Report

1) To start with, we need to import the necessary libraries that will help us to extract and preprocess the data. Once the libraries are imported, we will print the initial contents of the file to get an understanding of the data. Next, we will use Beautiful Soup, a library in Python, to extract the relevant text from the file.

Moving on to the preprocessing part, we will first convert the text to lowercase to ensure consistency. Next, we will remove the stop words which are common words that do not add much value to the overall meaning of the text. After that, we will perform tokenization which involves breaking down the text into smaller units such as words or phrases. Finally, we will remove the punctuations and any extra blank spaces to get a clean and structured text.

Once all the preprocessing steps are completed, we will print the final contents of the first 5 files, after each pre-processing step.

2) For this question, firstly we will create an unigram inverted index, which is quite easy. For this case, we created a dictionary with keys as the words and values as the list of documents that contains those words. The words are selected via .split from the pre-processed documents.

The, we later on used the pickle to save the dictionary, and that makes it easy to load at any given moment of time. And then after it, to work with AND, OR and other such boolean functions, we used sets of those results for each query, and then applied those boolean operations on those queries.

To support, the given format of taking the input and printing the output,

We used the format that is mentioned in the question itself, by taking the correct way of taking input and printing the results. The results can be explained based on how we are processing these queries.

3) To start with, we import the necessary libraries that will help us preprocess the input and create bigram and positional inverted indexes. We use the preprocessed data from Question-1, combine two words together (separated by a space) and add them to the dictionary. If the word is already present in the dictionary, we append the file_name to the existing value of the word in the dictionary. If not, we make a new list containing the file_name. Here, the key is the combined words, and the value is the list of files that contain the word. Once the bigram inverted index is created, we save the dictionary using the pickle library.

For creating the positional inverted index, we make a new dictionary called "index" where the key will be the word, and the value will be the dictionary containing a list of positions in the file where the word is found. We traverse all the data present in all the preprocessed files, line by line. We remove all the useless blank spaces, and convert the line to a list of words using the split() method. We then traverse the list of words and keep adding the position of the word in our dictionary along with its file name. The resulting dictionary is our Positional Inverted Index, and we save it using the pickle library as well.

Finally, we preprocess the input in the same way as we did in Question-1 by converting the text to lowercase, removing stopwords, performing tokenization, removing punctuations and blank spaces to obtain a clean and structured text.