IEEEXplore Indexing

Time limit: *2500 ms*
Memory limit: *256 MB*

Warm greetings to all IEEEXtreme Participants from the Xplore API Team!

As part of the IEEEXtreme competition, in this challenge you are asked to identify the most important index terms of an academic journal. This challenge emulates a real-world application - The IEEE Xplore API uses a similar approach!

For a full dynamic database search IEEE Xplore API is available for your IEEE research needs. Xplore API provides metadata on 4.9M academic works and is now delivering full-text content on 50K *Open Access* articles. Xplore API will meet your research needs fast and easy. The Xplore API Portal supports PHP, Python and Java as well as providing output in Json and XML formats. Many API use cases are listed within the API Portal.

Xplore API registration is free. To learn more about IEEE Xplore API please visit developer.ieee.org and register for an API key TODAY!

Challenge

An index term, subject term, subject heading, or descriptor, in information retrieval, is a term that captures the essence of the topic of a document. Index terms make up a controlled vocabulary for use in bibliographic records. They are an integral part of bibliographic control, which is the function by which libraries collect, organize and disseminate documents.

In this challenge you are given a list of index terms, a list of stop words, and a document. You are asked to parse the document and identify the top 3 index terms that have the highest keyword density.

Definitions

For the purpose of this challenge:

- A *word* is a consecutive sequence of lowercase letters (`a-z`) or apostrophe (`'`). A word must not contain any other characters. The sequence must have at least 4 characters.
- A *punctuation* is a comma (`,`), a period (`.`), a question mark (`?`), or an exclamation mark (`!`).
- A *token* is a consecutive sequence of lowercase or uppercase letters (`a-zA-Z`), apostrophes (`'`), or punctuation. A token must not contain spaces or newlines.
- A *index term* is a word. A *stop word* is a word.
- A *document* is a simplified XML with matched *tags* such as `<body>` , and `</abstract>` . An opening tag has the format `<[a-z]+>` . A closing tag has the format `</[a-z]+>` . In other words, a tag only has lowercase letters in them and is always enclosed by a pair of brackets `<>` .
- There are three *special tags*: `<title>` , `<abstract>` , and `<body>` .
- Tags can be nested, such as `<a><p></p></a>` .
- A document may have an arbitrary number of tokens, spaces, or newlines between its tags. No tokens appear outside all the tags.

Keyword Density

Here is the methodology to compute the keyword density for an index term w:

- Each token in the document is normalized to its underlying word, by first converting all uppercase letters to lowercase, and then removing all punctuation in it.
- All words that are *stop words* are ignored.
- Compute the total number of words in the special tags of the document as L. A same word may appear multiple times in a special tag and is counted multiple times in L. Note that words outside special tags are not counted towards L.
- Compute the index term score S_w by its occurrences in the special tags: Each occurrence in the `<title>` scores 5. Each occurrence in the `<abstract>` scores 3. Each occurrence in the `<body>` scores 1.
- The *keyword density* of an index term w is defined as $S_w / L \cdot 100$

Standard input

The first line of the input has a list of stop words separated by a single semicolon.

The second line of the input has a list of index terms separated by a single semicolon.

From line three to the end of the input file gives the XML document.

Constraints and notes

- The number of index terms is between 5 and 30.
- The number of stop words is between 5 and 150.
- All index terms are distinct. All stop words are distinct. No index term is a stop word, and vice versa.
- The document contains exactly one `<title>`, one `<abstract>`, and one `<body>` tag. No special tag is in another special tag. However, a special tag can be nested in other tags.
- The document contains only lowercase letters (`a-z`), uppercase letters (`A-Z`), apostrophes (`'`), punctuation (`,.?!`), XML tags, spaces (not tabs), or newlines.
- The document contains at most $20\,000$ characters.
- No line of the document has trailing spaces. However, a line may have leading spaces as indentation.
- No line of the document contains only the newline character. There are no empty lines.
- Each XML tag starts and ends on a same line.
- The input files of this challenge are not only chosen from real-world scenarios, but may also be specially constructed to test that your computation is correct, just as in the other challenges.

Input	Output	Explanation
being;does;have;haven't;more classification;cryptography; <response> <article> <title>A Novel Approach to <publicationtitle>IEEE Tra <abstract>Classification o </article> <body> <sec> <label>I.</label> Should Haven't That is I bet diseases you can <fig> <label>FIGURE.</label> <caption>This is a figu </fig> </sec> </body> </response>	classification: 19.512195122 stability: 9.756097561 probability: 3.658536585	There are a total of 82 words. Their scores S_w are: • classification: 16 • stability: 8 • probability: 3 • cryptography: 1 • diseases: 1 For `classification`, the keyword density is $16/82 \cdot 100 \approx 19.512195$. The other keyword densities can be computed similarly.
what;when;where;like;that welcome;ieee;xtreme;ieeextre <title>Welcome to IEEEXtreme <keyword>welcome, ieeextreme <abstract> Welcome!Participants!!! IEEE Xtreme is a global chal Compete in a twentyfour hour </abstract> <body>WELCOME. wel.come... A <other> Mark your calender and don't </other>	ieee: 30.000000000 ieeextreme: 20.000000000 welcome: 17.500000000 xtreme: 17.500000000	Make sure that you read the problem statement correctly. Do not forget to handle the tied index terms and output them in lexicographical order.