**Lab 3.** In this lab, you are going to compare the TF-IDF algorithm to the Okapi BM25 algorithm. First, compute the 20 most relevant documents for each query using the Okapi BM25 algorithm. Use k1=1.2, b=0.75, and k2=100. Save the result in a hash map. The key will be the query number and the value will be an array list of the IDs of the documents that are relevant in ranked order.

Next, examine the file human\_judgement.txt. It contains the query number, document number, and degree of relevance. For this lab, assume that a number that is equal to 1, 2, or 3 means relevant and you can ignore entries with any other degree of relevance.

Your job is to evaluate the MAP score for the cosine similarity algorithm and Okapi BM25 algorithm and compare the results. Remember to use only the top 20 documents that are returned by the system in ranked order. In order to keep the running time manageable, compute the MAP score by taking the average over only the first 20 queries (i.e, query with ID 1 up to query with ID 20). Expected output is shown below.

Cosine MAP = 0.29881118219108416

Okapi MAP = 0.29305451674068533

**Design Considerations:**

Create class Labs.Lab3.

* main method:
  + Reads the document and query DocumentCollection objects and the HashMap with cosine results (created in Lab2).
  + Calculates the top 20 documents for each query using the Okapi distance and stores the result in a HashMap:

HashMap<Integer, ArrayList<Integer>> okapiResults

* + Reads the data from the human\_judgement.txt file and stores the result in the humanJudgement variable. The variable contains the IDs of the relevant documents for each query.

HashMap<Integer, ArrayList<Integer>> humanJudgement

* + Calls the computeMap function twice.

System.out.println("Cosine MAP = " +

computeMAP(humanJudgement, cosineResults));

System.out.println("Okapi MAP = " +

computeMAP(humanJudgement, okapiResults));

* computeMAP method. Computes the MAP score between the two input hash maps using only the first 20 queries.

Create the DocumentClasses.OkapiDistance class. Implements DocumentDistance and overrides the findDistance method. Note that the method uses the **raw** (and not normalized) frequencies. Also, note that second part of the Okapi BM25 formula uses the raw frequency of the matching word in the document, while the third part of the formula uses the raw frequency of the matching word in the query.