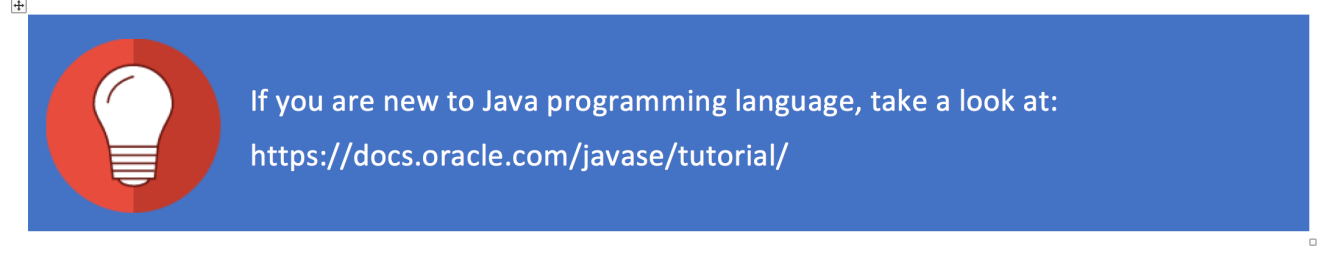
Machine Problem 0: Java Programming Essentials

**1 Overview**

Welcome to the Java Programming Assignment. This assignment evaluates the Java programming skills you will need to work on the upcoming assignment.

**2 Requirements**

This assignment is designed for and will be graded based on **JDK 7**.



**3 Procedures**

**Step 1:**You will need your unique ID to complete this assignment. You can find your unique ID in the PDF document below.

[CourseraIDs.pdf](https://d18ky98rnyall9.cloudfront.net/_897b9123ef4f7f047f3443d37901a435_CourseraIDs.pdf?Expires=1485129600&Signature=ZLet07TgDpBzExxhFszb3zja0H8vNmNtQapbgY8azPg7aacqLrkskJcL8x~SfqoZ-6urO6A9icZj~PRPNSB3Imrm5v34iLxEjGAe8sprZYT~04M0FSYqsIDGAW9Eoa70ydFt6o3fq8H1GVBh4PxXy6BbbA5ijUL1-c3ru7MwIcY_&Key-Pair-Id=APKAJLTNE6QMUY6HBC5A)

**Step 2:** Download the project files from [GitHub](https://github.com/uiuc-srg/cloudapp-mp0).

**Step 3:**Edit the following template file. All you need to edit is the part marked with **TODO**. You will find more information on what to do in the next section.



Don’t change the filename, class name, or the main function.

**Step 4:**Compile the file and run it on the provided input. Use **your own unique user ID** as the first argument. The following is an example of a command to run the application. The command line might need some modifications for your platform:



Make sure the first argument is an exact match with your Coursera user ID; otherwise, your submission might not be graded.

**Step 5:**Save the output in a text file.

**Step 6:**After you are done with the assignment, make a submission with a zip file containing: 1). your code and 2). your output files.

You have to submit both the results and the Java source code. Make sure that the result is generated using your own **user ID**. The Java source code should also be compilable using Oracle JDK 7.

Exercise: **Selective** **Word Count**

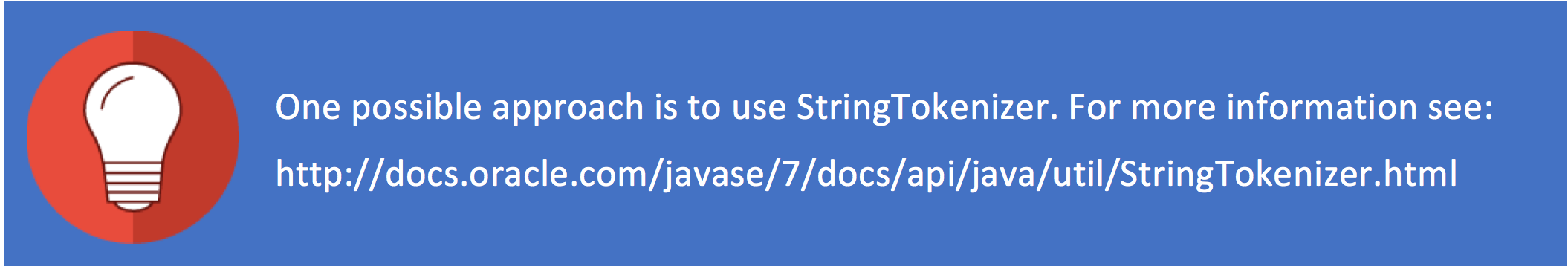
In this exercise, you are to implement an application to find the top **20** words used in Wikipedia titles (provided). To make the implementation easier, we have provided a boilerplate for this exercise in the following file:



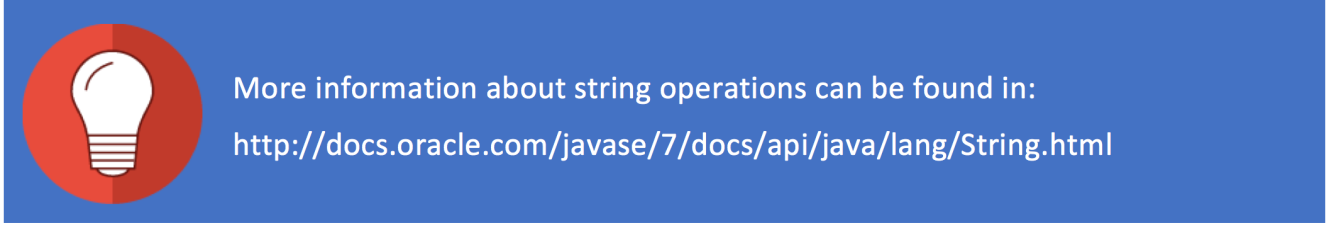
All you need to do is to make necessary changes in the file.

Your application takes a huge list of Wikipedia titles (one in each line) as an input. You need to make some preprocessing on the input, and then return the top **20** words that appear the most in a selection of titles. One possible procedure is the following:

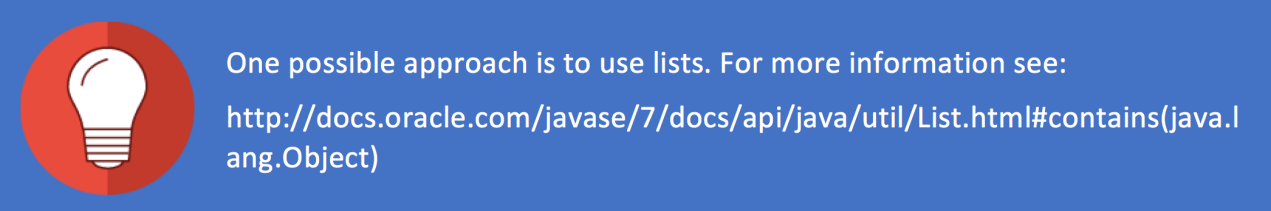
1. Divide each sentence into a list of words using delimiters provided in the “**delimiters**” variable.



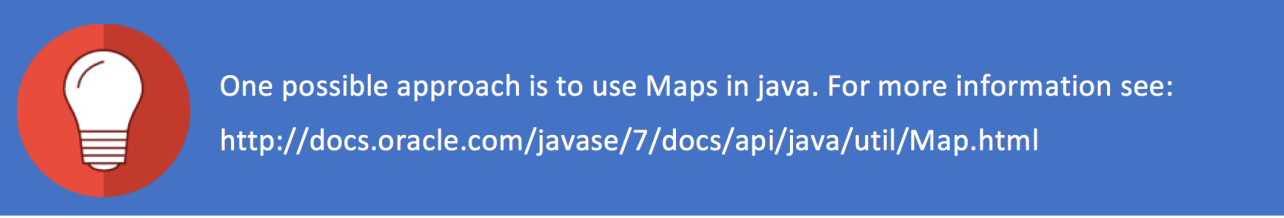
2. Make all the tokens lowercase and remove any tailing and leading spaces.



3. Ignore all common words provided in the “**stopWordsArray”**variable.



4. Keep track of word frequencies. To make the application more interesting, you have to process **only the titles with certain indexes**. These indexes are accessible using the “**getIndexes**” method, which returns an Integer Array with 0-based indexes to the input file. It is possible to have an index appear several times. In this case, just process the index multiple times.



5. Sort the list by frequency in a descending order. If two words have the same number count, use the lexigraphy. For example, the following is a sorted list:



6. Return the top 20 items from the sorted list as a String Array.

Here istheoutput of this application if “**0**” is used for user ID:



Here istheoutput of this application if “**1**” is used for user ID:



Remember, in order to submit the application, **your own unique user ID** should be used as the user ID. Changing the user ID may change the output. **You will submit both your code and output files to Coursera.**

}

}

}