National Institute of Technology Karnataka Surathkal,
Mangalore - 575025

DEPARTMENT OF INFORMATION TECHNOLOGY
DEPARTMENT OF CIVIL ENGINEERING

LAB ASSIGNMENT 3

Submitted for Parallel Computing (IT301) By

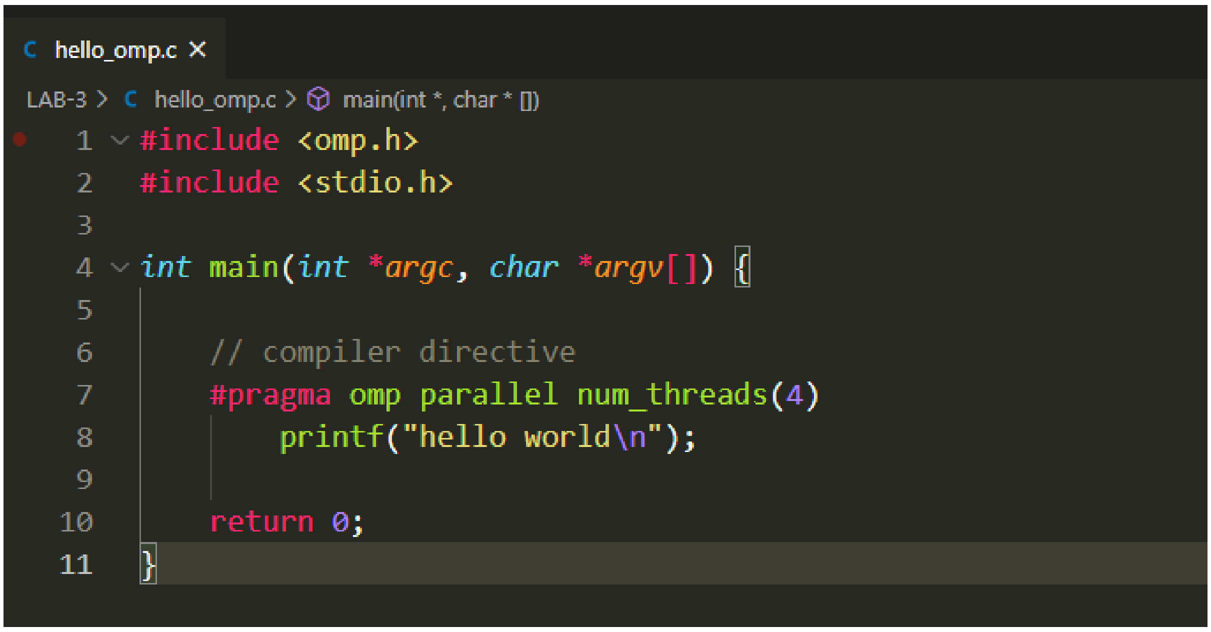
Gaurav Chaurasia	181CV155

To

Dr. Geetha V	*Dept of IT, NITK Surathkal*

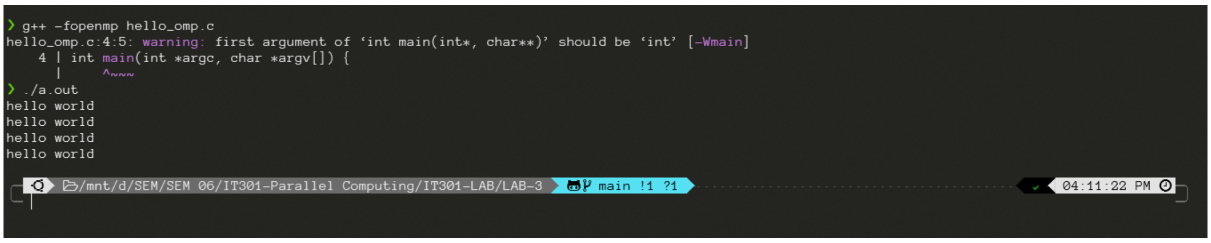
Code link
github

problem - 1 Output for number of threads = 4 is:



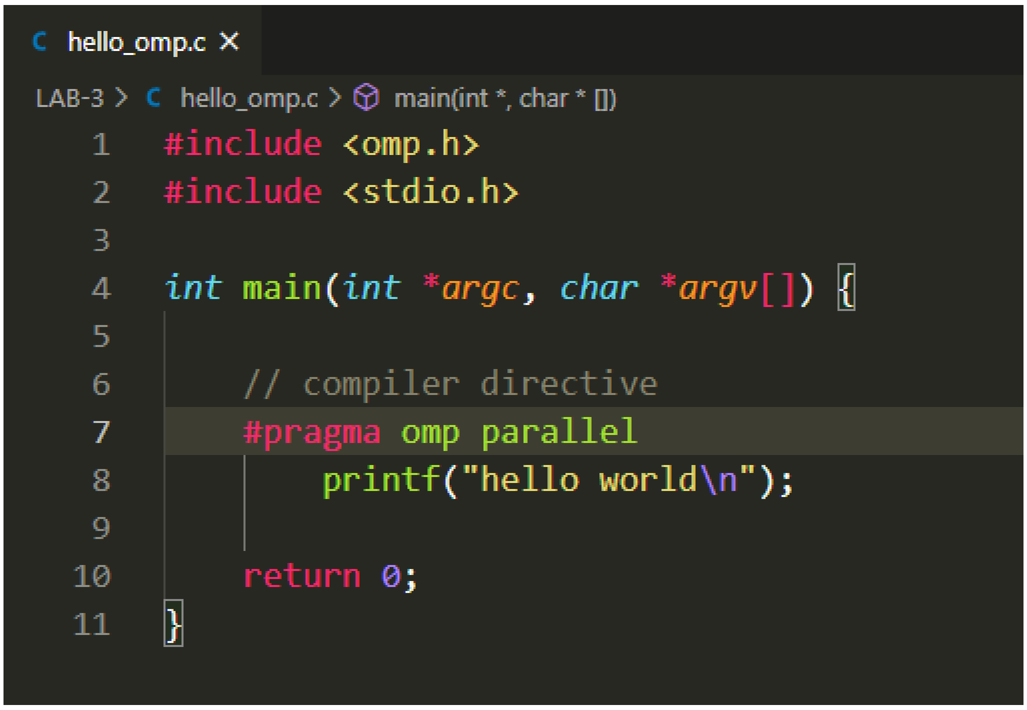
hello_omp.c ×
LAB-3 > hello_omp.c > main(int *, char * [])

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

int main(int *argc, char *argv[]) {

    // compiler directive
    #pragma omp parallel num_threads(4)
        printf("hello world\n");

    return 0;
}
```



```
❯ g++ -fopenmp hello_omp.c
hello_omp.c:4:5: warning: first argument of ‘int main(int*, char**)’ should be ‘int’ [-Wmain]
    4 | int main(int *argc, char *argv[]) {
      |     ^~~~
❯ ./a.out
hello world
hello world
hello world
hello world
 /mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3  main !1 ?1 ........ ✓ 04:11:22 PM
```

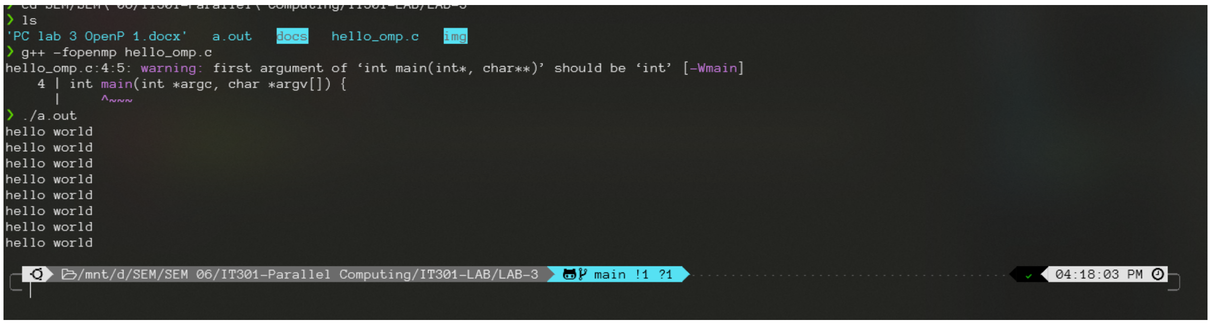
So that ***num_thread(4)*** specify that how many threads to be used in case if you don’t specify it run in all present thread or equal to environmental variable ***OMP_NUM_THREADS***
If we remove *num_thread(4)*



hello_omp.c ×
LAB-3 > hello_omp.c > main(int *, char * [])

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

int main(int *argc, char *argv[]) {

    // compiler directive
    #pragma omp parallel
        printf("hello world\n");

    return 0;
}
```

As you can see in below screenshot by default it ran in 8 diff threads



```
❯ ls
'PC lab 3 OpenP 1.docx'   a.out   docs   hello_omp.c   img
❯ g++ -fopenmp hello_omp.c
hello_omp.c:4:5: warning: first argument of ‘int main(int*, char**)’ should be ‘int’ [-Wmain]
    4 | int main(int *argc, char *argv[]) {
      |     ^~~~
❯ ./a.out
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world

 /mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3  main !1 ?1 ·························· ✓  04:18:03 PM
```

We can set environment variable by the command

```
$ export OMP_NUM_THREADS=3
```

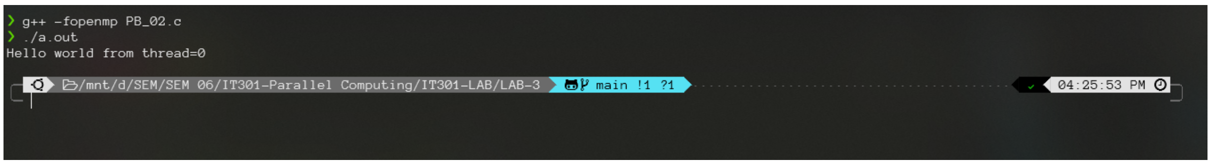
PB 02: Write a C/C++ simple program to display hello world by each thread and use if() clause

```
#include <stdio.h>
#include <omp.h>
int main()
{
    int p = 0;
#pragma omp parallel if (p == 1) num_threads(4)
    {
        int tid = omp_get_thread_num();
        printf("Hello world from thread=%d\n", tid);
    }
}
```

output

```
❯ g++ -fopenmp PB_02.c
❯ ./a.out
Hello world from thread=0

╭─ /mnt/d/SEM/IT301-LAB/LAB-3························ ✓  04:41:46 PM ─╮
╰─
```

output

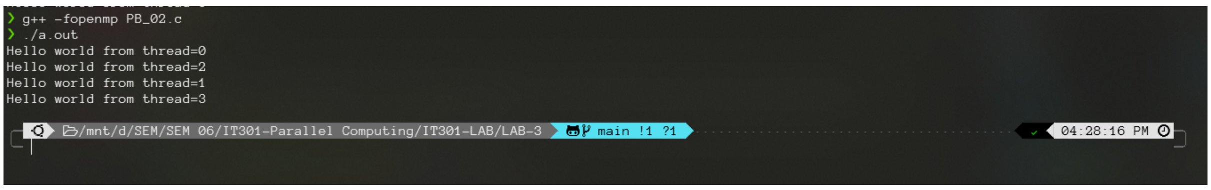


```
❯ g++ -fopenmp PB_02.c
❯ ./a.out
Hello world from thread=0
 /mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3   main !1 ?1 ........................ ✓  04:25:53 PM
```

```c
#include <stdio.h>
#include <omp.h>
int main()
{
    int p = 1;
#pragma omp parallel if (p == 1) num_threads(4)
    {
        int tid = omp_get_thread_num();
        printf("Hello world from thread=%d\n", tid);
    }
}
```

output

```
❯ g++ -fopenmp PB_02.c
❯ ./a.out
Hello world from thread=0
Hello world from thread=2
Hello world from thread=1
Hello world from thread=3

╭─ /mnt/d/SEM/IT301-LAB/LAB-3························ ✓  04:41:46 PM ─╮
╰─
```



```
❯ g++ -fopenmp PB_02.c
❯ ./a.out
Hello world from thread=0
Hello world from thread=2
Hello world from thread=1
Hello world from thread=3
 /mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3   main !1 ?1 ........................ ✓  04:28:16 PM
```

PB 03. Write a program to demonstrate private() firstprivate() and shared() clauses with parallel directive.

```
#include <stdio.h>
#include <omp.h>
int main()
{
    int a = 10;
//default(shared/none) private(list)  firstprivate(list) shared(list)
#pragma omp parallel num_threads(4) private(a)
    {
        a = a + 20;
        int tid = omp_get_thread_num();
        printf("Hello world from thread=%d value of a=%d\n", tid, a);
    }
    printf("After parallel loop a=%d\n", a);
}
```

#PRIVATE(a)
Here, the value of a declared outside the pragma omp parallel is not considered, hence a = a+20 = 20 (a’s default value is 0). Outside this block origin value of a = 10 is printed

output

```
❯ g++ -fopenmp PB_03.c
❯ ./a.out
Hello world from thread=2 value of a=20
Hello world from thread=0 value of a=20
Hello world from thread=3 value of a=20
Hello world from thread=1 value of a=20
After parallel loop a=10
```

#FIRSTPRIVATE(a)
Here, the value of a declared outside is accessible inside the pragma parallel block, hence a = 10 + 20 = 30 is printed. But outside this block original declared value of a = 10, is printed

output

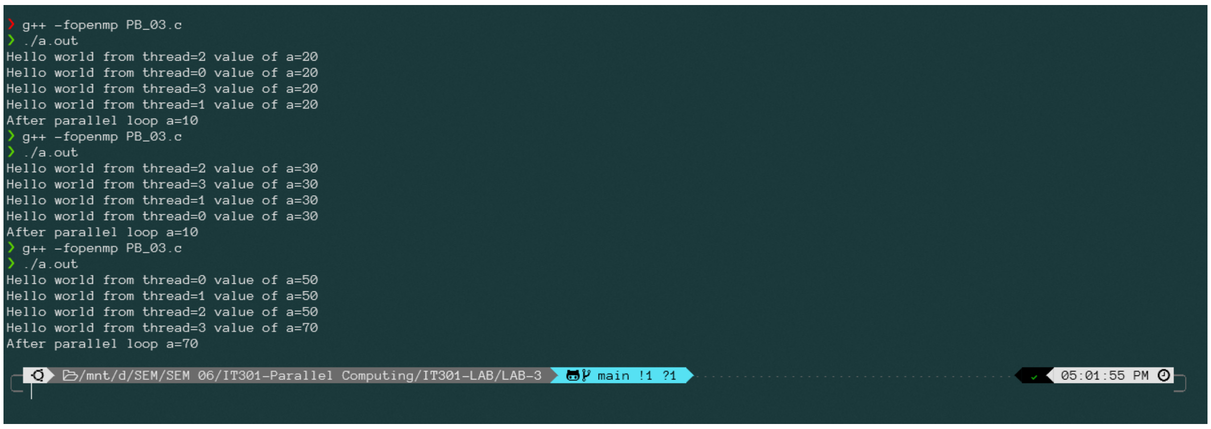
```
❯ g++ -fopenmp PB_03.c
❯ ./a.out
```

```
Hello world from thread=2 value of a=30
Hello world from thread=3 value of a=30
Hello world from thread=1 value of a=30
Hello world from thread=0 value of a=30
After parallel loop a=1
```

#SHARED(a)
Here, the value of the first thread is shared with the next thread.
Hence, for first thread, a = 10 + 20 = 30
for second thread, a = 30 + 20 = 50
for third thread, a = 50+20 = 70
for fourth thread, a = 70 + 20 = 90 Since, it is shared, so last value of a is printed outside the pragma parallel block.

output

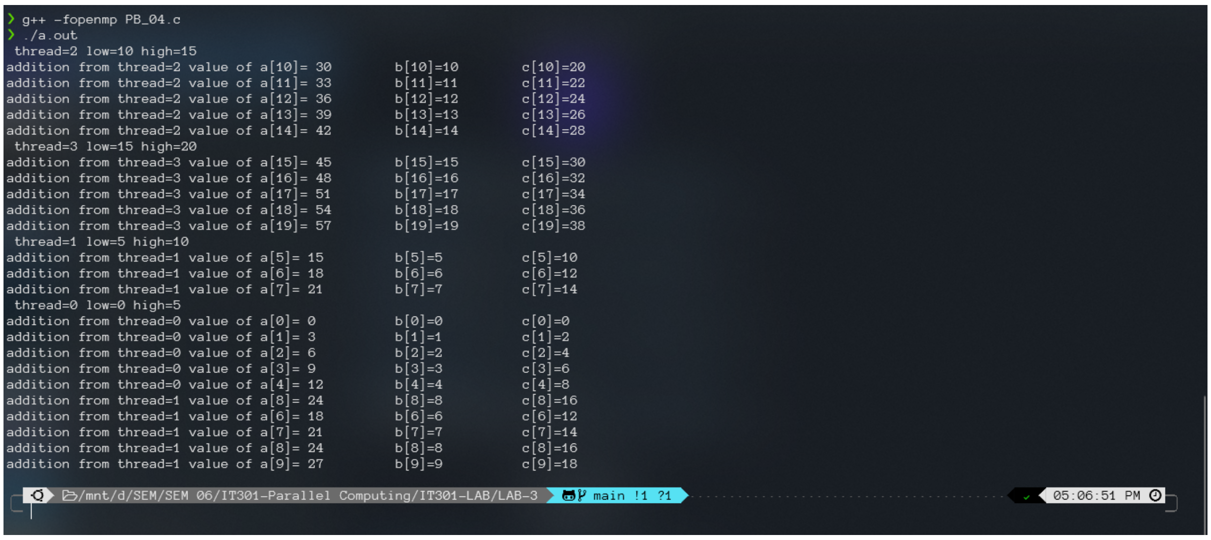
```
❯ g++ -fopenmp PB_03.c
❯ ./a.out
Hello world from thread=0 value of a=50
Hello world from thread=1 value of a=50
Hello world from thread=2 value of a=50
Hello world from thread=3 value of a=70
After parallel loop a=70


╭─ /mnt/d/SEM/IT301-LAB/LAB-3························ ✓  04:41:46 PM ─╮
╰─ |
```



```
❯ g++ -fopenmp PB_03.c
❯ ./a.out
Hello world from thread=2 value of a=20
Hello world from thread=0 value of a=20
Hello world from thread=3 value of a=20
Hello world from thread=1 value of a=20
After parallel loop a=10
❯ g++ -fopenmp PB_03.c
❯ ./a.out
Hello world from thread=2 value of a=30
Hello world from thread=3 value of a=30
Hello world from thread=1 value of a=30
Hello world from thread=0 value of a=30
After parallel loop a=10
❯ g++ -fopenmp PB_03.c
❯ ./a.out
Hello world from thread=0 value of a=50
Hello world from thread=1 value of a=50
Hello world from thread=2 value of a=50
Hello world from thread=3 value of a=70
After parallel loop a=70

/mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3  main !1 ?1 ························ ✓  05:01:55 PM
```

PB 04

we can see that for each thread arrays a[], b[], and c[] are executed
And as low and high were private initialized by default value 0



```
❯ g++ -fopenmp PB_04.c
❯ ./a.out
 thread=2 low=10 high=15
addition from thread=2 value of a[10]= 30       b[10]=10        c[10]=20
addition from thread=2 value of a[11]= 33       b[11]=11        c[11]=22
addition from thread=2 value of a[12]= 36       b[12]=12        c[12]=24
addition from thread=2 value of a[13]= 39       b[13]=13        c[13]=26
addition from thread=2 value of a[14]= 42       b[14]=14        c[14]=28
 thread=3 low=15 high=20
addition from thread=3 value of a[15]= 45       b[15]=15        c[15]=30
addition from thread=3 value of a[16]= 48       b[16]=16        c[16]=32
addition from thread=3 value of a[17]= 51       b[17]=17        c[17]=34
addition from thread=3 value of a[18]= 54       b[18]=18        c[18]=36
addition from thread=3 value of a[19]= 57       b[19]=19        c[19]=38
 thread=1 low=5 high=10
addition from thread=1 value of a[5]= 15        b[5]=5          c[5]=10
addition from thread=1 value of a[6]= 18        b[6]=6          c[6]=12
addition from thread=1 value of a[7]= 21        b[7]=7          c[7]=14
 thread=0 low=0 high=5
addition from thread=0 value of a[0]= 0         b[0]=0          c[0]=0
addition from thread=0 value of a[1]= 3         b[1]=1          c[1]=2
addition from thread=0 value of a[2]= 6         b[2]=2          c[2]=4
addition from thread=0 value of a[3]= 9         b[3]=3          c[3]=6
addition from thread=0 value of a[4]= 12        b[4]=4          c[4]=8
addition from thread=1 value of a[8]= 24        b[8]=8          c[8]=16
addition from thread=1 value of a[6]= 18        b[6]=6          c[6]=12
addition from thread=1 value of a[7]= 21        b[7]=7          c[7]=14
addition from thread=1 value of a[8]= 24        b[8]=8          c[8]=16
addition from thread=1 value of a[9]= 27        b[9]=9          c[9]=18

 /mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3  main !1 ?1            ✓  05:06:51 PM
```

PB_05 Write a C/C++ program to calculate the sum of all the elements in an array. Assume array size =20 and number of threads = 04

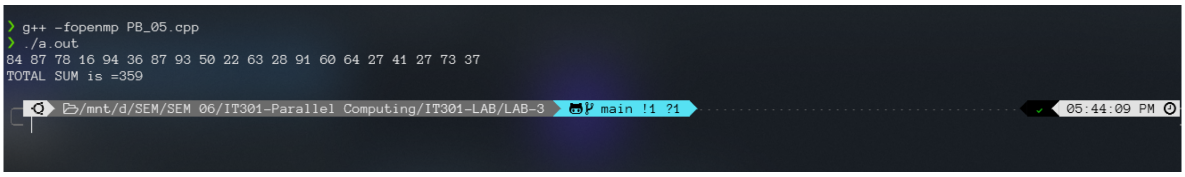
This ***get_randome_array*** accept array as an input parameter by reference and set it to random number for calculating sum
And then in four diff threads we calculate sum of array in parts and finally sum it up

```
#include <stdio.h>
#include <iostream>
#include <climits>
#include <omp.h>

using namespace std;

#define SIZE 20

void get_randome_array(int arr[]) {
    for (int i = 0; i < SIZE; i++) {
        arr[i] = 1 + (rand() % 100);
    }
}

int main()
{
    int SUM1 = 0,
        SUM2 = 0,
        SUM3 = 0,
        SUM4 = 0;
    int arr[SIZE];
    get_randome_array(arr);
    for (int i = 0; i < SIZE; i++) {
        cout << arr[i] << " ";
    } cout << endl;

    // shared(list)
#pragma omp parallel num_threads(1) shared(SUM1)
    {
        if (omp_get_thread_num() == 0) {
            for (int i = 0; i < SIZE/4; i++) {
                SUM1 += arr[i];
            }
        }
    }
#pragma omp parallel num_threads(1) shared(SUM2)
    {
        if (omp_get_thread_num() == 1) {
            for (int i = SIZE / 4; i < SIZE / 2; i++) {
                SUM2 += arr[i];
            }
        }
    }
#pragma omp parallel num_threads(1) shared(SUM3)
    {
        if (omp_get_thread_num() == 2) {
            for (int i = SIZE / 2; i < 3 * (SIZE / 4); i++)
{
```

```
                SUM3 += arr[i];
            }
        }
    }
#pragma omp parallel num_threads(1) shared(SUM4)
    {
        if (omp_get_thread_num() == 3) {
            for (int i = 3 * (SIZE / 4); i < SIZE; i++) {
                SUM4 += arr[i];
            }
        }
    }
    printf("TOTAL SUM is =%d\n", SUM1 + SUM2 + SUM3 +
SUM4);
}
```

Output

```
❯ g++ -fopenmp PB_05.cpp
❯ ./a.out
84 87 78 16 94 36 87 93 50 22 63 28 91 60 64 27 41 27 73 37
TOTAL SUM is =359

╭─ /mnt/d/SEM/IT301-LAB/LAB-3························ ✓  04:41:46 PM ─╮
╰─ ./a.out
```



```
❯ g++ -fopenmp PB_05.cpp
❯ ./a.out
84 87 78 16 94 36 87 93 50 22 63 28 91 60 64 27 41 27 73 37
TOTAL SUM is =359
  /mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-3  main !1 ?1 ···················· ✓  05:44:09 PM
```