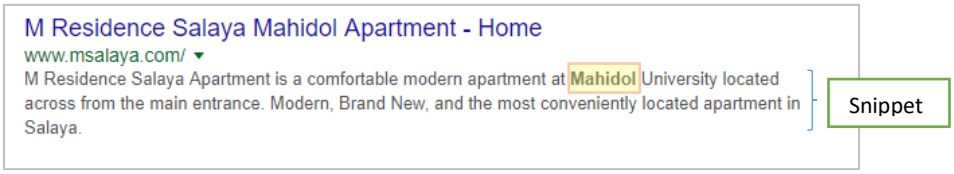
ITCS 209 Object Oriented Programming	Name:	Lab Score	Challenge Bonus	Peer Bonus
	ID:			
	Sec:			

Lab14: Context Searching

Modern search engines such as Google and Microsoft Bing enable preview snippets as part of search results. To generate a snippet, search engines must locate the query word in the document, then pull out the surrounding text to display. Below is an example of a search result, using keyword "mahidol," with a snippet that contains such a keyword.



In this lab, you will be implementing a simple mechanism to generate a snippet when given a keyword as the query, using **sequential** search. The system is ***case-insensitive***, in the sense that every character will be automatically lowercased before processing. A document is defined as a sequence of Word objects. The class Word is provided for you. DO NOT MODIFY Word.java. A Word object contains the following information:

String word: represents the textual content of this word
int position: stores the position of this word that appears in the document.
Word previousWord: stores the reference to the previous Word object; null if it is the first word.
Word nextWord: stores the reference to the next Word object; null if it is the last word.

A query word (i.e. search keyword) has a negative position (since it does not appear in a document). The method compareTo() is already implemented for you, where w1.compareTo(w2) is less/greater than 0 if w1 is lexicographically less/greater than w2. If w1 is lexicographically the same as w2, their positions in the document are then compared. If either w1 or w2 is a query word (whose position is negative), then they are only lexicographically compared. Refer to Word.compareTo() for the logical flow.

In addition, Word.equals() is already overridden for you, where w1.equals(w2) returns true when w1.compareTo(w2) == 0, false otherwise.

We also provide an abstract class ContextSearcher (DO NOT MODIFY ContextSearcher.java) that contains the following:

protected ArrayList<Word> sortedWords: A sorted list of Word objects in the input document.

ContextSearcher(String filename): A constructor that loads the text file (specified by filename), cleans the text, tokenizes and sorts the words, and stores the sorted words in sortedWords. This method is already implemented for you.

public static String getSnippet(Word root, int window): A static method that takes a Word reference ***root***, and an integer ***window*** as input. If root is valid, this method will return a String snippet that contains the word referenced by root, along with window words that come before and after it. This method is already implemented for you.

public abstract String find(Word query, int window): An abstract method whose purpose is to find the first occurrence of the ***query*** word in the document. If found, return the corresponding snippet with the given ***window***, otherwise return null.

Your task is actually very simple. Besides understanding the provided code (code reading is a necessary skill for a good programmer), you must create class **LinearContextSearcher** (in LinearContextSearcher.java) that extends **ContextSearcher**, then implement the necessary constructor, and **find(Word query, int window)** that uses ***linear search*** algorithm discussed in class, by linearly scanning through ***sortedWords***. You must use **Word.equals()** and **Word.compareTo()** whenever you need to check for equality or compare two Word objects. You may use the provided **commons-io-2.6.jar** to facilitate file I/O.

The output from ContextSearcherTester should look like:

```
***************** NORMAL****************
... institution in thailand had its [origin] in the establishment of siriraj ...
# Comparisons (Linear):133

null
# Comparisons (Linear):220

... within large scale implicit social media data [suppawong] tuarob sunghoon lim faculty of information and ...
# Comparisons (Linear):9275

... waits for an [iphone5] in this cold ...
# Comparisons (Linear):5196

null
# Comparisons (Linear):11709

... all characters voice acting and expressions are spot on including [noctis] especially suggest not skipping this
cut scene or this entire ...
# Comparisons (Linear):36881

... cup quest this quest will have us killing off rouge [behemoth] enemy that is level 35 first talk to the man ...
# Comparisons (Linear):8848

... the first true [boss] battle at the ...
# Comparisons (Linear):9687

final fantasy xv faq walkthrough by [codebreak1337] version 424 last updated 2017 01 20 table of contents ...
# Comparisons (Linear):12826

null
# Comparisons (Linear):68864
```

Challenge Bonus (Optional):

Implement **BinaryContextSearcher** that extends **ContextSearcher,** then implement the necessary constructor, and **find(Word query, int window)** that uses ***binary search*** algorithm on **sortedWords**. You must use **Word.equals()** and **Word.compareTo()** whenever you want to check for equality or compare two Word objects.

The output from ContextSearcherTester (Bonus) should look like:

```
********* BONUS for CRAZY PEOPLE************
... institution in thailand had its [origin] in the establishment of siriraj ...
# Comparisons (Binary):12

null
# Comparisons (Binary):16

... within large scale implicit social media data [suppawong] tuarob sunghoon lim faculty of information and ...
# Comparisons (Binary):25

... waits for an [iphone5] in this cold ...
# Comparisons (Binary):22

null
# Comparisons (Binary):26

... all characters voice acting and expressions are spot on including [noctis] especially suggest not skipping this
cut scene or this entire ...
# Comparisons (Binary):124

... cup quest this quest will have us killing off rouge [behemoth] enemy that is level 35 first talk to the man ...
# Comparisons (Binary):29

... the first true [boss] battle at the ...
# Comparisons (Binary):84

final fantasy xv faq walkthrough by [codebreak1337] version 424 last updated 2017 01 20 table of contents ...
# Comparisons (Binary):34

null
# Comparisons (Binary):32
```

Numbers of comparisons do not have to be the same as above, but should be approximately equivalent.

Peer Bonus (Optional):

Sticker Here

Sticker Here

Sticker Here