INVERTED INDEX AND ADVANCED OPERATORS FOR INFORMATION RETRIEVAL

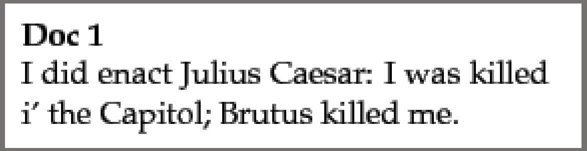
By Victor and Akam

OUR CHOSEN TOPIC

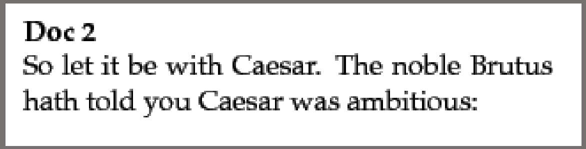
Implementing the `ANDNOT` operator over posting lists.

We will present the ANDNOT method, the logic and algorithm behind it and explain its purpose.

DOCUMENTS

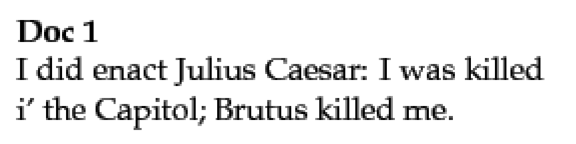


Doc 1
I did enact Julius Caesar: I was killed
i' the Capitol; Brutus killed me.

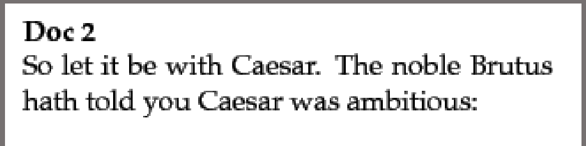


Doc 2
So let it be with Caesar. The noble Brutus
hath told you Caesar was ambitious:

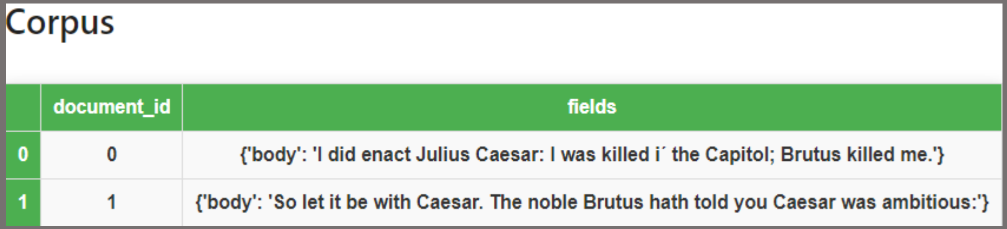
CORPUS: LIST OF DOCUMENTS WITH IDS AND BODY FIELDS



Doc 1
I did enact Julius Caesar: I was killed i’ the Capitol; Brutus killed me.



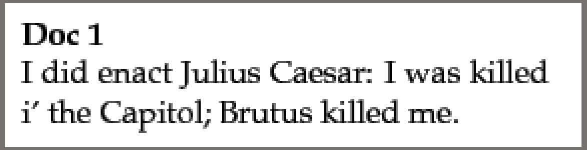
Doc 2
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:



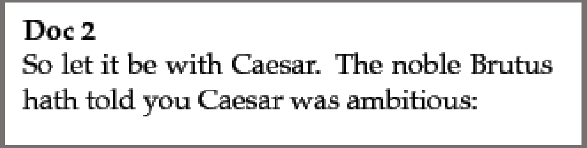
Corpus

	document_id	fields
0	0	{'body': 'I did enact Julius Caesar: I was killed i´ the Capitol; Brutus killed me.'}
1	1	{'body': 'So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:'}

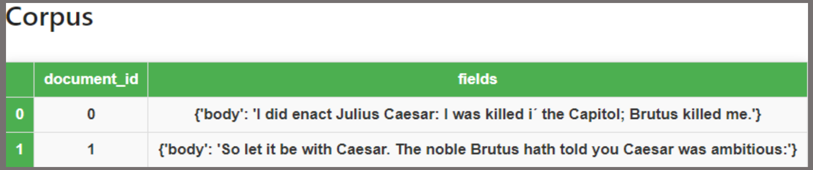
PROCESSING PIPELINE – TOKENIZATION, NORMALIZATION, MAPPING TERMS TO AN INCREMENTAL ID AND SORTING OF THE TERMS



Doc 1
I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.

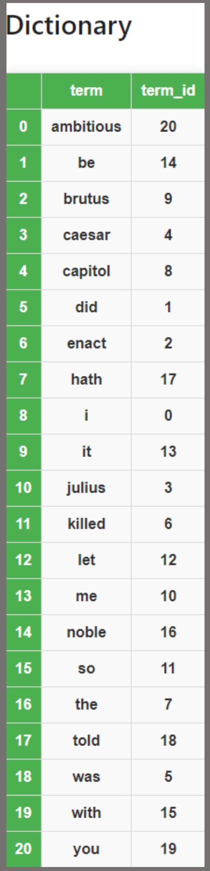


Doc 2
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:



Corpus

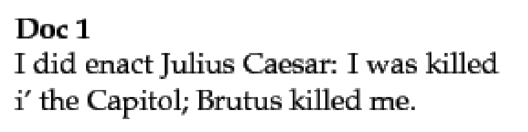
	document_id	fields
0	0	{'body': 'I did enact Julius Caesar: I was killed i´ the Capitol; Brutus killed me.'}
1	1	{'body': 'So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:'}



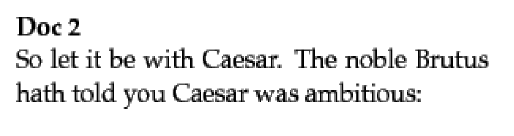
Dictionary

	term	term_id
0	ambitious	20
1	be	14
2	brutus	9
3	caesar	4
4	capitol	8
5	did	1
6	enact	2
7	hath	17
8	i	0
9	it	13
10	julius	3
11	killed	6
12	let	12
13	me	10
14	noble	16
15	so	11
16	the	7
17	told	18
18	was	5
19	with	15
20	you	19

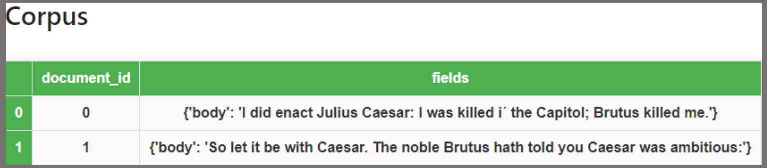
INVERTED INDEX -
MAPPING TERMS TO POSTINGS LISTS



Doc 1
I did enact Julius Caesar: I was killed
i' the Capitol; Brutus killed me.

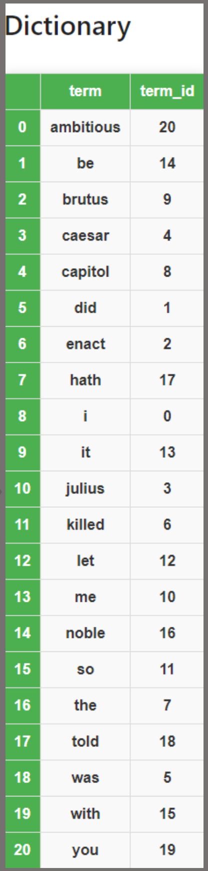


Doc 2
So let it be with Caesar. The noble Brutus
hath told you Caesar was ambitious:



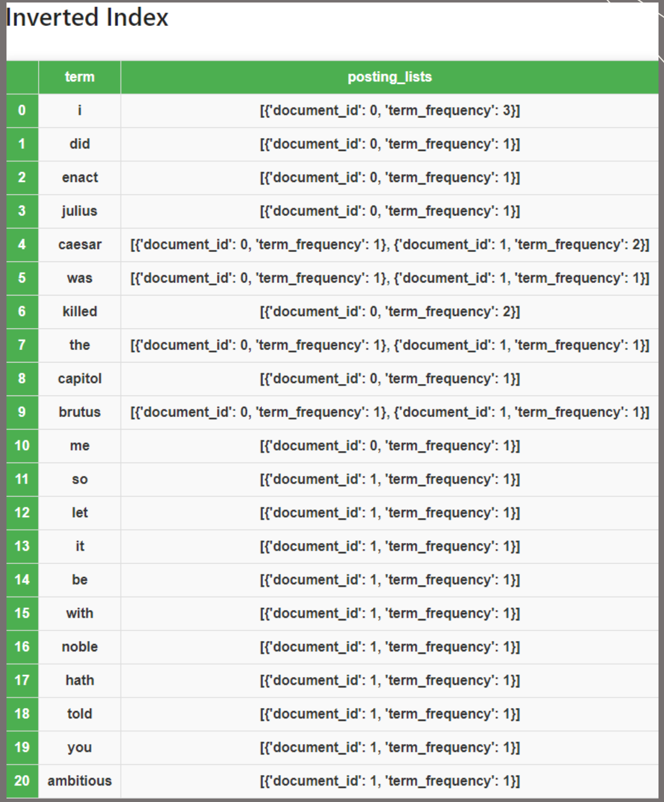
Corpus

	document_id	fields
0	0	{'body': 'I did enact Julius Caesar: I was killed i´ the Capitol; Brutus killed me.'}
1	1	{'body': 'So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:'}



Dictionary

	term	term_id
0	ambitious	20
1	be	14
2	brutus	9
3	caesar	4
4	capitol	8
5	did	1
6	enact	2
7	hath	17
8	i	0
9	it	13
10	julius	3
11	killed	6
12	let	12
13	me	10
14	noble	16
15	so	11
16	the	7
17	told	18
18	was	5
19	with	15
20	you	19



Inverted Index

	term	posting_lists
0	i	[{'document_id': 0, 'term_frequency': 3}]
1	did	[{'document_id': 0, 'term_frequency': 1}]
2	enact	[{'document_id': 0, 'term_frequency': 1}]
3	julius	[{'document_id': 0, 'term_frequency': 1}]
4	caesar	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 2}]
5	was	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
6	killed	[{'document_id': 0, 'term_frequency': 2}]
7	the	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
8	capitol	[{'document_id': 0, 'term_frequency': 1}]
9	brutus	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
10	me	[{'document_id': 0, 'term_frequency': 1}]
11	so	[{'document_id': 1, 'term_frequency': 1}]
12	let	[{'document_id': 1, 'term_frequency': 1}]
13	it	[{'document_id': 1, 'term_frequency': 1}]
14	be	[{'document_id': 1, 'term_frequency': 1}]
15	with	[{'document_id': 1, 'term_frequency': 1}]
16	noble	[{'document_id': 1, 'term_frequency': 1}]
17	hath	[{'document_id': 1, 'term_frequency': 1}]
18	told	[{'document_id': 1, 'term_frequency': 1}]
19	you	[{'document_id': 1, 'term_frequency': 1}]
20	ambitious	[{'document_id': 1, 'term_frequency': 1}]

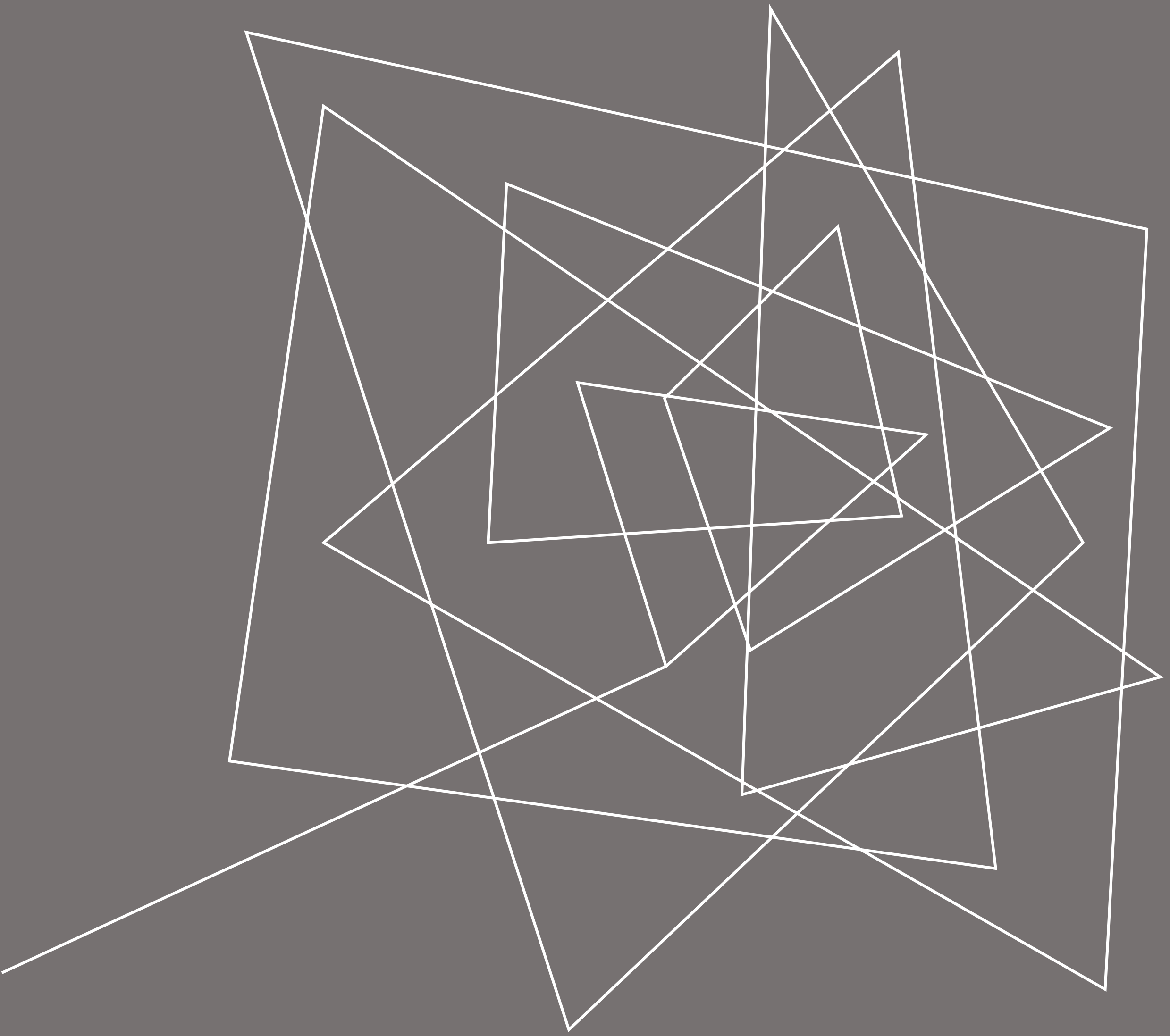
BOOLEAN RETRIEVAL OPERATIONS: DIFFERENT OPERATIONS ON POSTING LISTS:

Conjunctive AND operation

Finding documents that contain both term A and term B and retaining only the common document_ids between the two posting lists, e.q: posting list for term A is [1,3,5,6] and posting list for term B is [2,3,4,6], the **AND** operation results in [3,6]

Disjunctive OR operation

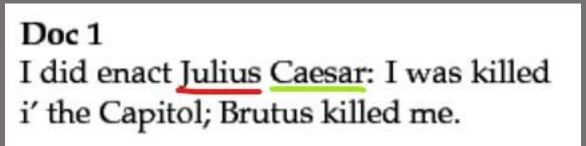
Finding documents that contain either term A or term B and including all unique document_ids from the two posting lists, e.q: posting list for term A is [1,3,5,7] and posting list for term B is [3,5,7,9], the **OR** operation results in [1,3,5,7,9]



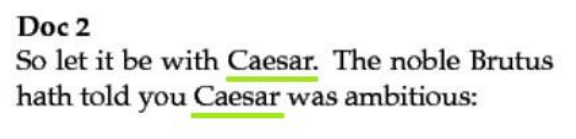
PRESENTING THE ANDNOT OPERATOR OVER POSTING LISTS

Finding documents that contain term A but not term B

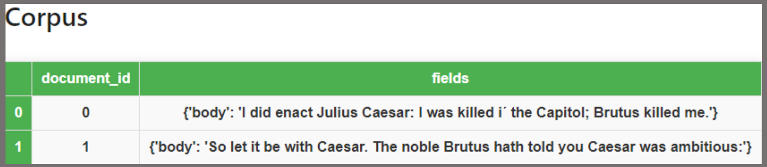
ANDNOT OPERATOR - LET'S FIND THE DOCUMENT CONTAINING THE TERM CAESAR BUT NOT JULIUS



Doc 1
I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.

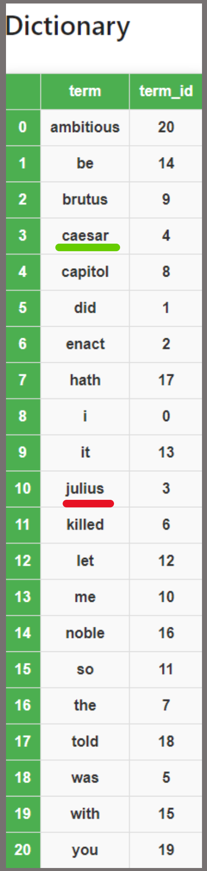


Doc 2
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:



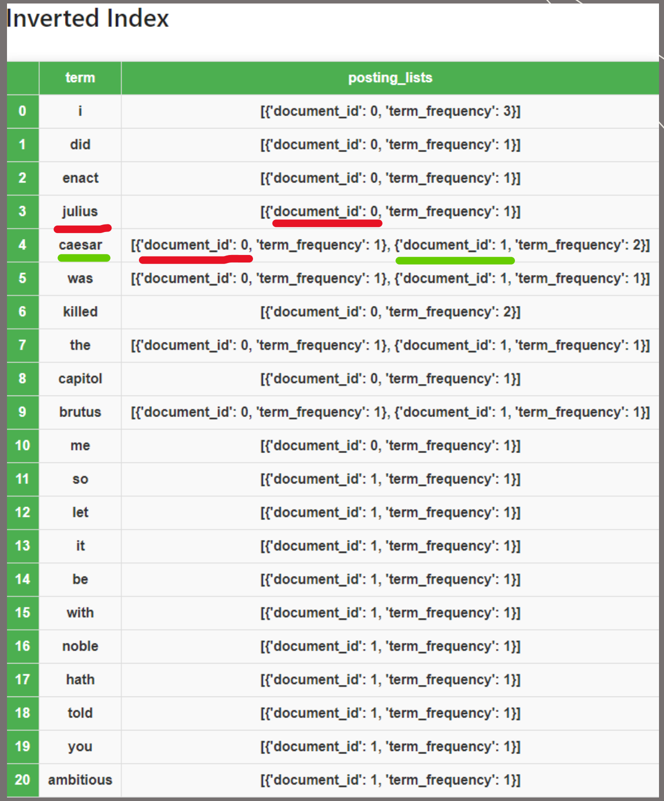
Corpus

	document_id	fields
0	0	{'body': 'I did enact Julius Caesar: I was killed i´ the Capitol; Brutus killed me.'}
1	1	{'body': 'So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:'}



Dictionary

	term	term_id
0	ambitious	20
1	be	14
2	brutus	9
3	caesar	4
4	capitol	8
5	did	1
6	enact	2
7	hath	17
8	i	0
9	it	13
10	julius	3
11	killed	6
12	let	12
13	me	10
14	noble	16
15	so	11
16	the	7
17	told	18
18	was	5
19	with	15
20	you	19

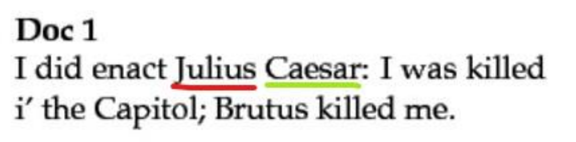


Inverted Index

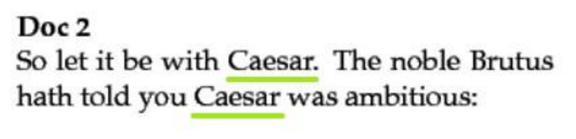
	term	posting_lists
0	i	[{'document_id': 0, 'term_frequency': 3}]
1	did	[{'document_id': 0, 'term_frequency': 1}]
2	enact	[{'document_id': 0, 'term_frequency': 1}]
3	julius	[{'document_id': 0, 'term_frequency': 1}]
4	caesar	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 2}]
5	was	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
6	killed	[{'document_id': 0, 'term_frequency': 2}]
7	the	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
8	capitol	[{'document_id': 0, 'term_frequency': 1}]
9	brutus	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
10	me	[{'document_id': 0, 'term_frequency': 1}]
11	so	[{'document_id': 1, 'term_frequency': 1}]
12	let	[{'document_id': 1, 'term_frequency': 1}]
13	it	[{'document_id': 1, 'term_frequency': 1}]
14	be	[{'document_id': 1, 'term_frequency': 1}]
15	with	[{'document_id': 1, 'term_frequency': 1}]
16	noble	[{'document_id': 1, 'term_frequency': 1}]
17	hath	[{'document_id': 1, 'term_frequency': 1}]
18	told	[{'document_id': 1, 'term_frequency': 1}]
19	you	[{'document_id': 1, 'term_frequency': 1}]
20	ambitious	[{'document_id': 1, 'term_frequency': 1}]

IMPLEMENTING THE ANDNOT OPERATOR OVER POSTING LISTS -

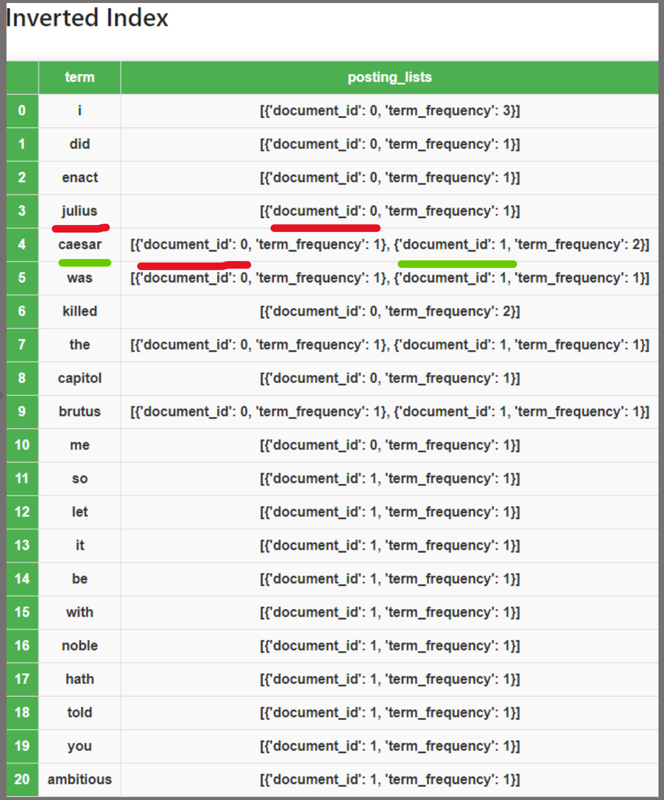
LET'S FIND THE DOCUMENT CONTAINING THE TERM CAESAR BUT NOT JULIUS



Doc 1
I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.



Doc 2
So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:



Inverted Index

	term	posting_lists
0	i	[{'document_id': 0, 'term_frequency': 3}]
1	did	[{'document_id': 0, 'term_frequency': 1}]
2	enact	[{'document_id': 0, 'term_frequency': 1}]
3	julius	[{'document_id': 0, 'term_frequency': 1}]
4	caesar	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 2}]
5	was	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
6	killed	[{'document_id': 0, 'term_frequency': 2}]
7	the	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
8	capitol	[{'document_id': 0, 'term_frequency': 1}]
9	brutus	[{'document_id': 0, 'term_frequency': 1}, {'document_id': 1, 'term_frequency': 1}]
10	me	[{'document_id': 0, 'term_frequency': 1}]
11	so	[{'document_id': 1, 'term_frequency': 1}]
12	let	[{'document_id': 1, 'term_frequency': 1}]
13	it	[{'document_id': 1, 'term_frequency': 1}]
14	be	[{'document_id': 1, 'term_frequency': 1}]
15	with	[{'document_id': 1, 'term_frequency': 1}]
16	noble	[{'document_id': 1, 'term_frequency': 1}]
17	hath	[{'document_id': 1, 'term_frequency': 1}]
18	told	[{'document_id': 1, 'term_frequency': 1}]
19	you	[{'document_id': 1, 'term_frequency': 1}]
20	ambitious	[{'document_id': 1, 'term_frequency': 1}]

Implementation:

We have two pointers from two posting lists, each pointing to the first element of each list

1. As long as the first list is not empty, we will check for the statements below in the shown order:

2. *If the second list is empty*, we yield the current element of first list and we move to next element of the first list and repeat again from step 1.

3. *If* the document id of the element in first list is smaller than the document id of the element in second list, we yield the current element of first list and we move to next element in the first list and repeat again from step 1.

4. *If the document ids are equal*, we move to next element in the first list and repeat again from step 1.

5. *If* the document id of the element in first list is greater than the document id of the element in second list, we move to next element in the second list and repeat again from step 1.

After this process, we have gone through and compared both lists and chosen the elements that are in the first list but not in the second list.

DEMONSTRATION OF THE ANDNOT OPERATION

SUMMARY

This ANDNOT operation is valuable in refining search results and providing more specific and relevant information to the user in information retrieval systems.

THANK YOU