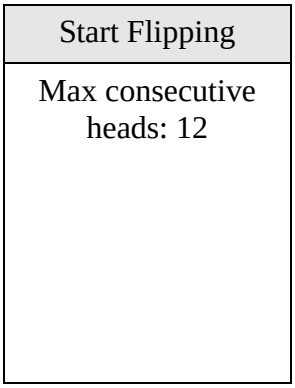
Multithreaded Programming: Part II

1. Make a variation of the coin-flipping problem from the previous set of exercises, where you start five threads, and each thread keeps track of the number of consecutive threads. This time, however, keep track of the highest number of consecutive heads (by any thread) and display the final results in the Android window. That is, after *all* of the coins are done flipping, display the highest number of consecutive heads found by any of the threads. Your result will look something like this:



Hint 1: use "shutdown" and "awaitTermination". Hint 2: since your threads will presumably frequently update a shared instance variable that stores the max head-count from any of the threads, you will probably need to use "synchronized". But, if you use the inner-class approach for the Runnable class, you cannot do "synchronized(this)" because each "this" is different. So, do "synchronized(OuterClass.this)", or use the main-class-implements-Runnable approach.

2. Do a similar task, but this time, every time you have a new max, display it right away. Use View.post. You might add a small sleep time (e.g., 0.01 seconds) after each coin flip so that you can see the updates. Use Thread.sleep or copy my ThreadUtils and use ThreadUtils.pause.

3. Launch five different AsyncTasks, each of which flips a different number of coins and collects the maximum number of consecutive heads. Have the doInBackground method return a TextView that is added to the bottom of the main LinearLayout (just as in the Rotations exercises).