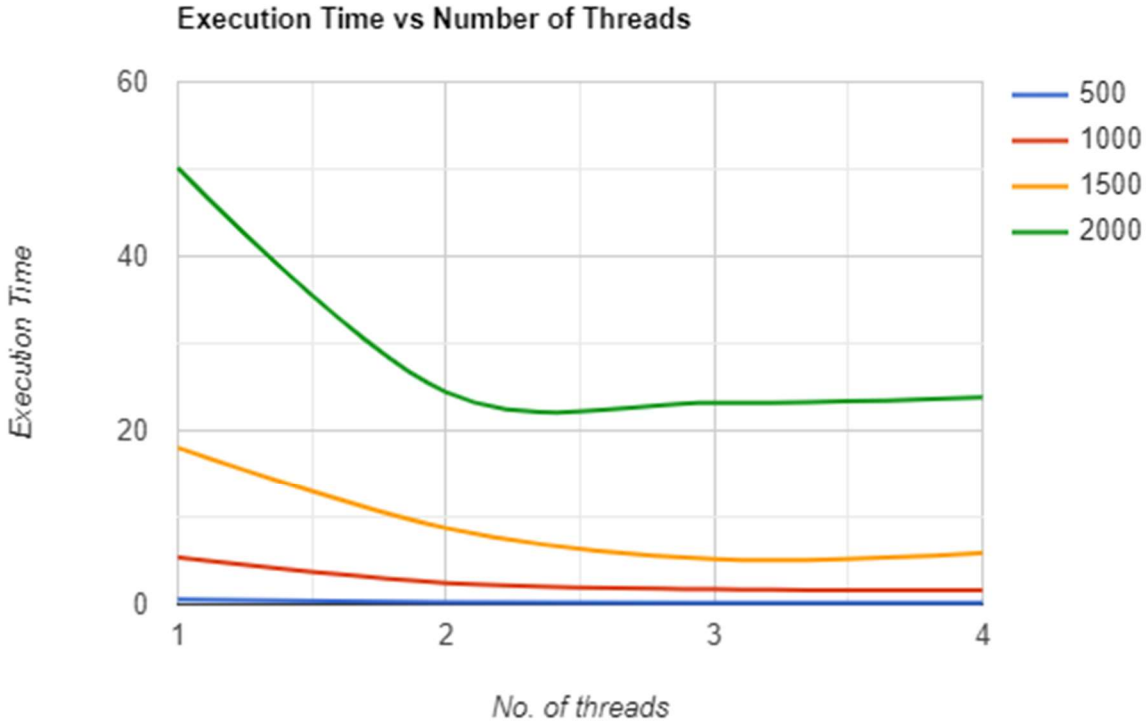
```
#include<omp.h>
#include<stdio.h>
#include<stdlib.h>
int main(){
        int it=1;
        printf("The execution time are\nSize\t\t1\t\t2\t\t4\t\t8\n");
        while(it<=4){
                int r = 500*it, c = 500*it, i, j, sum =0, k;
                //dynamically allocate arrays
                int **arr1 = (int **)malloc(r * sizeof(int *));
                for (i=0; i<r; i++)
                arr1[i] = (int *)malloc(c * sizeof(int));
                int **arr2 = (int **)malloc(r * sizeof(int *));
                for (i=0; i<r; i++)
                arr2[i] = (int *)malloc(c * sizeof(int));
                int **arr3 = (int **)malloc(r * sizeof(int *));
                for (i=0; i<r; i++)
                arr3[i] = (int *)malloc(c * sizeof(int));

                for(i = 0;i < r; i++)
                for(j = 0;j < c; j++)
                arr1[i][j] = rand()/r;

                for(i = 0;i < r; i++)
                for(j = 0;j < c; j++)
                arr2[i][j] = rand()/r;
```

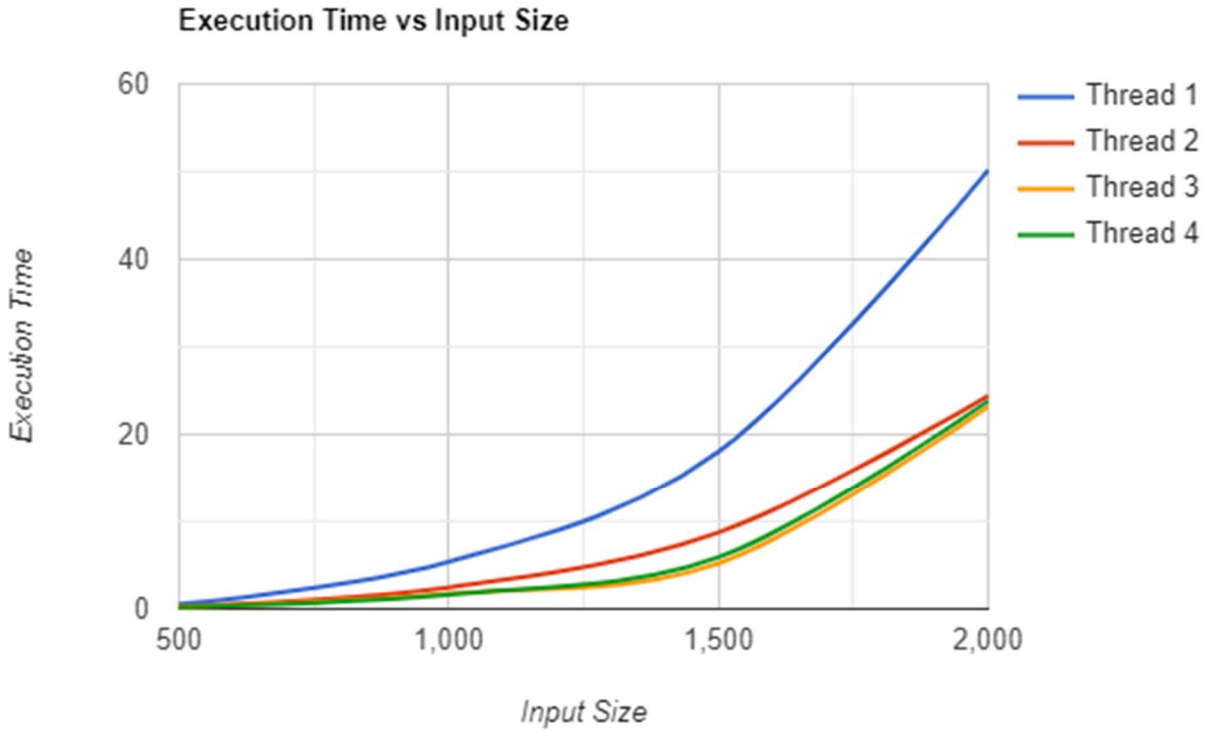
```
                double x = omp_get_wtime();
                for(i = 0;i < r; i++)
                for(j = 0;j < c; j++)
                for(k = 0;k < r; k++)
                arr3[i][j] += arr1[i][k] * arr2[k][j];
                double y = omp_get_wtime();

                printf("%d\t\t",r);
                printf("%lf\t\t", y-x);
                for(int p=2;p<=8;p=p*2)
                {
                        double x = omp_get_wtime();
                        omp_set_num_threads(p);
                        #pragma omp parallel for private(j, k)
                        for(i = 0;i < r; i++)
                        for(j = 0;j < c; j++)
                        {
                                arr3[i][j]=0;
                                for(k = 0;k < r; k++)
                                        arr3[i][j] += arr1[i][k] * arr2[k][j];
                        }
                        double y = omp_get_wtime();
                        printf("%lf\t\t", y-x);
                }
                printf("\n");
                it++;
        }
        return 0;
}
```

Output:


Activities
Terminal
Oct 28 20:23
mahesh@mahesh-VirtualBox: ~/Desktop/padp
mahesh@mahesh-VirtualBox:~/Desktop/padp$ cc -fopenmp prog1.c
prog1.c:3:1: warning: return type defaults to 'int' [-Wimplicit-int]
3 | main(){
| ^~~~
mahesh@mahesh-VirtualBox:~/Desktop/padp$ ./a.out
enter the number of rows:2
enter the number of columns:2
enter the first matrix elements:
2 3
6 7
enter the number of rows:2
enter the number of columns:2
enter the second matrix elements:
4 9
8 7
Matrix multiplication compatible
Product:
32 39
80 103
mahesh@mahesh-VirtualBox:~/Desktop/padp$ cc -fopenmp prog1-1.c
mahesh@mahesh-VirtualBox:~/Desktop/padp$ ./a.out
The execution time are
Size 1 2 4 8
500 0.539319 0.246710 0.165114 0.199560
1000 5.367418 2.461341 1.736393 1.651173
1500 18.025806 8.720426 5.178107 5.895571
2000 50.160669 24.408631 23.188865 23.814466
mahesh@mahesh-VirtualBox:~/Desktop/padp$

Graphs:


Execution Time vs Number of Threads
60
40
20
0
1
2
3
4
Execution Time
No. of threads
500
1000
1500
2000


Execution Time vs Input Size
60
40
20
0
500
1,000
1,500
2,000
Execution Time
Input Size
Thread 1
Thread 2
Thread 3
Thread 4