CS511: Homework Assignment 1

Due: Sunday, September 09, 11:55pm

1 Assignment Objectives

Get acquainted with how to implement threads in Java.

2 Assignment Policies

Collaboration Policy This homework may be done individually or in pairs. Use of the internet is allowed, but should not include searching for existing solutions.

Under absolutely no circumstances can code be exchanged between students. Excerpts of code presented in class can be used.

Assignments from previous offerings of the course must not be reused. Violations will be penalized appropriately.

Late Policy. Late submissions are allowed with a penalty of 2 points per hour past the deadline.

3 Assignment

Implement a class **AssignmentOne** that includes a method

```
public static List<Integer> lprimes(List<Integer[]> intervals);
```

that given a list of arrays of integers of size 2 $[[g_1, d_1], ..., [g_k, d_k]]$ such that the following conditions hold (your code must check this!):

- the list $[g_1, d_1, ..., g_k, d_k]$ is increasing
- each number is greater than or equal to 2

creates k threads where each thread i computes the list of primes in the interval $[g_i, d_{i+1})$. For example, given the list of arrays

$$[[2, 101], [101, 201], [201, 301], [301, 401], [401, 501]]$$

it will spawn 5 processes. Process 2 will, for example, compute all the primes between 101 and 200, namely:

$$[101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199]$$

The result of the whole process would be the list:

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101,
103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197,
199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311,
313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431,
433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499]

Additional guidelines:

1. The list of arrays should be built from arguments that are passed on to `main`, i.e. command line arguments. For example, the list $[2, 101], [101, 201], [201, 301], [301, 401], [401, 501]]$ arises from the user having passed the arguments `2 101 201 301 401 501` to the `main` method of class `AssignmentOne`. The main function should yield the result of `lprimes` to `stdout`.

2. The class `AssignmentOne` should make use of a helper class `PrimeFinder` that implements the `Runnable` interface and has the following attributes:

   ```
   private Integer start;
   private Integer end;
   private List<Integer> primes;
   ```

   and the following methods:

   ```
   /* Constructs a PrimeFinder */
   public PrimeFinder(Integer startNum, Integer endNum);

   /* Returns the value of the attribute 'primes' */
   public List<Integer> getPrimesList();

   /* Determines whether the argument is prime or not */
   public Boolean isPrime(int s);

   /* Adds all primes in [this.start, this.end) to the attribute 'primes' */
   public void run();
   ```

3. In `AssignmentOne` you will require a list of threads. Also, you will have to wait for all the threads in the list to finish before collecting the results. In order to wait for a thread `t` to finish you may use the command `t.join()`.

4 Submission Instructions

Submit a zip file named `traces_<Surnames>.zip` (where `<Surnames>` should be replaced by the surnames of the members of the group separated by underscores) through Canvas containing the files `PrimeFinder.java` and `AssignmentOne.java` that includes the surnames of all members of the group as a comment header in `AssignmentOne.java`.