IR Assignment 4

Aakash Khadka, Gowtham Premkumar, Max Neubauer

4.a)
In order to solve the first assignment we built a CustomAnalyser class with a StandardTokenizer and LowercaseFilter in order to tokenize the documents. Then we use a TreeSet and a TreeMap to keep a sorted Collection of our terms and TF-IDF scores. Then we calculate the idf value for each term. Then we loop through each document and calculate the tf values of each term in that document. After assembling the tf values of a document we calculate the TF-IDF values and store them in the vectors. The TF-IDF vectors are stored in a map. In order to calculate the euclidean distance we implemented a separate function that iterates over the vector and applies the sum of the squared differences. For the dot product we iterate over the vectors, multiply them and add the products. To calculate the cosine similarity we do the dot product of the vectors and divide with the product of the vectors length. Then we display the solutions for document 0 and 1.

```
Euclidean Distance: 1.2908633236550593
Dot Product: 0.12162718980527158
Cosine Similarity: 0.16475679254915546
```

4.b)
In order to solve the second part we use a query object to parse a string query using StandardAnalyzer. Then we create the indexSearcher from the given documents. In order to use the BM25 or Vector Space Model we set the corresponding similarity class in the indexSearcher. Then we search and retrieve the top 5 documents for the given query. After that we print the retrieved documents. For the Vector Space Model we use the ClassicSimilarity class instead of the TFIDFSimilarity because it inherits from TFIDFSimilarity and is not abstract.

```
Use Vector Space Model
3.000619 : She is a sunny girl.
0.99381393 : Today is sunny.
0.99381393 : Sunny Berlin!
0.97753894 : She is in Berlin today.
Use BM25 Model
2.6349254 : She is a sunny girl.
0.8236318 : She is in Berlin today.
0.59518534 : Today is sunny.
0.59518534 : Sunny Berlin!
```