819. Most Common Word

Description (/problems/most-common-word/description/) Hints (/problems/most-common-word/hints/) Submissions (/prob

Pick One (/problems/random-one-question/)

Given a paragraph and a list of banned words, return the most frequent word that is not in the list of banned words. It is guaranteed there is at least one word that isn't banned, and that the answer is unique.

Words in the list of banned words are given in lowercase, and free of punctuation. Words in the paragraph are not case sensitive. The answer is in lowercase.

```
Example:
Input:
paragraph = "Bob hit a ball, the hit BALL flew far after it was hit."
banned = ["hit"]
Output: "ball"
Explanation:
"hit" occurs 3 times, but it is a banned word.
"ball" occurs twice (and no other word does), so it is the most frequent non-banned wor
Note that words in the paragraph are not case sensitive,
that punctuation is ignored (even if adjacent to words, such as "ball,"),
and that "hit" isn't the answer even though it occurs more because it is banned.
```

Note:

- `1 <= paragraph.length <= 1000`.
- `1 <= banned.length <= 100`.
- `1 <= banned[i].length <= 10`.
- The answer is unique, and written in lowercase (even if its occurrences in `paragraph` may have uppercase symbols, and even if it is a proper noun.)
- `paragraph` only consists of letters, spaces, or the punctuation symbols `!?',;.`
- Different words in `paragraph` are always separated by a space.
- There are no hyphens or hyphenated words.
- Words only consist of letters, never apostrophes or other punctuation symbols.

Companies ▲

Amazon 68 (/company/amazon)

Related Topics ▲

String (/tag/string)

Java

```
class Solution {
    public String mostCommonWord(String paragraph, String[] banned) {
        Set<String> bannedSet = new HashSet<>();
        for (String w : banned) {
            bannedSet.add(w);
        }

        WordParser parser = new WordParser(paragraph);
        Map<String, Integer> wordCount = new HashMap<>();
        String word, maxWord = null;
        while ((word = parser.next()) != null) {
            if (bannedSet.contains(word)) {
                continue;
            }

            wordCount.put(word, wordCount.getOrDefault(word, 0) + 1);
            if (maxWord == null || wordCount.get(word) >
wordCount.get(maxWord)) {
                maxWord = word;
            }
        }
        return maxWord;
    }

    private static class WordParser {
        private final String text;
        private int nextIndex;
        private StringBuilder sb;

        public WordParser(String text) {
            this.text = text;
            sb = new StringBuilder();
        }

        public String next() {
            sb.setLength(0);
            nextIndex--;

            do {
                if (++nextIndex >= text.length()) {
                  return null;
                }
            } while (!Character.isLetter(text.charAt(nextIndex)));

```

Notes

```
44 ▾            do {
45                  sb.append(Character.toLowerCase(text.charAt(nextIndex++)));
46              } while (nextIndex < text.length() &&
    Character.isLetter(text.charAt(nextIndex)));
47
48              return sb.toString();
49          }
50
51      }
52  }
```

Notes

Custom Testcase (Contribute !)

Run Submit