In this assignment you have to process a set of text files to generate data for word autocompletion.

1. First, you have to implement a Spark program to find for each prefix of length 3 or more of each word in a given set of input documents, the 3 most frequently occurring words with that prefix.
   - The output should be lines, with one line for each prefix, with the line containing the 3 most frequent words. See sample output later in this assignment.
   - Note: JavaSparkContext.textFile can take as argument a directory name, in which case it reads all files in the directory, or even patterns (e.g. /home/sudarsha/*.tex, which will read all .tex files).
   - *In case there are multiple words with the same frequency tied at the 3rd spot, pick any of them, and always return 3 words. The application can only show a few words for auto-completion, so there is no point returning more than 3 words.*
   - *As an example, if there are 2 words tied for 1st spot, and 2 words for 2nd spot, you will print the first two words, and any of the 2nd two words*
   - *The 3 words should be sorted in decreasing order of frequency. If there are ties at any position, the words can be printed in any order.*
2. Next, modify the program to consider not just prefixes of words, but also preceding words. For example, if the user has just typed "database" the most commonly occurring words that follow database would probably include "systems", "management", and so on. So a system that uses your programs output can suggest the next word even before the user has typed a single letter. To distinguish this case from prefixes, the first word would be terminated by a $ sign.
3. Now add a combiner, which can reduce the amount of data transferred.
4. Submit the final version which includes both prefixes and preceding words, as well as the combiner.

NOTE 1: Treat any character that is not a letter or number as a space, and map upper case (capital) letters to lower case (small) letters.
NOTE 2: you may be able to find Spark/Hadoop programs for solving this task on the Web; avoid searching for them and do it on your own..

NOTE 3: For preceding words, the 3 character limit does not apply; for example, words such as 'a','an' or 'I' would be considered even though they have less than 3 characters.
NOTE 4: You can assume for simplicity that the number of values for a specific reduce key is small enough to keep in memory; a fully general solution is to get the list at the reducer sorted by (reduce key, word), but that is non-trivial to implement, so don't bother about it.
NOTE 5: You can modify and use existing code for sorting values in a Map, e.g. http://stackoverflow.com/a/2581754

Ex: Assume below are the input Documents.

file1={"She shells shenanigan sherlock sherpa Venn Venkatesh Venkataraman ."}

file2={"She shells sherlock shepherd Venn Venkatesh Venkateshwaran Venkataraman.}

Output for above case:

```
she: she shells sherlock
she$: shells
shel: shells
shell: shells
shells: shells
shells$: sherlock shenanigan
shen: shenanigan
    .. and as above, for all prefixes of shenanigan
shenanigan$: sherlock
shep: shepherd
    .. and as above, for all prefixes of shepherd
shepherd$: venn
sher: sherlock sherpa
sherl: sherlock
    .. and as above, for all prefixes of sherlock
sherlock$: sherpa shepherd
sherp: sherpa
sherpa: sherpa
sherpa$: venn
ven:  venn venkatesh venkataraman
venn:  venn
venn$: venkatesh
venk: venkatesh venkateshwaran venkataraman
venka: venkatesh venkateshwaran venkataraman
venkat: venkatesh venkateshwaran venkataraman
venkate: venkatesh venkateshwaran
     .. and so on ... for all prefixes of all above words, and for all word$ cases
```

Submission guidelines: Submit your .java file(s) along with a README (in case you want to specify anything about your assignment) as a single tar.gz or zip file.

NOTE: Late submission policy: you lose 10% for each day or fraction thereof that you are late. No submissions will be accepted after 29 Sep 10 PM. Do not email your assignments to instructor/TAs if the system is not accepting submissions after the deadline.