UFAZ
FRENCH-AZERBAIJANI UNIVERSITY

Practical Session 4: Synchronization and Tasks

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.

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

2 First Prime Numbers

Study and parallelize the following code:

```
#include <stdio.h>
#include <omp.h>
int main()
{
size_t primes[],nb_primes=0;

size_t divisor;
bool is_prime;
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;
}
if (is_prime) {
primes[nb_primes] = i;
nb_primes++;
}
}
printf("Nb primes=%d\n",nb_primes);
}
```

3 Exercise - synchronization using lock

Consider the following code:

```
#include <omp.h>
#include <stdlib.h>
#include <stdio.h>
int main()
{
  int p;
  #pragma omp parallel sections default(shared)
  {
     #pragma omp section
    {
      p = omp_get_thread_num();
      printf("Th%d: Hello\n",p);
    }
     #pragma omp section
    {
      p = omp_get_thread_num();
      printf("Th%d: World\n",p);
    }
     #pragma omp section
    {
      p = omp_get_thread_num();
      printf("Th%d: Bye\n",p);
    }
  }
  return 0;
}
```

1. Compile the program, and observe the behavior over multiple runs. What do you observe?
2. Modify the program and use locks in order to obtain a correct execution. (Note: you need two locks to obtain the correct behavior).