Parallel Programming L3 UFAZ

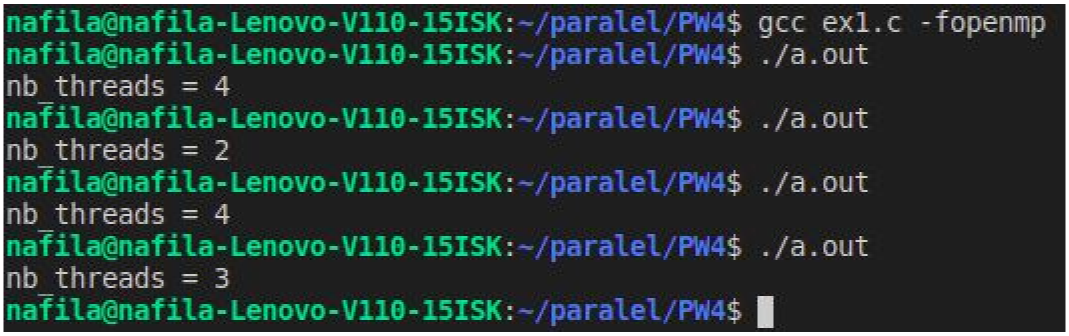
REPORT- Practical Session 4: Synchronization and Tasks

Nafila Amirli

1 Number of Threads

Study and correct the following code using two different approaches. You are only allowed to add OpenMP directive without the reduction clause.

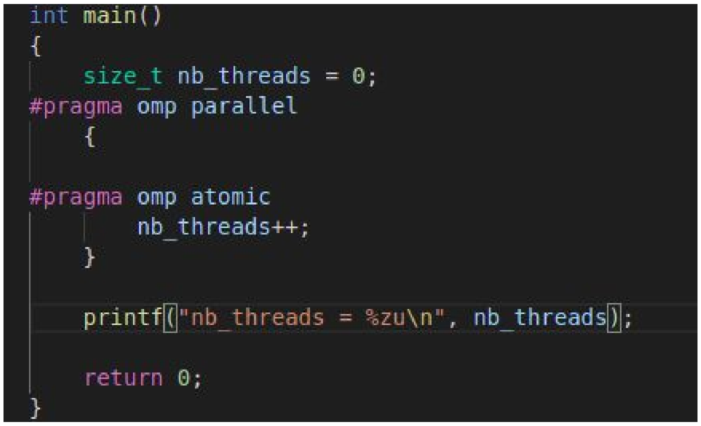
When we run the given version:



```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ gcc ex1.c -fopenmp
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 2
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 3
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ 
```

As we see, the results are random in each run.

First correction code:



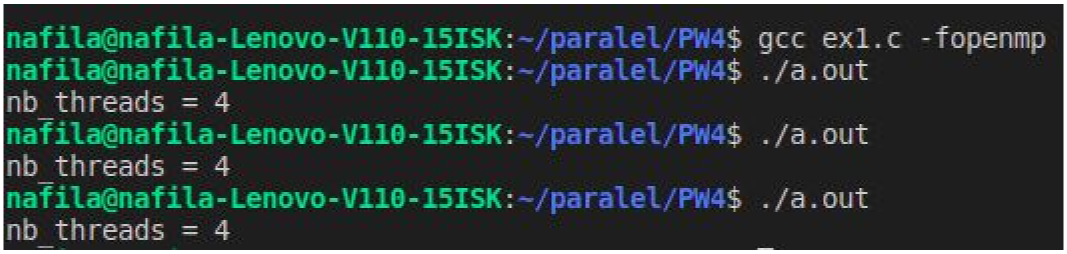
```
int main()
{
    size_t nb_threads = 0;
#pragma omp parallel
    {

#pragma omp atomic
        nb_threads++;
    }

    printf("nb_threads = %zu\n", nb_threads);

    return 0;
}
```

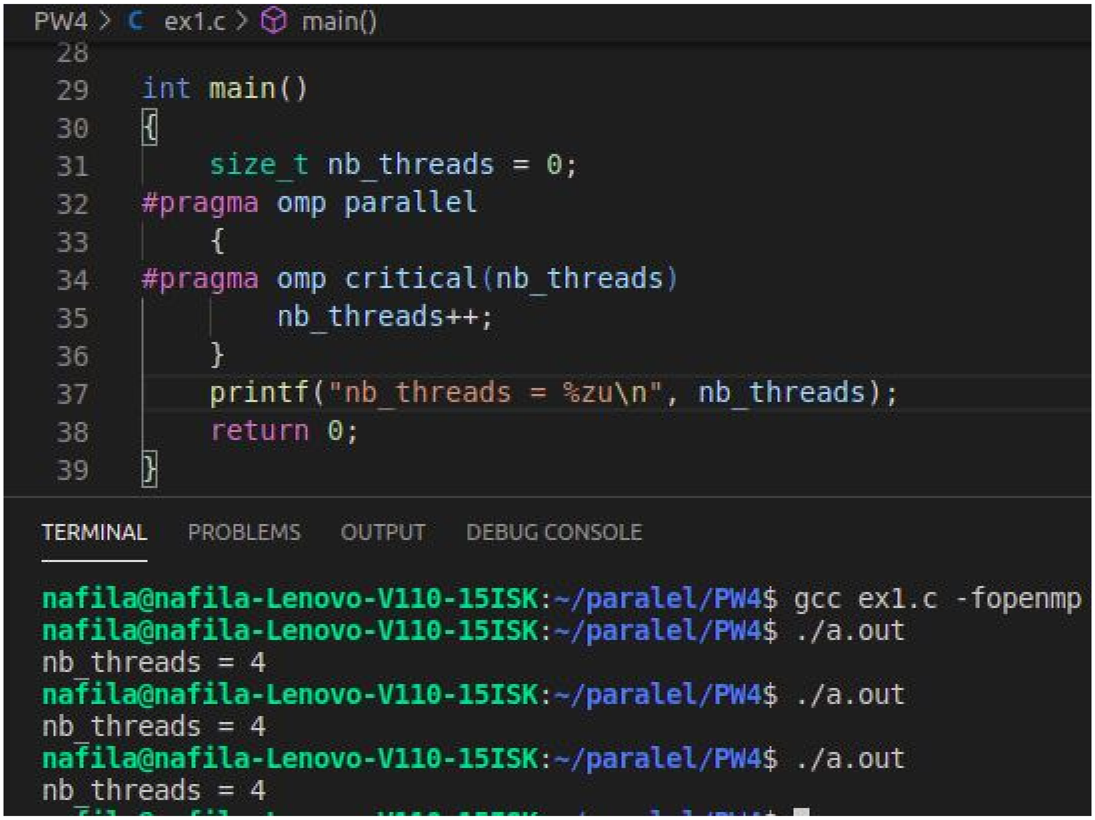
The terminal result of first correction:



```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ gcc ex1.c -fopenmp
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
```

In first correction the atomic directive is used which forbids all the threads to access simultaneously to the shared variable.

Second correction code and terminal result:



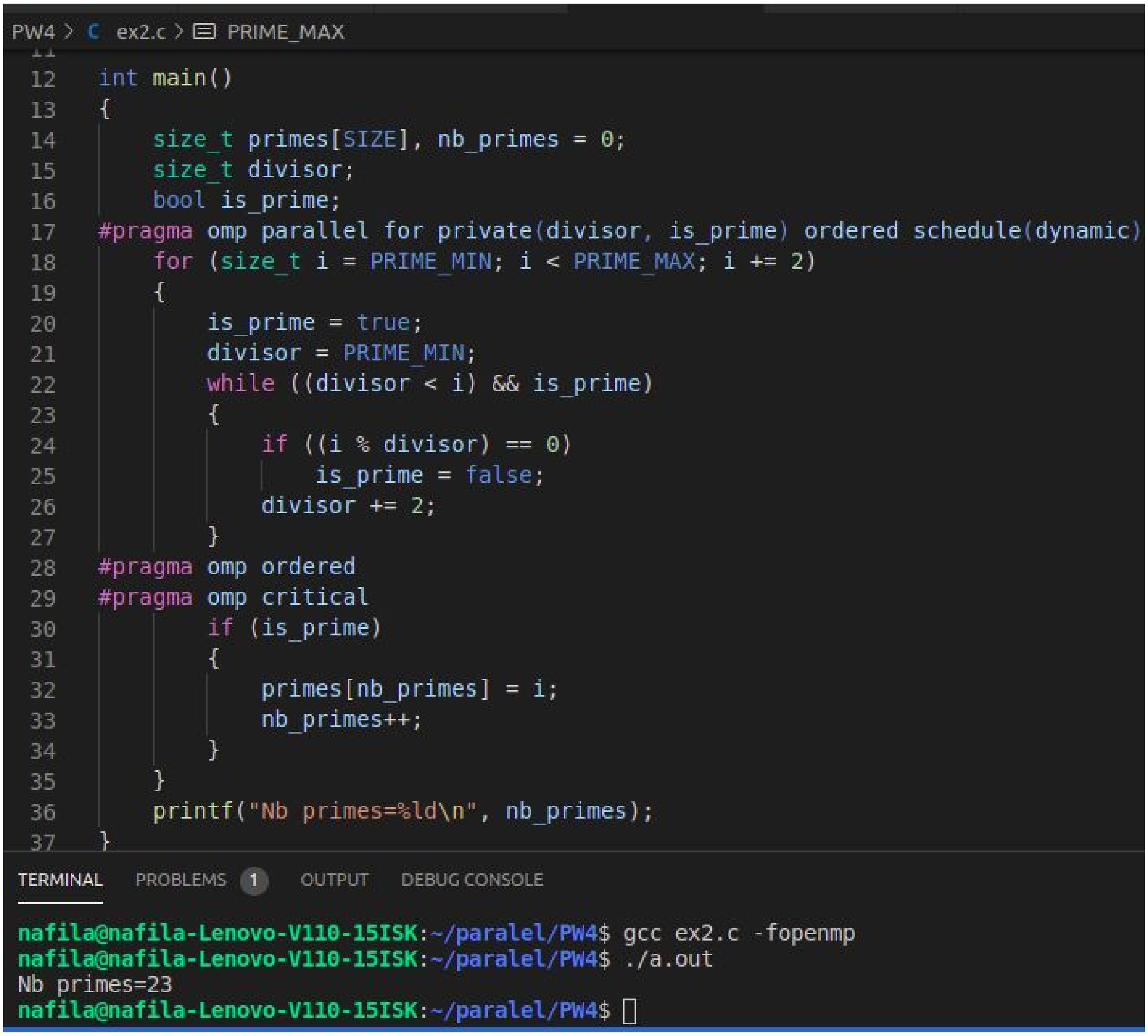
```c
int main()
{
    size_t nb_threads = 0;
#pragma omp parallel
    {
#pragma omp critical(nb_threads)
        nb_threads++;
    }
    printf("nb_threads = %zu\n", nb_threads);
    return 0;
}
```

```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ gcc ex1.c -fopenmp
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
nb_threads = 4
```

The critical directive is used to prevent simultaneous access to the nb_threads variable.

2 First Prime Numbers

The parallelized version of the given code:



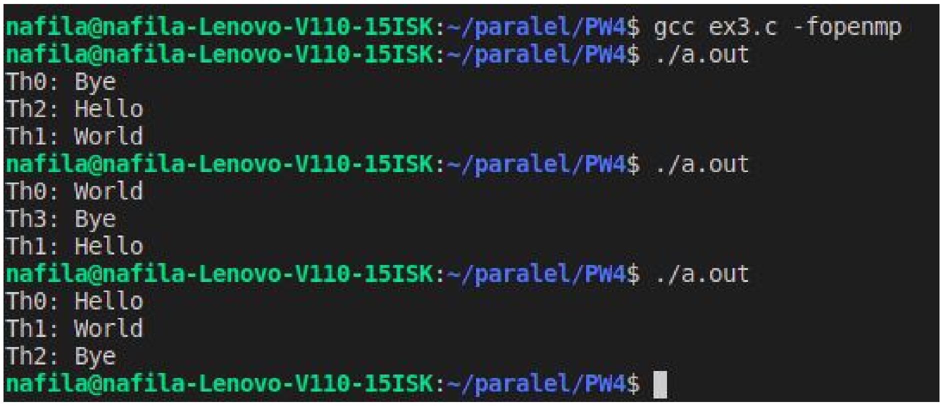
```c
int main()
{
    size_t primes[SIZE], nb_primes = 0;
    size_t divisor;
    bool is_prime;
#pragma omp parallel for private(divisor, is_prime) ordered schedule(dynamic)
    for (size_t i = PRIME_MIN; i < PRIME_MAX; i += 2)
    {
        is_prime = true;
        divisor = PRIME_MIN;
        while ((divisor < i) && is_prime)
        {
            if ((i % divisor) == 0)
                is_prime = false;
            divisor += 2;
        }
#pragma omp ordered
#pragma omp critical
        if (is_prime)
        {
            primes[nb_primes] = i;
            nb_primes++;
        }
    }
    printf("Nb primes=%ld\n", nb_primes);
}
```

```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ gcc ex2.c -fopenmp
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Nb primes=23
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ 
```

3 Exercise - synchronization using lock

1. Compile the program, and observe the behavior over multiple runs. What do you observe?

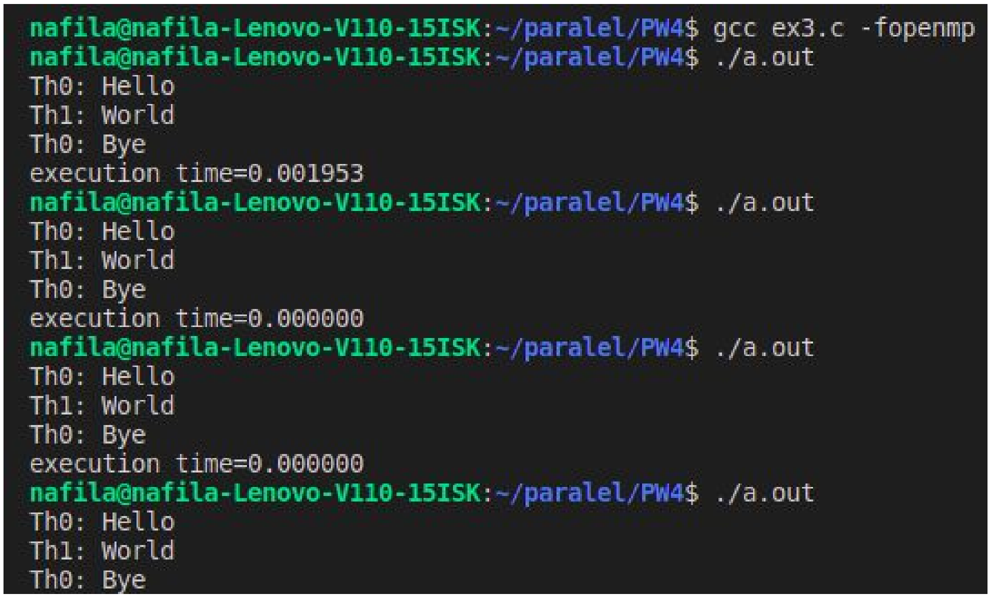
-The results are random in every run as follows:



```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ gcc ex3.c -fopenmp
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: Bye
Th2: Hello
Th1: World
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: World
Th3: Bye
Th1: Hello
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: Hello
Th1: World
Th2: Bye
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ 
```

2. Modify the program and use locks in order to obtain a correct execution. (Note: you need two locks to obtain the correct behavior).

-When we use lock the results are ordered in every run:



```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ gcc ex3.c -fopenmp
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: Hello
Th1: World
Th0: Bye
execution time=0.001953
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: Hello
Th1: World
Th0: Bye
execution time=0.000000
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: Hello
Th1: World
Th0: Bye
execution time=0.000000
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW4$ ./a.out
Th0: Hello
Th1: World
Th0: Bye
```