Comp 479/6791 **Project 3** **Fall 2023**

Objectives: Refine your indexing procedure. Implement ranking of returns. Test and analyze your system, discuss how your design decisions influence the results.

Due Date: 7.11.2021

Data: Use Reuters21578 for testing and if needed, continue your text scrubbing skills for the final project. Note that the text preprocessing should be secondary in this project.

Description: this project consists of two subprojects that build on each other. Each subproject should be very simple to execute, discuss with your peers and during Lab Q&A if there are any hurdles.

Subproject I: Upgrade your Project 2 Subproject 1 system. In particular:

1. reuse from Project 2 the module that while there are still more documents to be processed, accepts a document as a list of tokens (omit punctuation) and outputs term-docID pairs. Instead of appending new term-docID pairings to a list, make sure you directly append the docID to the postings list for the term. You may use a hash table. No boxes required.
   - (a) compare timing of this SPIMI inspired procedure with the naive indexer **(for 10000 term-docID pairings).**
   - (b) compile an inverted index for Reuters21578 without using any compression techniques

docID hint: Use the NEWID values from the Reuters corpus to make your retrieval comparable.

Subproject II: Convert your indexer into a probabilistic search engine

1. using the assumptions made in Chapter 11 (not covered in class) about independence of terms and documents etc. and
2. using the BM25 formula (p233 (11.32) in Chapter 11.4.3),
3. rank the documents your index returns and
4. for a given query, return a ranked list of results.

Notes: experiment with different values for the parameters k_1 and b as described in the textbook.

Test queries:

1. design **four** test queries:
   - (a) a single keyword query. Compare results for the same queries of Subproject I with the results for your Naïve indexer from Project 2
   - (b) a multiple keyword query for Subproject I that returns documents containing all the keywords (AND) for unranked retrieval
   - (c) a multiple keywords query returning documents containing at least one keyword (OR), where documents are ordered by how many of the keywords they contain. This does not specify a unique ordering, only a partial ordering)
   - (d) a query consisting of several keywords for ranking with BM25
2. run your **four** test queries to showcase your code for the Demo file and comment on the results in your report

Deliverables:

1. individual project
2. well documented code
3. well documented sample runs for your queries on the information needs:
   (a) *Democrats' welfare and healthcare reform policies*
   (b) *Drug company bankruptcies*
   (c) *George Bush*
4. any additional testing or aborted design ideas that show off particular aspects of your project
5. a project report that summarizes your approach, illustrates your design and discusses what you have learned from the project. Note that an analysis of the results of your sample runs has to be included in the report

Submissions: submit on Moodle

Marks:

Subproject 1	3pt	Attr 1
Final inverted index	0.5 pt	Attr 1
Subproject II (Ranking)	3pts	Attr 1
Multiple keyword AND queries **test and development**	1pts	Attr 1
Multiple keyword OR queries **test and development**	1pts	Attr 1
Project report	1pt	Attr 1,6
Demo file	0.5pts	Attr 1