Information Retrieval Lab. (CSD358)

Assignment 1

Instructions

- Upload a zipfile containing your code, output screenshots and corpus.
- Properly **document/comment** your code.
- You can do the assignment in group of 3 students.
- There should be only one submission from each group.

Due Date and Time: 10:00 pm, Sept. 22, 2025

Submitting this Assignment

You will submit (upload) this assignment in Blackboard. **Email submissions will not be accepted.**
Please also submit the github link of your assignment. And, please keep on saving your github code 2-3 times per hour.

Grading Criteria: This assignment has 7 + 3(for novelty) = 10 marks.

Vector Space Model

In this assignment, you will be implementing ranked retrieval using vector space model. To implement the VSM, you may choose to implement your dictionary and postings lists in the following format. The only difference between this format and that in the textbook, is that you encode term frequencies in the postings for the purpose of computing tf×idf. The tuple in each posting represents (doc ID, term freq).

Term	doc freq (df)	→	postings lists
Ambitious	5	→	(1, 5)→ (7,2) → (21, 7) → ...
...	...		...

In addition to the standard dictionary and postings file, you will need to store information at indexing time about the document length, in order to do document normalization. In the textbook this is referred to as Length[N]. You may store this information with the postings, dictionary or as a separate file.

In the searching step, you will need to rank documents by cosine similarity based on tf×idf. In terms of SMART notation of ddd.qqq, you will need to implement the lnc.ltc ranking scheme (i.e., log tf and idf with cosine normalization for queries documents, and log tf, cosine normalization but no idf for documents. Compute cosine similarity between the query and each document, with the weights follow the tf×idf calculation, where term freq = 1 + log(tf) and inverse document frequency idf = log(N/df) (for queries). That is,

tf-idf = (1 + log(tf)) * log(N/df).

It's suggested that you use log base 10 for your logarithm calculations. The queries we provide are free text queries, i.e., you don't need to use query operators like AND, OR, NOT and parentheses. These free text queries are similar to those you type in a web search engine's search bar.

Your searcher should output a list of up to 10 most relevant (less if there are fewer than ten documents that have matching stems to the query) docIDs in response to the query. These documents need to be ordered by relevance, with the first document being most relevant. For those with marked with the same relevance, further sort them by the increasing order of the docIDs. Additionally, use Soundex algorithm for spelling matching of names.