1. Run the program and view results on the console.
2. Run the program and view results on the console.
3. Weighting Scheme 1

Query	Relevant Documents	Non-relevant Documents
1	13, 486, 359, 184	665
2	12, 51, 884, 746	1170
3	485, 5, 399, 144, 181	
4	166, 488, 236	1189, 1061
5	103, 943	1032, 1272, 625
6	491, 385, 257, 121, 386	
7	492, 434, 973, 56	124
8	122, 556, 492, 433	19
9	21, 45, 306, 550, 102	
10	493, 302, 1143, 949	1010
11	495, 654, 472, 72, 1327	
12	624, 650, 966	941, 506
13	496, 903, 313	38, 520
14	64, 291, 256, 170, 335	
15	462	1099, 761, 1025, 1043
16	498, 1006, 106	93, 869
17	700, 445, 1281, 916, 1108	
18	248, 498, 492, 197	1006
19	706	82, 1346, 716, 554
20	500, 268, 88, 270, 450	

Weighting Scheme 2

Query	Relevant Documents	Non-relevant Documents
1	13, 486, 184	665, 573
2	12, 51, 875, 884	1170
3	485, 5, 399, 144, 181	
4	166, 488	1189, 1255, 1061
5	943, 103	1032, 1272, 355
6	491, 121, 257, 385	148
7	492, 973, 122, 56	124
8	556, 232, 122, 492	19
9	45, 306, 21, 550, 102	
10	493, 302, 332, 1143	1010
11	495, 472, 654, 72, 570	
12	650, 624, 966, 1232	506
13	496, 903, 313	38, 520
14	64, 291, 256, 335, 568	
15	462	1099, 1043, 1025, 981
16	498, 1006, 106	93, 869
17	916, 372, 700, 1108, 445	
18	498, 248, 197, 492	1006

19	706, 1279	716, 82, 1346
20	500, 268, 88, 270, 450	

4. Top-ranked non-relevant document did not get a lower score because of many reasons. Some of the reasons are listed below.
    - Too big query: the query contains a lot of terms, which makes it difficult to keep a track of the context and it might easily lose relevant information while computing the cosine product scores.
    - Not so perfect weighting scheme: Term frequency (tf) is given a lot of importance. If a query term appears a lot in the document, that document will get a higher weight value even though it might not be relevant.
    - Small collection size: It might be possible that more documents corresponding to the query might not be present in the collection. And since we need to retrieve top-5 results, it can be possible that only 1 relevant document appears in the collection and just to make the top 5, we have to add the other 4 non-relevant documents.

5. W1: This weighting scheme considers maxtf while assigning weights. So, if a non-relevant term with respect to the query appears the most in the document, that document will get a higher weight, resulting in a better cosine product score.

   W2: This weighting scheme considers doclen while assigning weights. So, if a non-relevant document with respect to the query has too many lemmas (making the doclen value a lot more), that document will get a higher weight, resulting in a better cosine product score.

   The impact of doclen is worse than the impact of maxtf, as the variation in the maxtf can't be that large as there can be the variation in the doclen (we can have too large and too small documents). As a result, W2 weighting scheme is worse than W1 weighting scheme. And similar results are obtained, as shown in Question 3, where there are comparatively more non-relevant documents using W2 weighting scheme than with W1 weighting scheme.

6.
    - Cranfield collection is parsed, tokenized, and index is created. (As was done in assignment 2)
    - Queries from the input file are similarly parsed, tokenized, and a query index is created.
    - Weight vector map is generated for both the documents as well as queries. 2 different maps are generated for both the weighting schemes. This weight vector map contains the doc_id as the key and lemma_term:weight_value as the value.
    - While generating the weight vector map, after computing the weights using the W1 and W2 weighting schemes, they are normalized, and later the dictionary is sorted based on doc_id.

- For getting the top-5 documents for each query, for each document as well as for each query, a document and a query map is created respectively. It stores the lemma_term as the key and its corresponding normalized weight as the value. If the document term is present in query, the normalized weights corresponding to that term in both document as well as query is computed and the results are stored in a map, having key as the cosine product score and value as the document vector representation.
- Top-5 scores from the cosine product map are retrieved and shown on the console.
- In order to get the headline for each document in the top-5 results, a title map is created in the start where the headlines corresponding to each document are stored. This map is used while printing headline of the top-5 document vectors.