CSC 849: Information Retrieval
Assignment 2
Positional Inverted Index & Free-text Queries with Proximity Operator
100 Points

This assignment consists of three parts.

Part 1: Positional Inverted Index Construction
Extend your program from HW1 to create a **positional** inverted index for a given set of documents.

Part 2: Query Evaluation
Write a program that evaluates free-text queries with proximity operators using a tf.idf scoring function to generate a ranked result list of documents for each query. Here are the details:

1. Make sure that the pre-processing steps (various normalizations) that were applied to the documents, are also applied to the query.

2. Proximity operator: Your query evaluation program should be able to evaluate queries containing proximity operators. Use the following syntax for proximity operator:
   n(t1, t2)
   where n is the proximity window for terms t1 and t2. That is, terms t1 and t2 can have at most n terms between them in the document. The order in which the terms t1 and t2 appear in the document is important. A document containing t2 before t1 is not a valid match, unless there is another occurrence of t2 after t1 and the number of terms between them is at most n.

3. A query can contain any combination of terms with proximity operator and free terms. Thus your program should be able to parse and evaluate all of the following queries:
   a. nexus like love happy
   b. asus repair
   c. 0(touch screen) fix repair
   d. 1(great tablet) 2(tablet fast)
   e. tablet

   Note about term order: The query terms are **not** ordered. Thus queries: *charger 2(nexus tablet)* and *2(nexus tablet) charger* are equivalent. However, the order of terms inside the proximity operator is significant, as explained above.

4. Use the following tf.idf scoring function to assign a relevance score to each document for a given query, and rank the documents based on the score (descending order).

$$score(q,d) = \sum_{t \in q \cap d} w_{t,d}$$

$$where:\ w_{t,d} = \left[1 + log_{10}\left(tf_{t,d}\right)\right] * log_{10}\frac{N}{df_t}$$

For queries with proximity operators the evaluation consists of two phases:

1. Filtering: In this step the subset of documents that meet all the proximity criterion are identified.
2. Scoring: The documents identified by the filtering step are then scored using the above function to obtain the final ranked result list. The terms inside the proximity operator are considered as individual terms for the purpose of scoring function.

Part 3: Using your Search Engine

The Parts 1 and 2, together, make a simple search engine. In this last part we will make use of this search engine.

a. Use your program developed for Part 1 to create a positional inverted index for the collection of documents in *documents.txt* which is on ilearn.
b. Use your program developed for Part 2 along with the inverted index of *documents.txt* to evaluate the five queries in Part 2.3

Finally, upload to ilearn four files:

1. (10 Points) Program from Part 1,
2. (40 Points) Program from Part 2,
3. (20 Points) The inverted index from Part 3a. That is, a file containing the dictionary and the posting lists.
4. (20 Points) The query results from Part 3b.

You may use any of the following programming languages: C++, Java, Python/Perl, for this assignment. If you wish to use a different programming language, please contact the course instructor at ak@sfsu.edu.

Important note: Your programs must be well documented. (10 Points).

Good luck and start early!