**Implementation Task1:**

The source code for Lucene implementation was modified to provide the following functionalities:

* Read the queries from a text file
* Path for the corpus and index creation was hardcoded as it will not change in this assignment.
* Used SimpleAnalyzer as the analyser.
* Changed the results from top 3 to top 100.
* Stored the results in table format in a file.

**Implementation Task2:**

Retrieve the inverted index for the corpus from the file created in HW3.

Retrieve the file containing document tokens dictionary created in HW3.

Retrieve the document Id dictionary created in HW3.

Retrieve the file containing document Id and their terms created in HW3.

Read the file containing queries and put them into a dictionary.

* For each query,

Calculate the tf.idf of query and store it.

(Calculate the weight of each term in query) \* (idf for each term)

* For each document,

Calculate the tf.idf for each query term and store it.

For each term in the query, get the normalized term frequency

This will ensure proper weightage is given to each term and prevent lengthier documents being given preference.

* Calculate the magnitude of the query
* Calculate the magnitude of the document
* Calculate the vector cosine similarity scores for all the documents w.r.t the query.

Accumulate scores for each term in the query on the cosine similarity scores list.

* The cosine similarity score is calculated for each document as:

(Dot product of tf.idf of query and tf.idf of doc)/ (Magnitude of query) \*(Magnitude of doc)

* Sort the documents according to their scores.

Write the top 100 documents to a file.