INF3800/INF4800

Assignment A

WHAT TO DELIVER:
In this assignment you only have to submit the implementation task. The «on paper»-tasks are obligatory for INF4800 and optional for INF3800. We will still recommend all of you to do all the tasks.

DEADLINE:
The assignment must be submitted, using Devilry, by 06.02.17 at the latest.

On paper (Optional for INF3800)

Exercise 1:

For a conjunctive query, is processing postings lists in order of size guaranteed to be optimal? Explain why it is, or give an example where it isn't.

(Exercise 1.9, page 13)

Exercise 2:

Why are skip pointers not useful for queries of the form x OR y?

(Exercise 2.5, page 38)

Exercise 3:

Consider the following fragment of a positional index with the format:

word: document: <position, position, ...>; document: <position, …>; ...

Gates: 1: <3>; 2: <6>; 3: <2,17>; 4: <1>;
IBM: 4: <3>; 7: <14>;
Microsoft: 1: <1>; 2: <1,21>; 3: <3>; 5: <16,22,51>;

The /k operator, word1 /k word2 finds occurrences of word1 within k words of word2 (on either side), where k is a positive integer argument. Thus k = 1 demands that word1 be adjacent to word2.

a. Describe the set of documents that satisfy the query Gates /2 Microsoft.
b. Describe each set of values for k for which the query Gates /k Microsoft returns a different set of documents as the answer

(Exercise 2.10, page 44)

Exercise 4:
Stemming:
Are the following statements true or false? Justify your answers.

a) In a Boolean retrieval system, stemming never lowers precision.
b) In a Boolean retrieval system, stemming never lowers recall.
c) Stemming increases the size of the vocabulary.
d) Stemming should be invoked at indexing time but not while processing the query.

(This task is taken from the V15 final exam)

In code

In this course you will build simple in-memory search engine. From the course home page you can download a set of Java classes that provide a simple framework for you to code in. During this and future assignments, you will add various parts of code. The purpose of this first assignment is for you to get acquainted with the aforementioned Eclipse[1] project, and to implement the routine that builds a small in-memory inverted index..

- Download the Eclipse project from from the course homepage. Unzip the Java project, load the project into Eclipse, and familiarize yourself with the classes and interfaces.
- `ObligATest.java` contains two simple JUnit tests that you can use as a basis for simple testing.
- The implementation of `InMemoryInvertedIndex.java` is incomplete, and it is your task to finish it. Basically, you should be able to run the JUnit tests in `ObligATest`, which create a small in-memory inverted index for two collections. The results expected from the unit tests will help you to see if you're on the right track. The task is to complete steps 1-4 on pages 6-7 in the textbook. However, steps 1-3 are already done, your job is to finish step 4.

[1] You are free to use another IDE than Eclipse, but then you're on your own.