INFORMATION RETRIEVAL ASSIGNMENT - 1

Team members

Harjeet Singh Yadav Varun Parashar
2020561 **2020482**

Part 1)

Relevant Text Extraction:-

We have used the regEx library in python to extract the file's relevant data.
To extract the data between the <TITTLE> AND <TEXT> tags, we have used the following regEx.

```
"<" + tag2 + ">(.*?)</" + tag2 + ">"
```

Then we concatenated both the strings to form the file string and wrote back to the same file.

Preprocessing:-

Step 1:- lowering the data to lowercase to maintain uniformity.

Step 2:- Used the NLTK library's word tokeniser to tokenise the string.

Step 3:- Used the NLTK library to remove the stopwords and punctuations from the tokens.

Step 4:- Used the regEx (**[^\w\s]**) to remove the chars other than alphabets and numbers. E.g. **removed -, ., /, \, | etc**.

Step 5:- Finally, join the tokens to form a string and write back to the same file.

Assumption:- No emojis are present in the files

With the above steps, we extracted the relevant information from the files and preprocessed the data.

Part 2)

Unigram Inverted Index

To form the unigram Inverted Index, we used a dictionary of lists to store the postings lists. The key of the dictionary is the words or tokens, and the value is a list of files containing that word. Posting lists contain the file ids in ascending order.

We used the standard boolean merging techniques to merge the two posting lists.
We have used standard AND, OR and NOT merging techniques.

For boolean operations like T1 AND NOT T2 & T1 OR NOT T2, we used an optimised approach to minimise the number of comparisons.
Instead of doing NOT(T2) first and then performing the AND operation, we performed the T1 AND NOT T2 operation in one go using the modified AND technique.
Similarly, for T1 OR NOT T2, we have used AND NOT technique to minimise the number of comparisons.
Code is present in the code files.

Assumptions:-
We have assumed that no. of operations will be compatible with the no of tokens after preprocessing.

Evaluating the boolean queries:-

We have ignored the precedence order of the boolean operations and simply processed the queries from left to right, as mentioned in the google classroom.

Saving the Inverted Index:-

We used the pickle library from python to dump the dictionary (Inverted index) to a file using the dump method.
And used the load file to load the data into the file.

Part 3)

Bigram and Positional Inverted Index

To form the bigram Inverted Index, we used a dictionary of lists to store the postings lists. The dictionary's key is the bigrams (spaces separated between two words), and the value is a list of files containing that bigram. We created bigrams for all adjacent words in the preprocessed data.
Posting lists contain the file ids in ascending order.

Positional Inverted Index

To form the Positional Inverted Index, we used a dictionary of dictionaries to store the postings lists. The key to the dictionary is the words or tokens. Each file id is the key to the dictionary of each word. The value of the dictionary of file ids is a list containing the positions where the word occurred in the file.
The positions in which the words occur are stored in ascending order for each file.

Evaluating the queries using bigrams:-

We process our phrase query and create a list of bigrams from it. We evaluate the files where each bigram exists and store them in a list. We perform AND operation on the lists to find the most relevant files and have a higher chance of containing the phrase.

Evaluating the queries using the inverted index:-

We process our phrase query and create a list of words or tokens from it. We take the dictionaries of files containing each word and perform AND operation on all of them, to get the files that contain all of these words. This is our list of files that might contain the phrase.
For each file, we find the positions of each word of the phrase and see if any match. If they match, then we add that file to our answer.

Saving the bigram inverted index and positional Inverted Index:-

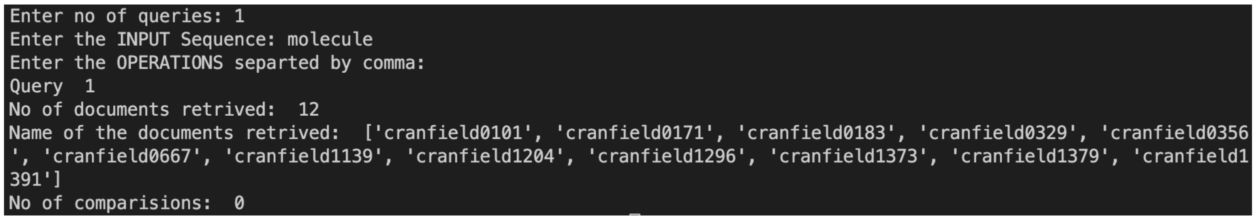
We used the pickle library from python to dump the dictionaries to a file using the dump method.
And used the load file to load the data into the file.

Assumptions:-

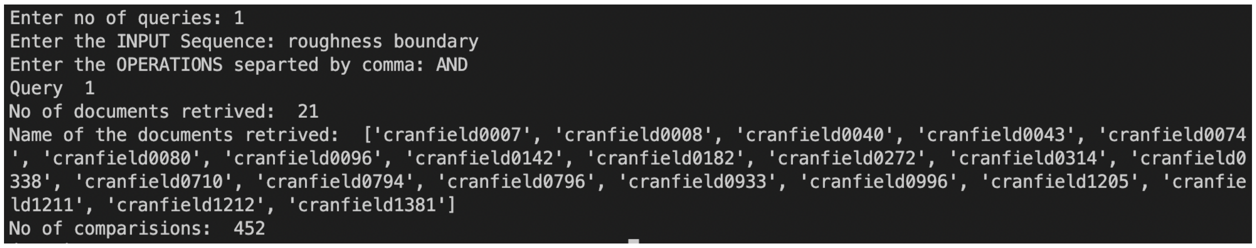
All the queries will be preprocessed in parts 2 and 3.
I.e. all the stopwords will be removed from the query.

Results
Bigram might give false positives as the bigram may exist at different places in the same file. But the positional index will consider the positions of the individual tokens. Hence, there will be no false positive cases.

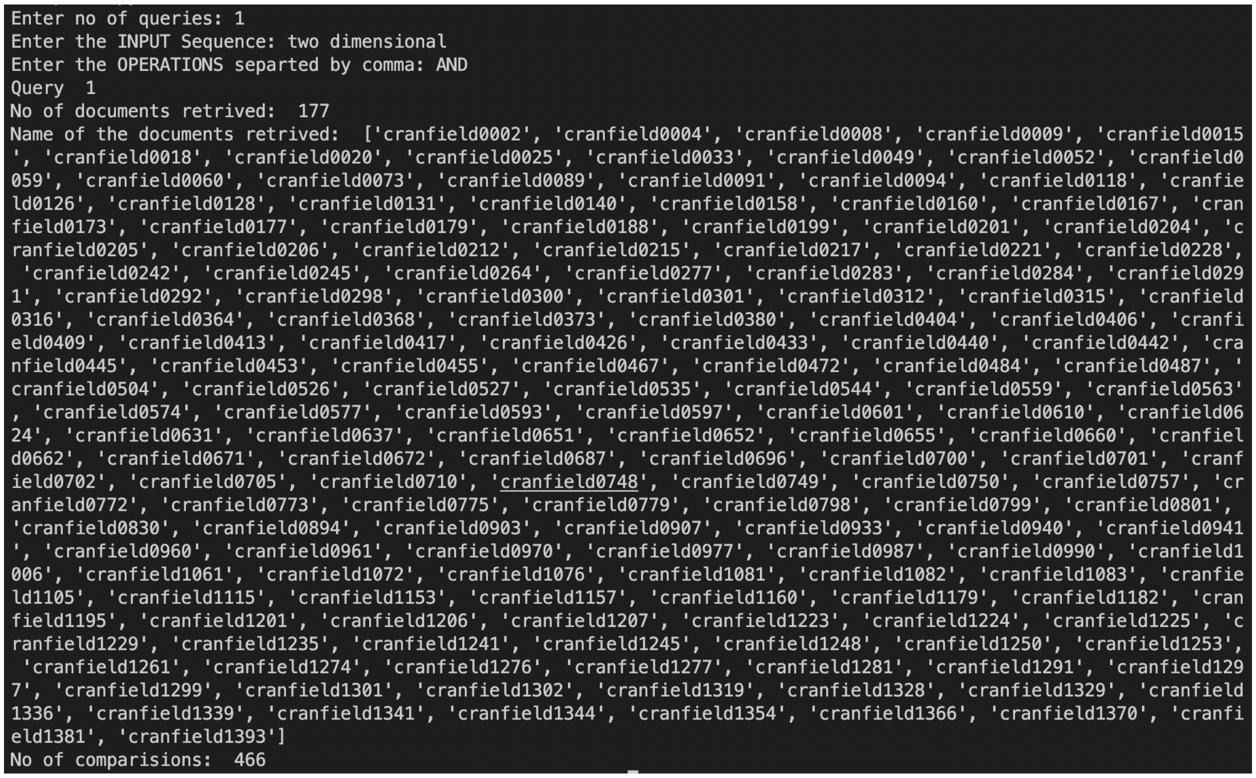
the Results of part-2 use operations



```
Enter no of queries: 1
Enter the INPUT Sequence: molecule
Enter the OPERATIONS separted by comma:
Query  1
No of documents retrived:  12
Name of the documents retrived:  ['cranfield0101', 'cranfield0171', 'cranfield0183', 'cranfield0329', 'cranfield0356
', 'cranfield0667', 'cranfield1139', 'cranfield1204', 'cranfield1296', 'cranfield1373', 'cranfield1379', 'cranfield1
391']
No of comparisions:  0
```

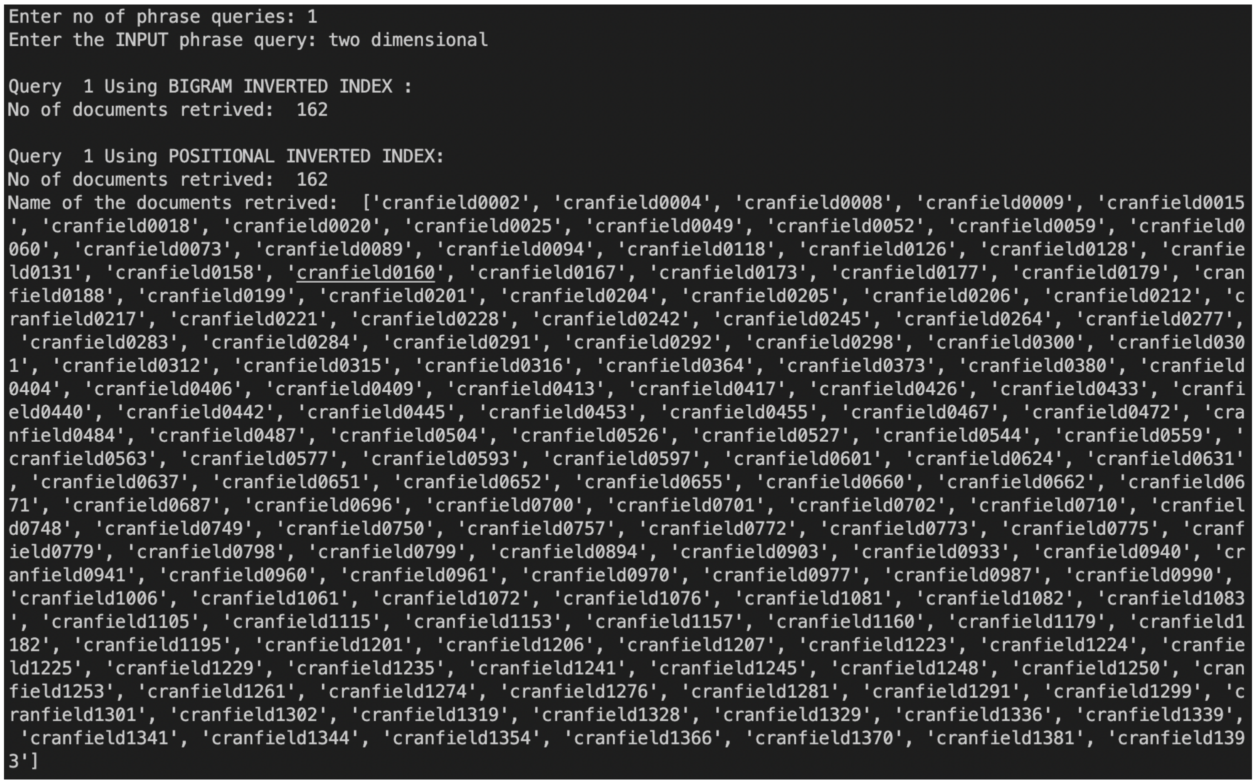


```
Enter no of queries: 1
Enter the INPUT Sequence: roughness boundary
Enter the OPERATIONS separted by comma: AND
Query  1
No of documents retrived:  21
Name of the documents retrived:  ['cranfield0007', 'cranfield0008', 'cranfield0040', 'cranfield0043', 'cranfield0074
', 'cranfield0080', 'cranfield0096', 'cranfield0142', 'cranfield0182', 'cranfield0272', 'cranfield0314', 'cranfield0
338', 'cranfield0710', 'cranfield0794', 'cranfield0796', 'cranfield0933', 'cranfield0996', 'cranfield1205', 'cranfie
ld1211', 'cranfield1212', 'cranfield1381']
No of comparisions:  452
```



```
Enter no of queries: 1
Enter the INPUT Sequence: two dimensional
Enter the OPERATIONS separted by comma: AND
Query  1
No of documents retrived:  177
Name of the documents retrived:  ['cranfield0002', 'cranfield0004', 'cranfield0008', 'cranfield0009', 'cranfield0015
', 'cranfield0018', 'cranfield0020', 'cranfield0025', 'cranfield0033', 'cranfield0049', 'cranfield0052', 'cranfield0
059', 'cranfield0060', 'cranfield0073', 'cranfield0089', 'cranfield0091', 'cranfield0094', 'cranfield0118', 'cranfie
ld0126', 'cranfield0128', 'cranfield0131', 'cranfield0140', 'cranfield0158', 'cranfield0160', 'cranfield0167', 'cran
field0173', 'cranfield0177', 'cranfield0179', 'cranfield0188', 'cranfield0199', 'cranfield0201', 'cranfield0204', 'c
ranfield0205', 'cranfield0206', 'cranfield0212', 'cranfield0215', 'cranfield0217', 'cranfield0221', 'cranfield0228',
 'cranfield0242', 'cranfield0245', 'cranfield0264', 'cranfield0277', 'cranfield0283', 'cranfield0284', 'cranfield029
1', 'cranfield0292', 'cranfield0298', 'cranfield0300', 'cranfield0301', 'cranfield0312', 'cranfield0315', 'cranfield
0316', 'cranfield0364', 'cranfield0368', 'cranfield0373', 'cranfield0380', 'cranfield0404', 'cranfield0406', 'cranfi
eld0409', 'cranfield0413', 'cranfield0417', 'cranfield0426', 'cranfield0433', 'cranfield0440', 'cranfield0442', 'cra
nfield0445', 'cranfield0453', 'cranfield0455', 'cranfield0467', 'cranfield0472', 'cranfield0484', 'cranfield0487', '
cranfield0504', 'cranfield0526', 'cranfield0527', 'cranfield0535', 'cranfield0544', 'cranfield0559', 'cranfield0563'
, 'cranfield0574', 'cranfield0577', 'cranfield0593', 'cranfield0597', 'cranfield0601', 'cranfield0610', 'cranfield06
24', 'cranfield0631', 'cranfield0637', 'cranfield0651', 'cranfield0652', 'cranfield0655', 'cranfield0660', 'cranfiel
d0662', 'cranfield0671', 'cranfield0672', 'cranfield0687', 'cranfield0696', 'cranfield0700', 'cranfield0701', 'cranf
ield0702', 'cranfield0705', 'cranfield0710', 'cranfield0748', 'cranfield0749', 'cranfield0750', 'cranfield0757', 'cr
anfield0772', 'cranfield0773', 'cranfield0775', 'cranfield0779', 'cranfield0798', 'cranfield0799', 'cranfield0801',
 'cranfield0830', 'cranfield0894', 'cranfield0903', 'cranfield0907', 'cranfield0933', 'cranfield0940', 'cranfield0941
', 'cranfield0960', 'cranfield0961', 'cranfield0970', 'cranfield0977', 'cranfield0987', 'cranfield0990', 'cranfield1
006', 'cranfield1061', 'cranfield1072', 'cranfield1076', 'cranfield1081', 'cranfield1082', 'cranfield1083', 'cranfie
ld1105', 'cranfield1115', 'cranfield1153', 'cranfield1157', 'cranfield1160', 'cranfield1179', 'cranfield1182', 'cran
field1195', 'cranfield1201', 'cranfield1206', 'cranfield1207', 'cranfield1223', 'cranfield1224', 'cranfield1225', 'c
ranfield1229', 'cranfield1235', 'cranfield1241', 'cranfield1245', 'cranfield1248', 'cranfield1250', 'cranfield1253',
 'cranfield1261', 'cranfield1274', 'cranfield1276', 'cranfield1277', 'cranfield1281', 'cranfield1291', 'cranfield129
7', 'cranfield1299', 'cranfield1301', 'cranfield1302', 'cranfield1319', 'cranfield1328', 'cranfield1329', 'cranfield
1336', 'cranfield1339', 'cranfield1341', 'cranfield1344', 'cranfield1354', 'cranfield1366', 'cranfield1370', 'cranfi
eld1381', 'cranfield1393']
No of comparisions:  466
```

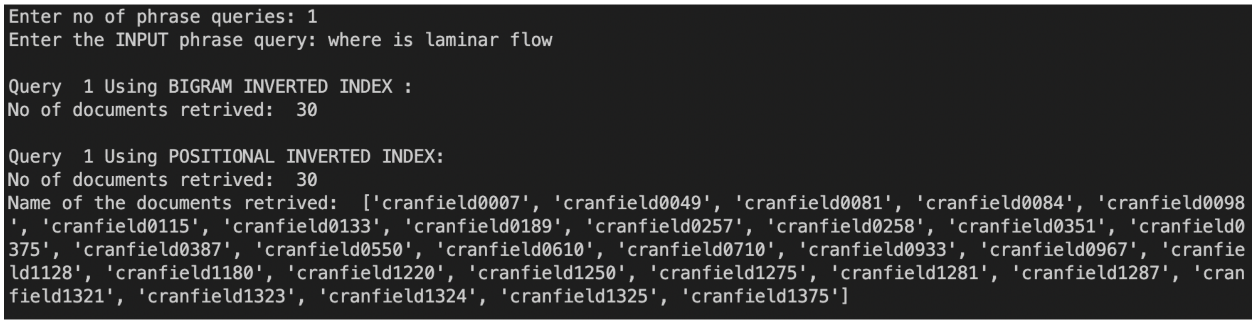
Results of part-3 use bigram and positional index.



```
Enter no of phrase queries: 1
Enter the INPUT phrase query: two dimensional

Query  1 Using BIGRAM INVERTED INDEX :
No of documents retrived:  162

Query  1 Using POSITIONAL INVERTED INDEX:
No of documents retrived:  162
Name of the documents retrived:  ['cranfield0002', 'cranfield0004', 'cranfield0008', 'cranfield0009', 'cranfield0015
', 'cranfield0018', 'cranfield0020', 'cranfield0025', 'cranfield0049', 'cranfield0052', 'cranfield0059', 'cranfield0
060', 'cranfield0073', 'cranfield0089', 'cranfield0094', 'cranfield0118', 'cranfield0126', 'cranfield0128', 'cranfie
ld0131', 'cranfield0158', 'cranfield0160', 'cranfield0167', 'cranfield0173', 'cranfield0177', 'cranfield0179', 'cran
field0188', 'cranfield0199', 'cranfield0201', 'cranfield0204', 'cranfield0205', 'cranfield0206', 'cranfield0212', 'c
ranfield0217', 'cranfield0221', 'cranfield0228', 'cranfield0242', 'cranfield0245', 'cranfield0264', 'cranfield0277',
 'cranfield0283', 'cranfield0284', 'cranfield0291', 'cranfield0292', 'cranfield0298', 'cranfield0300', 'cranfield030
1', 'cranfield0312', 'cranfield0315', 'cranfield0316', 'cranfield0364', 'cranfield0373', 'cranfield0380', 'cranfield
0404', 'cranfield0406', 'cranfield0409', 'cranfield0413', 'cranfield0417', 'cranfield0426', 'cranfield0433', 'cranfi
eld0440', 'cranfield0442', 'cranfield0445', 'cranfield0453', 'cranfield0455', 'cranfield0467', 'cranfield0472', 'cra
nfield0484', 'cranfield0487', 'cranfield0504', 'cranfield0526', 'cranfield0527', 'cranfield0544', 'cranfield0559', '
cranfield0563', 'cranfield0577', 'cranfield0593', 'cranfield0597', 'cranfield0601', 'cranfield0624', 'cranfield0631'
, 'cranfield0637', 'cranfield0651', 'cranfield0652', 'cranfield0655', 'cranfield0660', 'cranfield0662', 'cranfield06
71', 'cranfield0687', 'cranfield0696', 'cranfield0700', 'cranfield0701', 'cranfield0702', 'cranfield0710', 'cranfiel
d0748', 'cranfield0749', 'cranfield0750', 'cranfield0757', 'cranfield0772', 'cranfield0773', 'cranfield0775', 'cranf
ield0779', 'cranfield0798', 'cranfield0799', 'cranfield0894', 'cranfield0903', 'cranfield0933', 'cranfield0940', 'cr
anfield0941', 'cranfield0960', 'cranfield0961', 'cranfield0970', 'cranfield0977', 'cranfield0987', 'cranfield0990',
'cranfield1006', 'cranfield1061', 'cranfield1072', 'cranfield1076', 'cranfield1081', 'cranfield1082', 'cranfield1083
', 'cranfield1105', 'cranfield1115', 'cranfield1153', 'cranfield1157', 'cranfield1160', 'cranfield1179', 'cranfield1
182', 'cranfield1195', 'cranfield1201', 'cranfield1206', 'cranfield1207', 'cranfield1223', 'cranfield1224', 'cranfie
ld1225', 'cranfield1229', 'cranfield1235', 'cranfield1241', 'cranfield1245', 'cranfield1248', 'cranfield1250', 'cran
field1253', 'cranfield1261', 'cranfield1274', 'cranfield1276', 'cranfield1281', 'cranfield1291', 'cranfield1299', 'c
ranfield1301', 'cranfield1302', 'cranfield1319', 'cranfield1328', 'cranfield1329', 'cranfield1336', 'cranfield1339',
 'cranfield1341', 'cranfield1344', 'cranfield1354', 'cranfield1366', 'cranfield1370', 'cranfield1381', 'cranfield139
3']
```



```
Enter no of phrase queries: 1
Enter the INPUT phrase query: where is laminar flow

Query  1 Using BIGRAM INVERTED INDEX :
No of documents retrived:  30

Query  1 Using POSITIONAL INVERTED INDEX:
No of documents retrived:  30
Name of the documents retrived:  ['cranfield0007', 'cranfield0049', 'cranfield0081', 'cranfield0084', 'cranfield0098
', 'cranfield0115', 'cranfield0133', 'cranfield0189', 'cranfield0257', 'cranfield0258', 'cranfield0351', 'cranfield0
375', 'cranfield0387', 'cranfield0550', 'cranfield0610', 'cranfield0710', 'cranfield0933', 'cranfield0967', 'cranfie
ld1128', 'cranfield1180', 'cranfield1220', 'cranfield1250', 'cranfield1275', 'cranfield1281', 'cranfield1287', 'cran
field1321', 'cranfield1323', 'cranfield1324', 'cranfield1325', 'cranfield1375']
```