

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA SURATHKAL
DEPARTMENT OF INFORMATION TECHNOLOGY

IT 301 Parallel Computing LAB 4

02nd September 2020

Faculty: Dr. Geetha V and Mrs. Tanmayee

Name: Bhagyashri Nilesh Bhmare

Roll No.: 181IT111

Execute following programs and put screen shots of the output. Write analysis of the result before uploading in IRIS as a single pdf file. for programming exercises, write the code and also put screenshot of the results.

1. Program 1

Execute following code and observe the working of task directive. Check the result by removing if() clause with task.

```
#include<stdio.h>
#include<omp.h>
int fibo(int n); int
main(void)
{ int n,fib;
double t1,t2;
printf("Enter the value of n:\n");
scanf("%d",&n);
t1=omp_get_wtime(); #pragma
omp parallel shared(n)
{
#pragma omp single
{
fib=fibo(n);
}
}
t2=omp_get_wtime(); printf("Fib is
%d\n",fib); printf("Time taken is %f
s \n",t2-t1); return 0; }
int fibo(int n)
{ int
a,b;
if(n<2)
```

```

return
n; else
{
#pragma omp task shared(a) if(n>5)
{
printf("Task Created by Thread %d\n",omp_get_thread_num()); a=fibo(n-1);
printf("Task Executed by Thread %d \ta=%d\n",omp_get_thread_num(),a); }
#pragma omp task shared(b) if(n>5)
{
printf("Task Created by Thread %d\n",omp_get_thread_num()); b=fibo(n-2);
printf("Task Executed by Thread %d \tb=%d\n",omp_get_thread_num(),b); }
#pragma omp taskwait
return a+b;
}

```

With if() clause:

```

Time taken for sequential execution is 0.000002 s
Time taken for parallel execution is 0.013819 s
pryanka@pryanka: ~/Desktop/PC/Lab4$ gcc -o pi -fopenmp program1.c
pryanka@pryanka: ~/Desktop/PC/Lab4$ ./pi
Enter the value of n:
8
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Executed by Thread 1 a=1
Task Created by Thread 1
Task Executed by Thread 1 b=0
Task Executed by Thread 1 a=1
Task Created by Thread 1
Task Executed by Thread 1 b=1
Task Executed by Thread 1 a=2
Task Created by Thread 0
Task Created by Thread 0
Task Created by Thread 0
Task Executed by Thread 0 a=1
Task Created by Thread 0
Task Executed by Thread 0 b=0
Task Executed by Thread 0 a=1
Task Created by Thread 0
Task Executed by Thread 0 b=1
Task Executed by Thread 0 a=2
Task Created by Thread 0
Task Executed by Thread 0 a=1
Task Created by Thread 0
Task Executed by Thread 0 b=0
Task Executed by Thread 0 b=1
Task Executed by Thread 0 b=3
Task Created by Thread 1
Task Executed by Thread 1 a=1
Task Created by Thread 1
Task Executed by Thread 1 b=0
Task Executed by Thread 1 b=1
Task Executed by Thread 1 a=3
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Executed by Thread 1 a=1
Task Created by Thread 1
Task Executed by Thread 1 b=0
Task Executed by Thread 1 a=1
Task Created by Thread 1
Task Executed by Thread 1 b=1
Task Executed by Thread 1 b=2
Task Executed by Thread 1 a=5
Fib is 8
Time taken is 0.012136 s
pryanka@pryanka: ~/Desktop/PC/Lab4$

```

Without if() clause:

```

Terminal
Sep 6 21:22
priyanka@priyanka: ~/Desktop/PC/Lab4

Time consumed by Thread 1 a=5
Fib is 8
Time taken is 0.012136 s
priyanka@priyanka:~/Desktop/PC/Lab4$ gcc -o p1 -fopenmp program1.c
priyanka@priyanka:~/Desktop/PC/Lab4$ ./p1
Enter the value of n:
5
Task Created by Thread 0
Task Created by Thread 3
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Executed by Thread 1 b=0
Task Created by Thread 1
Task Executed by Thread 1 a=1
Task Executed by Thread 1 b=1
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Created by Thread 1
Task Executed by Thread 1 b=0
Task Created by Thread 1
Task Executed by Thread 1 a=1
Task Executed by Thread 1 a=1
Task Executed by Thread 1 a=2
Task Created by Thread 3
Task Created by Thread 3
Task Executed by Thread 3 b=0
Task Created by Thread 3
Task Executed by Thread 3 a=1
Task Executed by Thread 3 b=1
Task Created by Thread 3
Task Executed by Thread 3 b=1
Task Created by Thread 3
Task Executed by Thread 3 b=0
Task Created by Thread 3
Task Executed by Thread 3 a=1
Task Executed by Thread 3 a=2
Task Executed by Thread 3 a=3
Task Executed by Thread 1 b=3
Task Created by Thread 2
Task Created by Thread 2
Task Executed by Thread 2 b=1
Task Created by Thread 2
Task Created by Thread 2
Task Executed by Thread 2 b=0
Task Created by Thread 2
Task Executed by Thread 2 a=1
Task Executed by Thread 2 a=1
Task Executed by Thread 2 b=2
Task Executed by Thread 0 a=5
Fib is 8
Time taken is 0.013073 s
priyanka@priyanka:~/Desktop/PC/Lab4$

```

Task Scheduling if done for all values of n when if clause is not used, while when if clause is used then it is not done for all values of n

Programming exercises in OpenMP

2. Write a C/C++ OpenMP program to find ROWSUM and COLUMNSUM of a matrix $a[n][n]$. Compare the time of parallel execution with sequential execution.

Program:

```

#include <stdio.h>
#include <omp.h>
#include <stdlib.h> #include
<time.h>
int main(void)
{
    int n, i, j;
    double t1_s, t2_s, t1_p, t2_p;
    printf("Enter the value of n : ");
    scanf("%u", &n);
    int m[n][n];    int
    row_sum[n];    int
    column_sum[n];
    srand(time(0));

```

```

printf("Matrix\n");
for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        int k = (rand()) % 10000;
        m[i][j] = k;
    }
    printf("%d ", k);
    printf("\n");
}

//sequential    t1_s =
omp_get_wtime();    for (i
= 0; i < n; i++)
{
    int temp = 0;
for (j = 0; j < n; j++)
temp += m[i][j];
    row_sum[i] = temp;
}
for (i = 0; i < n; i++)
{
    int temp = 0;    for (j
= 0; j < n; j++)    temp
+=    m[j][i];
column_sum[i] = temp;
}
t2_s = omp_get_wtime();
int temp;

t1_p = omp_get_wtime(); #pragma
omp parallel shared(n)
{
#pragma omp for schedule(static, 10) private(i, j, temp)
for (i = 0; i < n; i++)
{
    temp = 0;
for (j = 0; j < n; j++)
temp += m[i][j];
    row_sum[i] = temp;
}
#pragma omp for schedule(static, 10) private(i, j, temp)
for (i = 0; i < n; i++)
{
    temp = 0;
for (j = 0; j < n; j++)

```

```

temp      +=      m[j][i];
column_sum[i] = temp;
    }
}
t2_p = omp_get_wtime();
//results
printf("Row Sum : \n");
for (i = 0; i < n; i++)
{
    printf("Row %d: %d \n", i, row_sum[i]);
}
printf("\nColumn Sum : \n");
for (i = 0; i < n; i++)
{
    printf("Column %d : %d \n", i, column_sum[i]);
}
printf("\nTime taken for sequential execution is %f s \n", t2_s - t1_s);
printf("\nTime taken for parallel execution is %f s \n", t2_p - t1_p);    return
0;
}

```

```

Terminal
priyanka@priyanka:~/Desktop/PC/Lab4$ gcc -o p2 -fopenmp program2.c
priyanka@priyanka:~/Desktop/PC/Lab4$ ./p2
Enter the value of n : 10
Matrix
4254 5258 3268 5244 7516 8007 5807 8312 7099 2353
6358 6251 5540 5419 4108 2143 7290 598 8350 7625
8443 58 4854 8151 4165 4731 3227 172 6548 1299
3457 882 6549 6725 2399 417 4732 4558 9730 8183
6911 5088 786 2451 6859 4893 947 4149 5401 5649
8127 196 2859 8833 8347 6224 9916 1575 2748 6465
9217 6285 3619 2118 9283 2376 2536 367 6928 1266
8551 192 2706 5689 8995 9566 6934 6294 67 8687
8295 4546 8884 6706 9731 3583 2931 9648 5158 2031
2465 728 4589 6084 2846 3872 4807 5382 4239 8087
Row Sum :
Row 0: 57118
Row 1: 53590
Row 2: 41139
Row 3: 46552
Row 4: 43134
Row 5: 54490
Row 6: 43909
Row 7: 57681
Row 8: 61513
Row 9: 43099
Column Sum :
Column 0: 66078
Column 1: 29324
Column 2: 42354
Column 3: 57420
Column 4: 64247
Column 5: 45086
Column 6: 49127
Column 7: 40965
Column 8: 55268
Column 9: 51636
Time taken for sequential execution is 0.000002 s
Time taken for parallel execution is 0.013019 s
priyanka@priyanka:~/Desktop/PC/Lab4$

```

The sequential takes less time than the parallel

3. Write a C/C++ OpenMP program to perform matrix multiplication. Compare the time of parallel execution with sequential execution.

Program:

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

#define N 10 //number of rows in matrix A
#define P 10 //number of columns in matrix B
#define M 10 //number of columns in matrix A

int main(int argc, char *argv[])
{
    int tid, nthreads, i, j, k;    int
    a[N][P], b[P][M], c[N][M];

    clock_t beginS = clock();
    /* code for sequential */
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < P; j++)
        {
            a[i][j] = i + j;
        }
    }

    for (int i = 0; i < P; i++)
    {
        for (int j = 0; j < M; j++)
        {
            b[i][j] = i * j;
        }
    }

    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < M; j++)
        {
            c[i][j]
= 0;
        }
    }
}
```

```

    for (i = 0; i < N; i++)
    {
        for (j = 0; j < M; j++)
            for (k = 0; k < P; k++)
c[i][j] += a[i][k] * b[k][j];
    }

    for (i = 0; i < N; i++)
    {
        for (j = 0; j < M; j++)
printf("%d\t", c[i][j]);
printf("\n");
    }

    clock_t endS = clock();
    double timeS = (double)(endS - beginS) / CLOCKS_PER_SEC;

    clock_t begin = clock();

#pragma omp parallel shared(a, b, c, nthreads) private(tid, i, j, k)
    {
        tid = omp_get_thread_num();

//initializing matrix a with values i+j
#pragma omp for
        for (i = 0; i < N; i++)
for (j = 0; j < P; j++)
a[i][j] = i + j;
//initializing matrix b with values i*j
#pragma omp for
        for (i = 0; i < P; i++)
for (j = 0; j < M; j++)
        b[i][j] = i * j;
//initializing matrix c with values 0
#pragma omp for
        for (i = 0; i < N; i++)
for (j = 0; j < M; j++)
c[i][j] = 0;

        printf("Thread %d starting matrix multiply\n", tid);
#pragma omp for
        for (i = 0; i < N; i++)

```

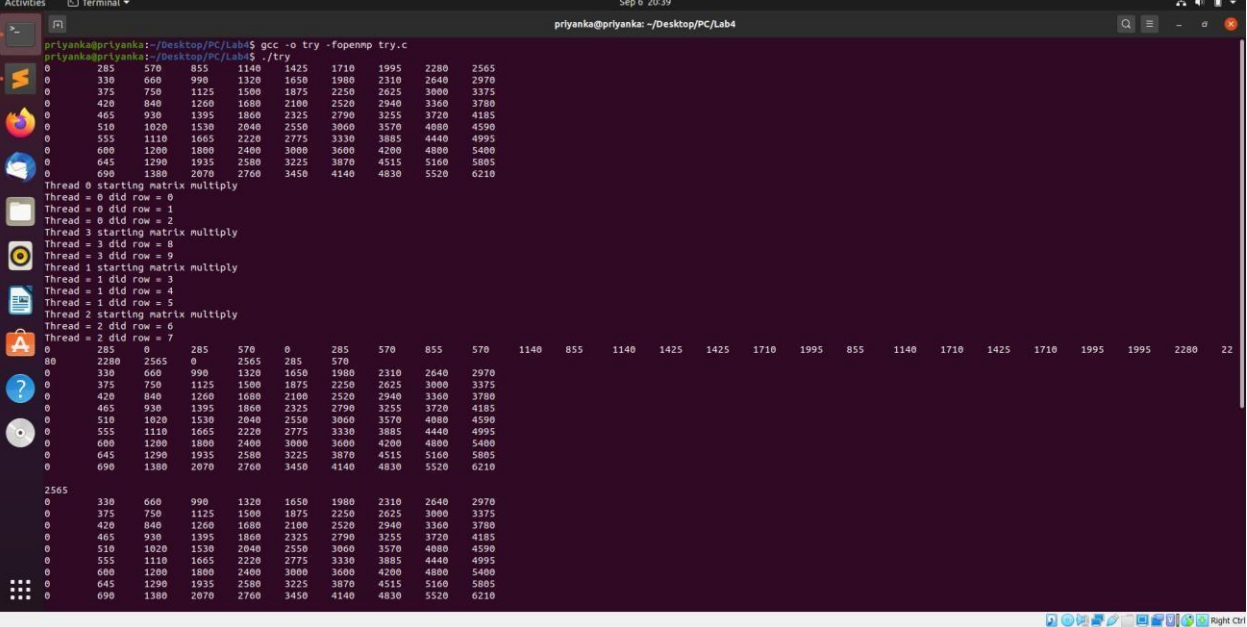
```

{
    printf("Thread = %d did row = %d\n", tid, i);
    for (j = 0; j < M; j++)
for (k = 0; k < P; k++)
    c[i][j] += a[i][k] * b[k][j];
}

    for (i = 0; i < N; i++)
    {
        for (j = 0; j < M; j++)
printf("%d\t", c[i][j]);
printf("\n");
    }
    printf("\n");
}
clock_t end = clock();
double time_spent = (double)(end - begin) / CLOCKS_PER_SEC;

printf("\t\tTIME IN SEQUENTIAL: %f\n", timeS);
printf("\t\tTIME IN PARALLEL: %f\n", time_spent);
}

```



```

prlyanka@prlyanka: ~/Desktop/PC/Lab4$ gcc -o try -fopenmp try.c
prlyanka@prlyanka: ~/Desktop/PC/Lab4$ ./try
0 285 570 855 1140 1425 1710 1995 2280 2565
0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

Thread 0 starting matrix multiply
Thread = 0 did row = 0
Thread = 0 did row = 1
Thread = 0 did row = 2
Thread 3 starting matrix multiply
Thread = 3 did row = 8
Thread = 3 did row = 9
Thread 1 starting matrix multiply
Thread = 1 did row = 3
Thread = 1 did row = 4
Thread = 1 did row = 5
Thread 2 starting matrix multiply
Thread = 2 did row = 6
Thread = 2 did row = 7
0 285 570 855 1140 1425 1710 1995 2280 2565
0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

2565
0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

```



```
Activities Terminal Sep 6 20:59
prityanka@prityanka: ~/Desktop/PC/Lab4

0 285 0 285 570 0 285 570 855 570 1140 855 1140 1425 1425 1710 1995 855 1140 1710 1425 1710 1995 1995 2280 22
80 2280 2565 0 2565 285 570
0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

2565
0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

855
0 1140 1425 1710 1995 2280 2565
0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

0 330 660 990 1320 1650 1980 2310 2640 2970
0 375 750 1125 1500 1875 2250 2625 3000 3375
0 420 840 1260 1680 2100 2520 2940 3360 3780
0 465 930 1395 1860 2325 2790 3255 3720 4185
0 510 1020 1530 2040 2550 3060 3570 4080 4590
0 555 1110 1665 2220 2775 3330 3885 4440 4995
0 600 1200 1800 2400 3000 3600 4200 4800 5400
0 645 1290 1935 2580 3225 3870 4515 5160 5805
0 690 1380 2070 2760 3450 4140 4830 5520 6210

TIME IN SEQUENTIAL: 0.000171
TIME IN PARALLEL: 0.009010
prityanka@prityanka:~/Desktop/PC/Lab4$
```

The sequential takes less time than the parallel