Exercise 6 (10 points)

- The first lines of all source files must be comment containing your name & ID
- Put all files (source, input, output) in folder Ex6_xxx where xxx = your full ID. That is, your source files must be in package Ex6_xxx and input/output files (if there is any) must be read from/write to this folder
- Zip Ex6_xxx & submits it to Google Classroom. Email submission is not accepted

==========

1. Complete class PrimeThread. Modify them as needed. You can add more variables & methods, but do not change the visibility of existing ones.

```
class PrimeThread extends Thread {
  private PrintWriter         out;        // each thread writes to separate file
  private ArrayList<Integer>  numbers;
  private int                 totalPrime, target;

  public PrimeThread(String n, int t)     { super(n); target = t; }

  public void run() {
    // Execute steps 1-4 in loop
    //   1. Random 5 integers (2-100) & put them in ArrayList
    //   2. Sort the ArrayList in increasing order
    //   3. Check each member. If it is a prime, add it to totalPrime.
    //   4. Print round number, all sorted values (primes must be printed with + sign, non-
    //      primes must be printed without + sign), and current totalPrime to file
    //
    // Stop the loop once totalPrime >= target
    // Report number of rounds to the screen
  }
}
```

** The output file must be placed in the same folder as your source file

Note 1 – to use java.util.Random object

```
Random random = new Random();
Random.nextInt(11);           // an integer in [0, 11) range
Random.nextInt(5, 11);        // an integer in [5, 11) range
```

2. Write another class that acts as the main class. In its main method
   2.1 Ask user for #threads
   2.2 Ask user for target
   2.3 Create PrimeThreads to perform tasks in (1)

```
--- exec-maven-plugin:3.0.0:exec
Enter #threads =
3
Enter target =
600
Thread_0 finishes in 14 rounds
Thread_2 finishes in 11 rounds
Thread_1 finishes in 13 rounds
------------------------------------
BUILD SUCCESS
------------------------------------
```

In different runs, the finishing order between threads should be different. If it is always Thread_0, Thread_1, Thread_2, …, then you may not do multithreaded program properly

Threads also compete for System.out. So, if #rounds are close, the one who finishes first may get System.out later

```
Thread_0, target = 600

Round   1  >>    14     46     54    +79     94         total prime =   79
Round   2  >>     6     14     35    +71     72         total prime =  150
Round   3  >>     4     +5     42     88     96         total prime =  155
Round   4  >>    +2     51     62     66     69         total prime =  157
Round   5  >>     4     +5    +23     34     96         total prime =  185
Round   6  >>    62     90     92     96    100         total prime =  185
Round   7  >>   +13     25     50     57     63         total prime =  198
Round   8  >>     8     26    +47     49     96         total prime =  245
Round   9  >>   +17     48     65     76     81         total prime =  262
Round  10  >>     6    +29     44     90     93         total prime =  291
Round  11  >>   +29     46     76    +79     94         total prime =  399
Round  12  >>    +2     +7    +11    +23    +97         total prime =  539
Round  13  >>   +17    +19     56     77     94         total prime =  575
Round  14  >>    22    +47     55    +59    +89         total prime =  770
```

```
Thread_1, target = 600

Round   1  >>     8     25     50     72     80         total prime =    0
Round   2  >>    22     63     70     81     85         total prime =    0
Round   3  >>    14     27     30     72     82         total prime =    0
Round   4  >>    +3     24     26    +71     99         total prime =   74
Round   5  >>    28     77    +83     93     96         total prime =  157
Round   6  >>     9     20     26     77    +97         total prime =  254
Round   7  >>    +2     12    +31     49    +67         total prime =  354
Round   8  >>   +13     35     42     82     87         total prime =  367
Round   9  >>    48     68     70     82     94         total prime =  367
Round  10  >>    35     42     44     62     81         total prime =  367
Round  11  >>    57    +61     66    +71     88         total prime =  499
Round  12  >>    18    +19    +19     24     54         total prime =  537
Round  13  >>     9     22     24    +79     80         total prime =  616
```

```
Thread_2, target = 600

Round   1  >>    40     46    +67    +83    +97         total prime =  247
Round   2  >>    +2     54     78     81     90         total prime =  249
Round   3  >>    10     20     39     62     98         total prime =  249
Round   4  >>     8     46     70     70    +97         total prime =  346
Round   5  >>   +11     20     70     74     76         total prime =  357
Round   6  >>    50    +71     80     81     96         total prime =  428
Round   7  >>    39     49     69     90     91         total prime =  428
Round   8  >>    14     25     63     84     94         total prime =  428
Round   9  >>    33    +41     50     52    100         total prime =  469
Round  10  >>     9    +29     56     57    +61         total prime =  559
Round  11  >>    28     44    +73     78     90         total prime =  632
```