These must be completed and shown to your TA either by the end of this lab, or at the start of your next lab. You may work in groups of up to two people.

1. Start an SSH session on one of the department servers.

2. Download the modified version of the AVL tree code you have already used (along with associated files) from the course web page under this lab's entry. Then, you can access the 2 big text files at:

   `/home/c/cs221/public_html/current/handouts/labs/lab9/samples`

   An example of running the program is:

   ```
   cs221@bowen:lab9>  ./avl /home/c/cs221/public_html/current/handouts/labs/lab9/samples/kjb.txt
   ```

   Or, if you have trouble with permissions, you can download the big text files from:

   `http://www.ugrad.cs.ubc.ca/~cs221/current/handouts/labs/lab9/samples/kjb.txt`

   or

   `http://www.ugrad.cs.ubc.ca/~cs221/current/handouts/labs/lab9/samples/warNpc.txt`

   Then, you can use the wget command to download the file. At the command line, run:

   `wget www.ugrad.cs.ubc.ca/~cs221/current/handouts/labs/lab9/samples/warNpc.txt`

3. Complete the function `findStatsHelper`, which is called by `findStats` in order to find some statistics on the word frequency distribution of a given text file using divide-and-conquer fork/join style parallelism.

   Your solution should fork off tasks to handle subtrees until you can guarantee by looking at the root node of a subtree that it has at most 1100 nodes, at which point it should switch to the provided sequential version instead. Be prepared to briefly justify your cutoff to your lab TA.

   Hint: `findStatsHelper` is essentially a modification of `findStatsSequential`, so you may find it useful to start by copying that code.

4. Time your solution using the two sample text files mentioned above:

   We provide the King James Bible and War and Peace because they're big and it's interesting to see the difference in their statistics as well as the parallel speed-up. Note that you do *not* need to copy these files! Just pass the path to one of the files as the command line argument to your `avl` program.

5. For each sample file that we provide, find a cutoff for which the sequential algorithm performs better (in terms of runtime) than the parallel version, and vice-versa.

6. Be sure to show your work to your TA, or you will not receive credit for the lab!