Members - Prakhar Kumar and Meenal Jain.

Requirements to run the file:

1. Python3 with libraries like numpy, nltk etc..
2. The Jupyter notebook library needs to be installed to open an ipynb file.

Preprocessing:

Q1:

1. We only kept alphabets and numbers and removed everything else.
2. We converted text to lowercase.
3. We removed stop words.
4. We lemmatized each word in text.

Q2: (required preprocessing steps are done)

1. Converted text to lowercase.
2. Tokenized and removed stopwords.
3. Punctuation removed.
4. Tokens whose length is <= 2 are removed.
5. Instead of just storing positional indexes for a word for a given document, we also store additional extra information (to help in faster processing) like a list of documents.
6. Assuming the phrase query is ordered i.e if query is A B then it is assumed that we only have to return the documents in which A comes before B and A and B are adjacent.

Assumptions:

1. We are not expecting any quer of the form NOT x. As we were not asked to code for such a query. We were only told to do it for AND NOT, OR NOT.
2. Input and operators are taken as input separately. Operators taken as input are to be separated by ‘,’ Input words are separated by a single space.

Methodology:

Index structure for Q1:

{
“Word”: [1, 3] // word is key, posting list is value.
}

Index structure for Q2:

{
“Word”:
    {
        “docList”: [1,2,3] // documents in which this word is present.
        “docId”: [33, 45] // positions in document with id “docId”
    }
}

For Q1 and Q2 some steps are same:

1. Preprocess files are required.
2. Loop through each file:
3. Loop through each word in file:
4. Insert the word in the dictionary if it is not already there.
5. For the word add the file index in this word's posting list.

For querying for Q1:

1. Read input and operator as defined in the assumptions step.
2. Process from left to right.
3. Take two operands and one operator. Perform the operation as defined by the operator.
4. Store the result of this operation and move forward to the next operand.
5. Take another operator and perform the required operation with result and this operands posting list.
6. Repeat 4-5 until all operands(words) are done.

For querying for Q2:
Taking input is the same as Q1 and processing is from left to right.

1. Two operands are taken and their posting list is obtained from the index.
2. We find common documents in which both words occur.
3. Then we check if the words are adjacent to each other in the document.
4. Result is stored.
5. Next two adjacent pairs are taken and the result is stored and merged with the previous result.
6. Repeat 4-5 until all words are done.