Project Part 2

Due November 21, 2016 at 11:59PM

The goal of this part is to read a number of news articles, and
1. Create a dictionary of the index terms. Each record in the dictionary will have the term, the number of occurrences of the term in the collection cf, the df, and a pointer to the posting list
2. Create a posting file. Each record in the posting file will have the DocID, and the raw tf of the term in the document. Each posting list should be sorted by DocID.
3. A file of records sorted by DocID containing for each document the DocID, the number of words in the document |D| excluding the stop words, headline, a static snippet (which is the first 40 words from the <TEXT> tag of the article) and a relative path of the document from the context where you are running your programme.
4. A file that contains the total number of words in the collection |C| excluding the stop words.
The program will read the files stored in the docs directory on Blackboard and create the four files above. The docs directory contains subdirectories each of which contains a small number of news articles.
5. Tokens should not consider the stop words as mentioned in the first part of the Assignment after tokenization is done make sure no token contains any special characters (unlike first assignment). You can consider anything except **A-Z a-z 0-9** as special character.
6. Each field in the output document should be separated by a comma. So your content (like tokens, headline or snippet etc) should not contain “comma”. If comma appears in the snippet or in the headline replace comma with a space. ~~If your your snippet sentence becomes very big you can take upto first 40 words of the first sentence in the <TEXT> tag of your article. Snippet should not contain any tags.~~
7. docId will be an integer starting from 1 till the number of documents in the collection.

To tokenize the documents use the rules specified in Part 1 along with the above rules.
The input parameter to the program will be a directory path name where all the files will reside. Input files can reside under subdirectories.
The output files will be:

1. dictionary.**csv**
2. postings.**csv**
3. docsTable.**csv**
4. total.**txt**

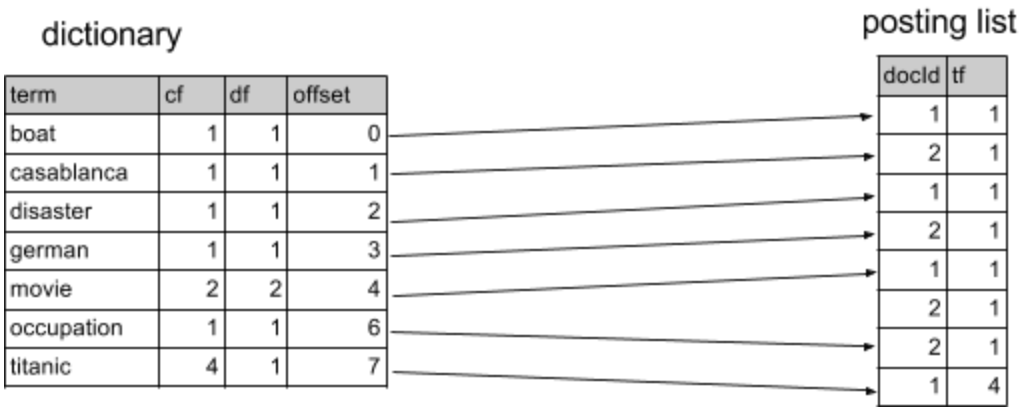
Each field in these files will be separated by a **comma**.

EXAMPLE
There are 2 documents Doc1 and Doc2.
Doc1 headline = Titanic
Doc2 headline = Casablanca

Doc 1 body = Titanic movie. Titanic disaster. Titanic boat.
Doc 2 body = German occupation movie.

Docs Table:

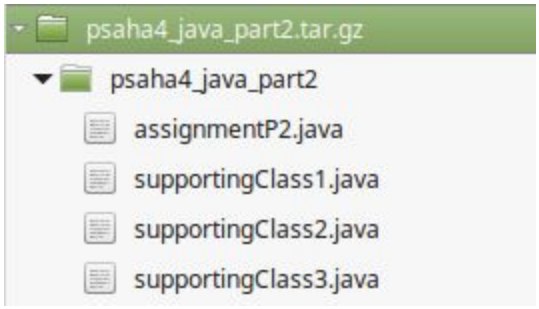
doc number	headline	doc length |D|	snippet	Doc path
1	titanic	7	Titanic Movie	./docs/group1/document.txt
2	casablanca	4	German Occupation movie	./docs/group2/document.xml



Submission directions:

1. You may write the code using C, C++ or Java. Your program should compile on **remote.cs.binghamton.edu** No exceptions. If you have trouble accessing the remote server please contact CS system administrator.
2. Your need to submit a **.tar.gz** file (not only .tar) which should follow the following naming convention. <usrid>_<language>_part2.tar.gz, after unpacking this .tar.gz it should have a directory named <usrid>_<language>_part2. Example: if your user id is psaha4 and you are writing the assignment in java then your submission file name will be **psaha4_java_part2.tar.gz**. For c++ and c developers it will be psaha4_cpp_part2.tar.gz and psaha4_c_part2.tar.gz respectively.
3. Your submission folder should **not** contain input files, object or class files and readme file. Presence of these file will cost you 5 points. It should contain a makefile and source code files.
4. **For Java developers** your entry point class ,which contains the main method, should have the name as “asignmentP2.java”. So make sure your makefile will produce a assignmentP2.class file after compilation and it can be run by the following command **java assignmentP2 <directory path>**

5. All your .java files should be inside a single folder, don't create many packages to place your java files. All source code files should be inside a single directory like below.



6. **<u>For C++ and C developers</u>** please write your makefile such a way that it produces object file with name assignmentP2.obj and it can be run with the below command **./assignmentP2.out <directory path>**
   All source code files should be inside a single directory like below.

7. Your program should generate the below output files. You can hardcode these file names in your source code.
   - 1. dictionary.csv
   - 2. postings.csv
   - 3. docsTable.csv
   - 4. total.txt

   Please make sure, file name and file extension are the same as mentioned. Exception will cost you points.
8. Please ask questions regarding assignment and submission instruction to TA, don't assume anything.