SE375 - Laboratory Assignment 02

Part 1: Word Count with Threads

You have developed the Word Count application in the previous laboratory assignment without using threads. Your task now is to write a Java application that will count the words in a list of text files that are passed as parameters to the main function *using one thread per file.*

You should follow these steps in your Java applications: (1) For each file passed as a parameter, you will need a thread to parse the file and a data structure passed to the thread to save the results. (2) Each file reads the file assigned to it, and uses the data structure to add or update the words. (3) Once all threads are complete, use data structures that are assigned to the threads to aggregate their results to another data structure. (4) Compare the results in this aggregated data structure to the results you have obtained from the Word Count application without the threads that you have implemented last week.

Consider the flow diagram given in Figure 1. Although the figure is for n number of files, assume that three files have been passed as arguments ($n = 3$). In the main method, four data structures are created: three of them for each of three files (indices 1 to 3), and one for aggregating the results (index 0). Data structures of indices 1 to 3 are passed as parameters to each thread along with the file names from the *args* parameter. Each thread works on the data structure and the file they are assigned to. Once they are all complete, the main method aggregates results to the remaining data structure with index 0.

Sample Run

```
C:\java ThreadedWordCount file1 file2 file3
Thread parsing file2...
Thread parsing file3...
Thread parsing file1...
Threads are complete. Aggregating results.
There are 4219 distinct words in files file1, file2, file3.
```

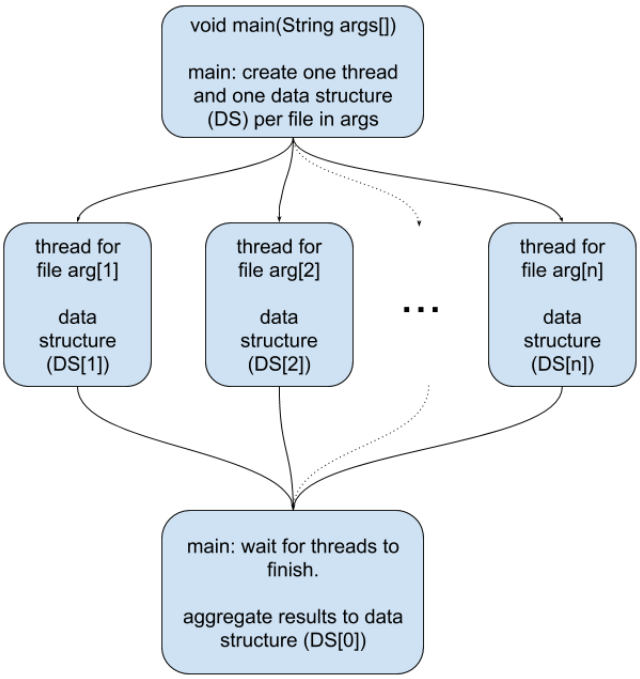

Figure 1: The flow diagram for both Word Count and Inverted Index.

Part 2: Inverted Index with Threads

Repeat the same approach for the Inverted Index. This time, you are asked to find the files the word appears in. Use one thread per file with a data structure passed as argument to that thread. Once the threads are complete, aggregate results from the data structures and compare it to the results of your Inverted Index application without threads.