CSC 483/583: Assignment #1 (75 pts)

Due by 11:59 P.M., September 5
(answer questions in Gradescope, and upload code to GitHub Classroom)

Because the credit for graduate students adds to more than 75 points, graduate students' grades will be normalized at the end to be out of 75. For example, if a graduate student obtains 80 points on this assignment, her final grade will be $80 \times 75/85 = 70.6$. Undergraduate students do not have to solve the problems marked "grad students only". If they do, their grades will not be normalized but will be capped at 75. For example, if an undergraduate student obtains 80 points on this project (by getting some credit on the "grad students only" problem), her final grade will be 75.

Note that the first four problems do not require coding. Only problem 5 requires coding.

Problem 1 (15 points)

Consider these documents:

Doc1 breakthrough drug for schizophrenia

Doc2 new approach for treatment of schizophrenia

Doc3 new hopes for schizophrenia patients

Doc4 new schizophrenia drug

1. Draw the term-document incidence matrix for this document collection.
2. Draw the inverted index representation for this collection, as in Figure 1.3 in IIR.
3. What are the returned results for these queries:
   (a) schizophrenia `AND` drug
   (b) for `AND NOT`(drug `OR` approach)

Problem 2 (20 points)

1. Write out a postings merge algorithm, in the style of Figure 1.6 in IIR, for an x `OR` y query.
2. Write out a postings merge algorithm, in the style of Figure 1.6 in IIR, for an x `AND NOT` y query.

Problem 3 (10 points)

Recommend a query processing order for:

(tangerine `OR` trees) `AND` (marmalade `OR` skies) `AND` (kaleidoscope `OR` eyes)

given the following postings list sizes:

Term	Postings size
eyes	213312
kaleidoscope	46653
marmalade	107913
skies	271658
tangerine	87009
trees	316812

Problem 4 (10 points)

How should the Boolean query x `OR NOT` y be handled? Why is the naive evaluation of this query normally very expensive? Write out a postings merge algorithm that evaluates this query efficiently. Hint: you may want to use one of de Morgan's laws (`https://mathworld.wolfram.com/deMorgansLaws.html` for the latter part: `NOT` (x `OR` y) = (`NOT` x) `AND` (`NOT` y).

Problem 5 (20 points undergrad / 30 grad)

Implement an inverted index that supports Boolean search. Your program must take in one file containing one document per line, in a format close to the one used in Problem 1. For example, the GitHub Classroom unit tests use this file:

```
Doc1    breakthrough drug for schizophrenia
Doc2    new approach for treatment of schizophrenia
```

```
Doc3    new hopes for schizophrenia patients
Doc4    new schizophrenia drug
```

In particular, the format requires:

- One document per line in the input file.
- The first token in each line be the document id. The id is not to be indexed as a term!
- The rest of tokens in a line be all terms appearing in the document. Assume that the text is already tokenized and the tokens are normalized.

To code this problem, you can use Python or Java (or another JVM language such as Scala). You can use data structures available in your programming language of choice, e.g., dictionaries in Python or hash maps in Java/Scala, but you are **not** allowed to use open-source code that implements inverted indices, such as Lucene. You have to implement the inverted index and corresponding search operations from scratch.

The code submitted **must** pass the unit tests in the GitHub Classroom repository to be considered for grading. Please see the submission instructions for GitHub Classroom at the end of this problem.

Please implement the following:

1. **(20 points)** Construct the inverted index and implement support for binary `AND` queries. What does your code return for the file above and the query: schizophrenia `AND` drug?
2. **(GRAD STUDENTS ONLY: 5 points)** Add support for binary `OR` queries. What does your code return for the query: breakthrough `OR` new?
3. **(GRAD STUDENTS ONLY: 5 points)** Add support for queries that combine multiple binary `AND` and `OR` operators, such as: (drug OR treatment) AND schizophrenia. You can choose a default operator priority, e.g., `AND` has a higher priority than `OR`, or use parentheses to make processing order explicit. Support for parentheses is optional. What does your code return for this query?

For all these questions, please make sure **you return the documents in lexicographical order**. For example, if your query returns the first three documents, they should be listed as: `Doc1`, `Doc2`, `Doc3`.

You will implement and submit this problem using GitHub Classroom:

- If you program in Python, click on this link and follow the instructions on the screen: `TBD`

- If you program in Java, click on this link and follow the instructions on the screen: `TBD`

Note: if you are an undergraduate student, you do not have to address the last two questions. Leave the code for these questions as is.

Very important note: **make sure the unit tests in your project pass on GitHub, after you submit your pull request!** If they do not, you will lose 50% of the credit for this problem, i.e., 15 points for graduate students, and 10 points for undergraduates.