Information Retrieval

Assignment 3
Team 3:
Shayan Amani
Soheil Gharatappeh

1. Please find our codes for assignment 3 in the following GitHub url:
https://github.com/SHi-ON/InfoRet/tree/master/src/main/java/edu/unh/cs/ir/pa3
You can find the information regarding to install the project on your system in project's README file.
https://github.com/SHi-ON/InfoRet
You can see the results from consul and they are going to be printed into a file named run_file. There are two options to change Tfidf. You can change the similarityBase(), or TFTIDSimilarity(). Because of some confusion we had about which one to choose, and if choose one, which parameters to change, we couldn't manage to change the search engine. However, we understand where we need to change those class, and we got some ideas about how these classes/methods work.

We used TFIDFSimilarity() methods, and override them to implement problem requirements. An input argument for IndexSearcher3() can specify which variant is aimed for TFIDF computation. In order to code for lnc.ltn, we used queryNorm()'s argument to keep the summation of squared weights variable, which is defined by variable sumSW, then by overriding tf() method we wrote the logarithmic variant of TF and also for IDF.

We have difficulties to figure out how we should change the similarities to implement ddd and qqq pairs of SMART notation using TFIDFSimilarity or the other similarities class, similarityBase(). In fact, the root of our confusion was where we couldn't understand that which method/argument belongs to the document vector, and which one belongs to the query one.

1.a. Which of these variants perform best?
Among these three methods, lnc.ltn makes the most sense. What bnn.bnn, does not consider is the fact that a document which 10 times more occurrence of a word is not usually ten times more important that the other one. Probably, it's just a long document. Or, it can be a document that is appended to itself multiple time. But, what makes anc.apc make less sense is the fact that in augmented weighting, terms that don't have the max tf, usually takes a weighting so close to 0.5. So, in if we have a long document, and one of the terms have an exceptionally high tf, this ruins the ranking. However, by using p(prob idf) in apc part of anc.apc, it's tried to somehow dampen the effect of terms with high tf.

1.b. Do they perform better or worse than Luecene's default ranking model?
Since Luecen's default ranking model does more advanced math, it considers more nonlinear effects, which makes the results more close to what a user might expect.

2. lnc.ltc should have the closest results to the Lucene's default ranking function.