COSC431 Assignment 1: Write a Search Engine

PREAMBLE **DUE DATE:3 April 2020**

In the first part of this assignment you will write a parser. In the second you will write an indexer, and in the third you will write the search engine. Marks will be given for parts one and three only. If your search engine works then your indexer must work. ***This assignment is worth 20% of your final mark***.

The files you need for this assignment are located in `~andrew/COSC431`. The XML file you are to use is `wsj.xml`, it is the TREC Wall Street Journal collection. Your search engine must run in less than one second per query on the machine we mark it on (including startup, input, output, and shutdown).

PART 1: PARSER 10%

Your program will parse the Wall Street Journal collection of documents marked up in XML. They look roughly like this:

```
<DOC>
<DOCNO>WSJ920102-0154</DOCNO>
<DOCID>920102-0154.</DOCID>
<HL>Notice to Readers</HL>
<DATE>01/02/92</DATE>
<SO>WALL STREET JOURNAL (J), PAGE 1</SO>
<LP>
<P>Today's Wall Street Journal appears in two sections, the
second of which is the year-end review of markets.</P>
<P>Rhythm &amp; Blues, Inc. filed for bankruptcy today.</P>
</LP>
</DOC>
```

Your program will output a stream of tokens suitable for you indexer, one per line with a blank line between each document.

PART 2: INDEXER

Build an inverted-file index of the Wall Street Journal collection. It is obvious to build this on top of your parser from part 1 of this assignment. You should write the index to disk in whatever format you consider appropriate.

PART 3: SEARCH ENGINE 10%

Your search program will read a query from `stdin` and produce results on `stdout`. ***If it does not then you will not get any marks for this part of the assignment*** (we use programs to mark your programs).

Each query will consist of a set of one or more search terms. You are not expected to support phrase, adjacency or proximity searching – in other words, a document-level index is sufficient. You are not expected to support wildcard searching or regular expression searching.

For the query, your program will output a separate line for each ***relevant*** document. Each line is in 2 columns:

```
<docno> <rsv>
```

where `<docno>` is the contents of the `<DOCNO>` element in the Wall Street Journal collection and `<rsv>` is the retrieval status value (relevance score) that your program has assigned to the document. The output must be sorted from highest to lowest rsv.

Your program will be tested on your indexes, but using queries you have not seen.

HOW TO SUBMIT YOUR ASSIGNMENT

Submit your program and your indexes using `submit431`. Include a small `README` explaining how to compile your programs, and how to run your search engine. You will not be marked on the contents of the `README` – I just need to know how to compile and run your program.