void findNeedles(const std::string &haystack, const std::vector<std::string> &needles)

Prints the number of matches found for each element of an input vector of strings in another vector created by splitting, on whitespace, an input string into a vector of substrings.

Parameters

`haystack`

Input string to search.

`needles`

Input vector of strings to search for and count (maximum of 5 elements).

Return Value

None.

Usage Notes

- Note that the function does not differentiate between different types of characters in the input haystack (except whitespace). For example, the function will not count an occurrence of "testwordA" if the input haystack has the value "testwordA! testwordB" .
- The function prints the number of occurrences for each search term in the order of the input vector according to the form <input vector element string>: <number of occurrences>.
- The function will output a message through the standard error stream if you pass a vector with more than five elements.

Implementation and Example:

```
#include <iostream>
#include <iterator>
#include <sstream>
#include <string>
#include <vector>

void findNeedles(const std::string &haystack,
                 const std::vector<std::string> &needles) {

  if (needles.size() > 5) {
    std::cerr << "too many words!";
  }

  else{
    int countArray[needles.size()];
    for(int j = 0; j < needles.size(); j++){
      countArray[j] = 0;
    }

    std::istringstream haystack_ss(haystack);
    std::vector<std::string> words;

    // Split haystack into words
```

```
    std::copy(std::istream_iterator<std::string>(haystack_ss),
    std::istream_iterator<std::string>(),
    std::back_inserter<std::vector<std::string> >(words));

    for(int i = 0; i < needles.size(); i++) {
      for(int j = 0; j < words.size(); j++){
        if(words[j] == needles[i]){
          countArray[i]++;
        }
      }
    }

    for(int j = 0; j < needles.size(); j++){
      std::cout << needles[j] << ": " << countArray[j] << std::endl;
    }
  }
}

int main()
{
  std::string dwight = "I always wanted to be hay king...";
  std::vector<std::string> keywords = { "always", "king" };
  findNeedles(dwight, keywords);
  return 0;
}
```

Example output:

```
always: 1
king: 0
```

[Not for publication] Comments and questions for the person who wrote the code:

Hi [person], here's a draft of a new topic documenting findNeedles. I also came up with a few questions and suggestions as I was writing, though I understand you probably have very good reasons for the way you implemented this code, your answers will help me improve the documentation.

Would it be problematic to use arrays instead of vectors throughout for better memory consumption?

If we plan on calling findNeedles many times to search for different words in the same string, could you move the splitting outside of the function so we don't have to loop through it every time? Would it be advantageous to sort the output by number of occurrences?

Would it help users to add an option to return the values currently printed to the terminal or read/write them to a file?

Where is this function defined?

I think it would be beneficial to add some additional comments to the code, especially a top level description comment. Perhaps my description at the top of the draft might suffice? It might also be good to rename this function to something more intuitive if you can come up with a better name.

Lastly, your comment about splitting the haystack into words implies that your intention is to split a string into words. However, the code splits a string into strings, not words specifically. Perhaps you could add checks to strip punctuation marks or use regex if this was not your intention?

Thanks for your help,

Taylor