Computer Science 435
Project 2: Concurrent Pattern Matching

Due: Fri, Oct. 28, at the beginning of class. You may implement your solution either in Java using Jthreads or C using Pthreads.

A *gap string* is a set of strings that all have a specified non-overlapping prefix and suffix. Any characters may appear between the prefix and the suffix. A *limited gap string*, (LGS) is a gap string where only a specified number of characters may appear between the prefix and the suffix. We specify a limited gap string using the notation `prefix[min,max]suffix`, $0 \leq$ `min` $\leq$ `max`. `Min` and `max` are integers giving the minimum and maximum number of characters that may appear between the prefix and suffix. For example the limited gap string `foo[0,3]bar` includes all the following strings (and many others): ``foobar'', ``fooabar'', ``foo sbar'', `fools bar''`. Case is significant.

Write a program that will take the name of a text file and a limited gap string as command line arguments, and for each line of the file, print out the following information:

1. the line number of the file, followed by a colon and a space;
2. the number of limited gap strings found in that line in the form "`%d matches`";
3. the original line from the file.

On output, this information must be printed in the same order as the lines in the original file. In addition, your program must meet the following specifications.

- You must define a `struct` or object that holds all of the information associated with one line of text, and a list that contains one of these objects for each line in the file.
- Reading the lines of the file and initializing the list should be done in the main program.
- Create one thread for each value in the LGS range. (The example above specified 4 threads.) Each thread is responsible for searching for one particular gap string, but it must search each line of the file.

You may assume that no line of the text file contains more than 1024 characters, excluding the terminating '\n', and that there are no more than 2048 lines in the file.

Do not constrain the concurrency of your program any more than necessary, subject to correct operation of your algorithm. Do not use any other synchronization facilities provided by the programming language except either `pthread_mutex_t`'s (if you are programming in C) or `ReentrantLock`'s (in Java).

What to turn in: When you have finished with the program, send an email message to me that contains, as an attachment, a copy of your source file (for C) or a tarball of your source files (for Java). Include in your email any instructions that I will need to follow to extract, compile, and execute your program on the departmental Linux server sand.

Then, print a hard copy of the source to turn in. (If you are writing in Java, make sure the Main class is on top.) You must submit your source hard copy within one class day of your electronic submission to receive full credit. I will use the date of your electronic submission as the official submission time, unless you fail to punctually submit the required hard copy.