# Task2 Implementation:

Firstly, it reads the unigram inverted index from the “file.out” that I get from HW3.

Then, it reads the document length /token count of each doc from “tokenPerDoc.out” which again I get from HW3.

It reads the queries from “queries.txt” , converts it to lower case (Same operation as we done when indexing) and assigns queryId to each query and then proceeds:

I am using tf.idf before applying Vector Space Model (Cosine Similarity)

## Query: -

For each query term the tf is the no of times it appears in the given query

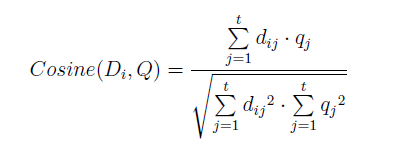
Idf =1

* Making a queryVector which is (1\*t), t is no of terms in indexer and assigns tf to the query terms in query, otherwise 0.

LinkedHashMap<String,Double> queryVector

### Document: -

For each document, calculating score based on Cosine Similarity formula,



Where, dij = (normalized tf of query term (j) which is tf/tokenCount in Di) \* (idf of that document)

idf =1 + log(no of docs/ df)

Basically, finding normalized tf and multiplying it with idf to get dij and then multiplying it qj

* Normalized based on document length so that this doesn’t prefer longer documents.
* Using idf which means rarer terms give higher contribution to the total score.

Finally, sorting based on score and writing to the .csv files in folder Task2/