ASSIGNMENT-1
README

- **Methodology for Question 1:**

1. Firstly the function “readfiles()” reads all the 20,000 files data, while reading each file I excluded the metadata from by skipping first 15 lines of every file.
2. The readfiles() function reads the entire data in a file (excluding the metadata) sends the data read to the preprocessing function “preprocess()” which returns the pre-processed data.
3. After receiving the pre-processed data we word tokenize it and check for every word whether it is present in the dictionary (contains posting lists for every term) or not. If not present, we create a key for that term in the dictionary and create a list with that respective filename in which the term is present.
4. If the key is already present we append the filename in which the key is present to the already existing list in the dictionary.
5. I have implemented the “AND” ,”OR”, “NOT” functions separately.
6. So when the user enters the query (operators should be present in caps) we pre-process the query and retrieve the required results such as, i) Number of docs retrieved ii) Number of comparisons iii) List of docs retrieved

- **Methodology for Question 2:**

1. The function “readfiles2()” reads every file one-by-one from the directories pre-process it.
2. After preprocessing we build a positional dictionary by maintaining a counter variable which stores the positions of words and append it to the list which contains filename and position of word in that file.
3. If the word is absent in the dictionary we create a new entry for that word which contains the total occurrences of that word in entire directories and position of that word in corresponding files.
4. Now for the given words in the query we compare the relative positions of every two words in the query with that of positions list.if one such position is encountered we break out from the loops. When a document satisfies the count (it should be (n)*(n+1)/2) then that document is added to final answer list of docs.

- **Preprocessing Steps:**

1. Removal of Stopwords. **#avoided in phrase query**
2. Case conversion (to lower case).
3. Eliminating words with length less than 3. **#avoided in phrase query**
4. Lemmatization. **#avoided in phrase query**
5. Eliminating multiple lines gap.
6. Removing special characters. **#avoided in phrase query**
7. Removing multiple spaces between two words.
8. Removing alpha-numeric words. (words containing numericals in them) **#avoided in phrase query**