**Experiment-01**

**--------------------------------------------------------------------**

**Name:** Atharva Paliwal

**Roll No:** 40

**Date:** 21/07/2021

**--------------------------------------------------------------------**

**Aim:** Write a program to implement a small search engine. Consider 10 documents/web pages as the search domain.

A. Create inverted index for the documents.

B. Create a search system for answering the search queries using the index created.

C. Modify the index to add the term frequency information and Rank the output based on term frequency.

**Details:**

* The documents can be simple text files with contents belonging to different domains or downloaded webpages in HTML format, You can parse the tags using regular expressions to extract the text from the corresponding tags.
* The documnts/web pages may contain non-alphanumeric characters and capital letters. However, we don’t want to index apple, Apple, and APPLE differently. They are basically all apple.
* Handle grammer: Considering the grammatical structure of the language, we don’t want to index research, researches, and researching separately. They are all about research. When we search for one of these terms we would expect results containing any of those variations.
* Removal of Stop words: We wouldn’t like to index words such as ‘the’, ‘a’, ‘an’ because they appear in almost every document and they don’t give us very much information about the document or the query.
* Query Types/ Searching- One Word Queries (Queries that consist of a single word), Multi-word queries

**Theory:**

An inverted index is a data structure that we build while parsing the documents that we are going to answer the search queries on. Given a query, we use the index to return the list of documents relevant for this query. The inverted index contains mappings from terms (words) to the documents that those terms appear in. Each vocabulary term is a key in the index whose value is its postings list. A term’s postings list is the list of documents that the term appears in.

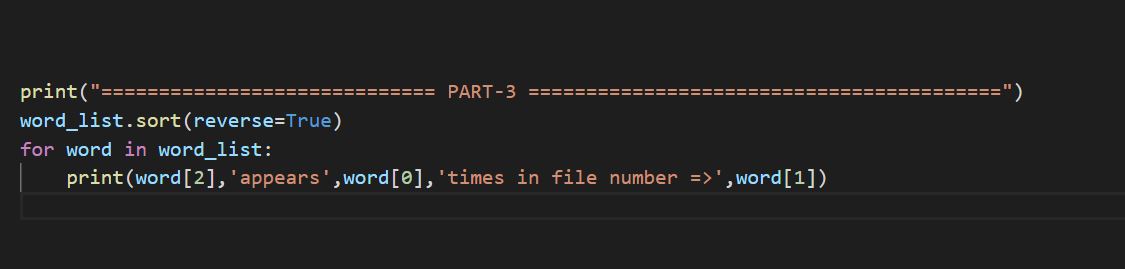
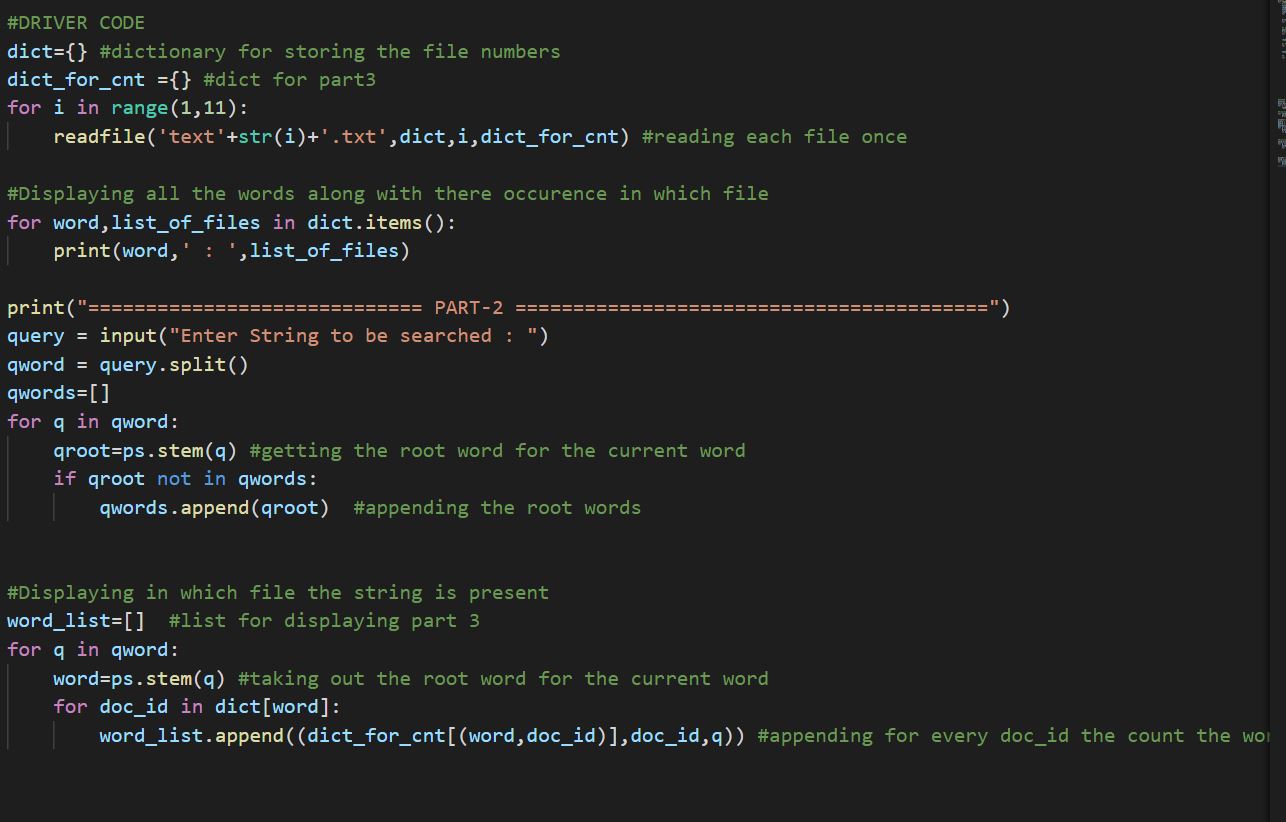
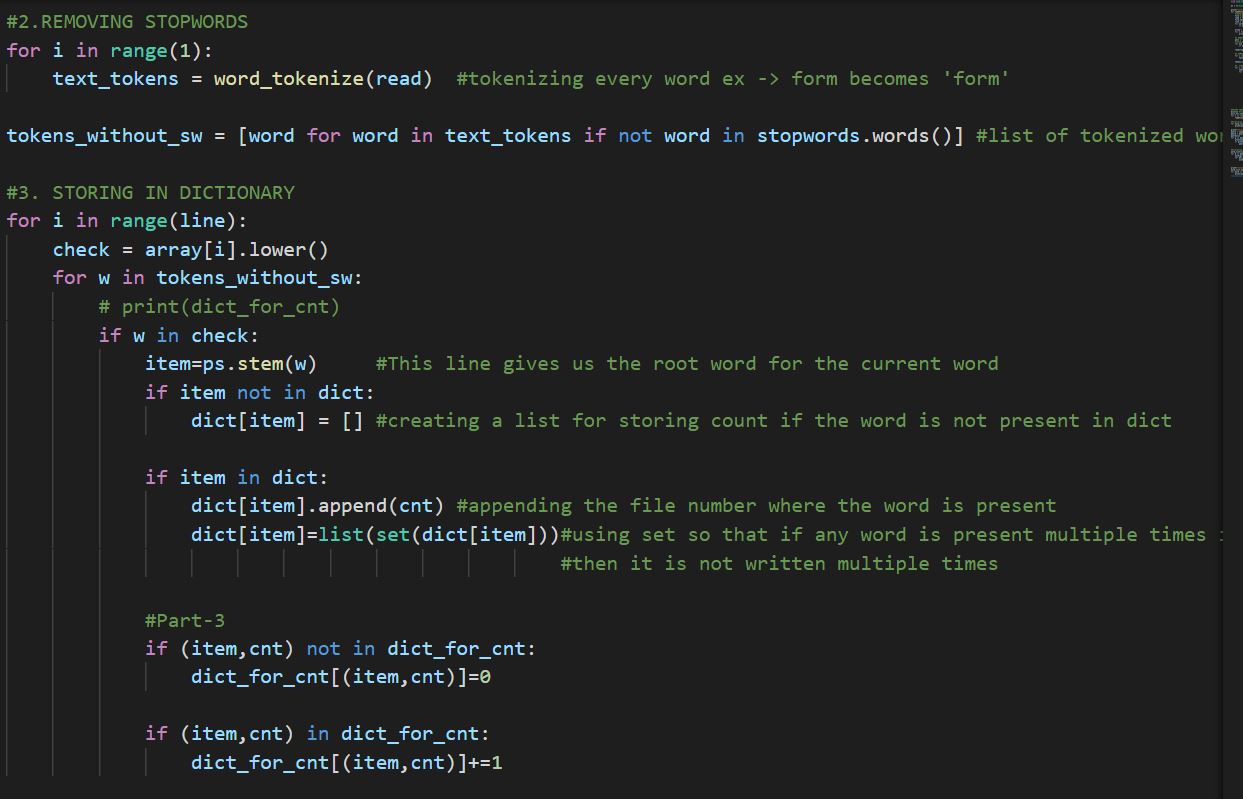
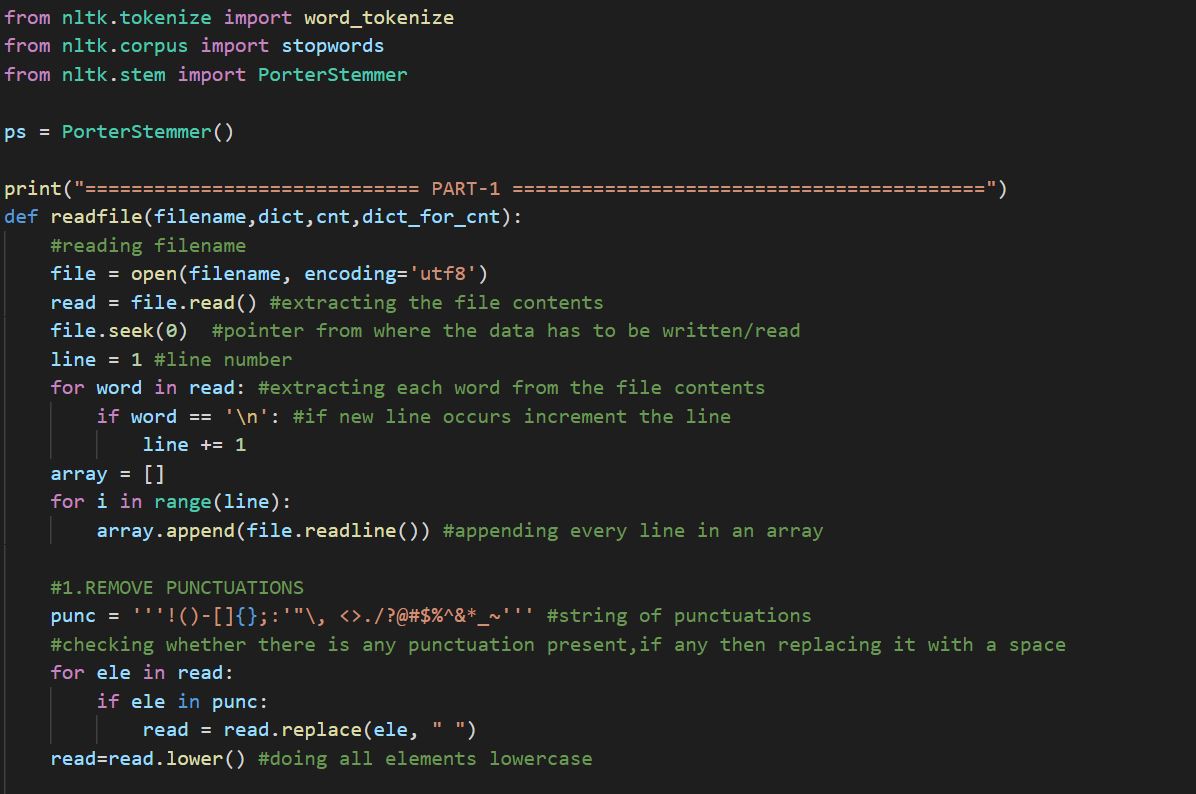
To illustrate with an example, if we have the following documents:

Document 1: Information Retrieval and Web Search

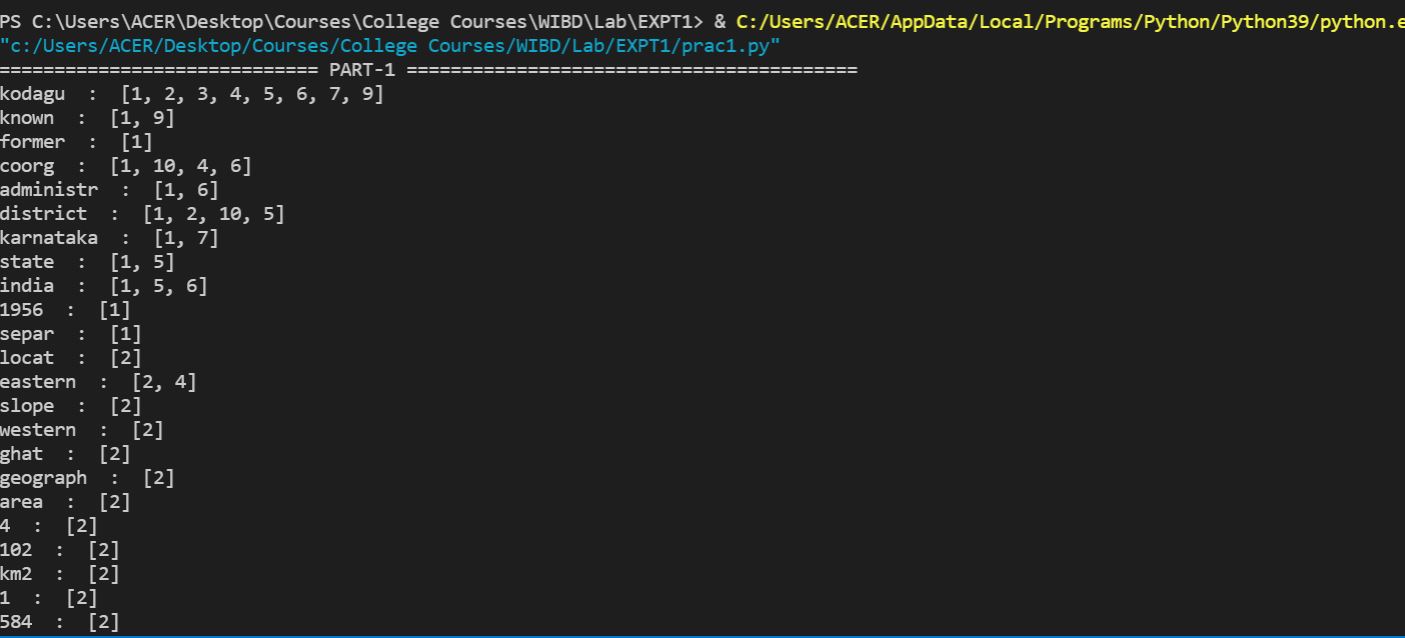
Document 2: Search Engine Ranking

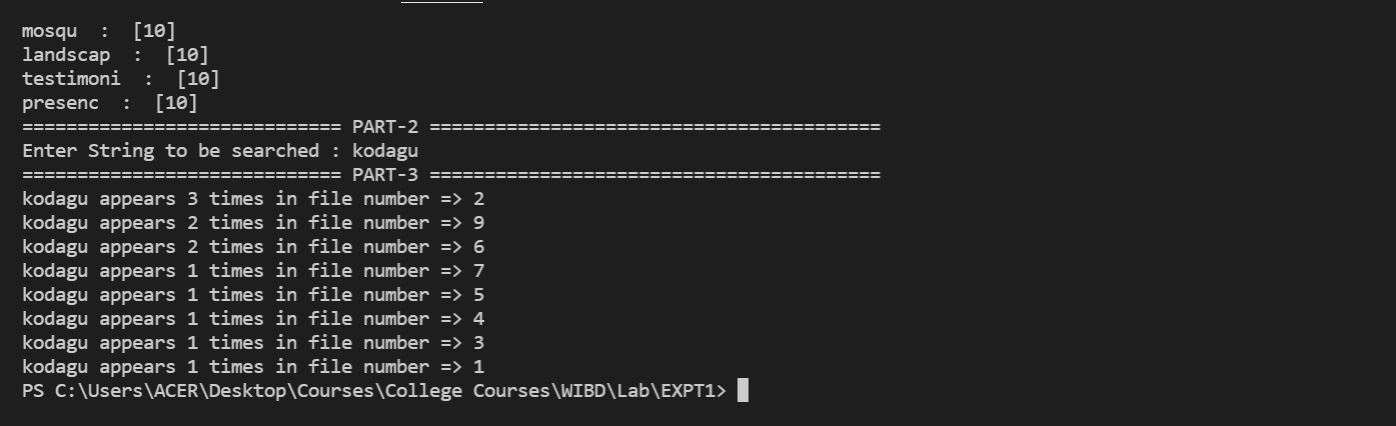
Document 3: Web Search Course

Then the postings list of the term ‘web’ would be the list [1, 3], meaning the term ‘web’ appears in documents with IDs 1 and 3. Similarly the postings list of the term ‘search’ would be [1, 2, 3], and for the term ‘course’ the postings list would be [3]. We may want to keep additional information in the index such as the number of occurrences of the term in the whole collection (its term frequency), or the number of different documents that the term appears in (its document frequency), the positions of the term’s occurrences within a document etc. The amount of information we keep in our index will grow as we add more functionality to our search engine.

**Program:**

**Output:**





**\*\*\*END\*\*\***