C'mon K-mers!

A useful operation on strings, specially when analyzing certain types of data, like genomic data, is to find all the substrings of length k in a string. These substrings are called the string's k-mers and we are typically interested in finding out how many times each k-mer appears within the string. For example, consider the following string:

```
aaabaab
```

This string has three unique 2-mers:

```
aaabaab


aa
 aa
  ab
   ba
    aa
     ab
```

More specifically, the 2-mers are `aa` (appearing three times), `ab` (appearing two times), and `ba` (appearing once). Notice how counting the k-mers considers *overlapping* occurrences of each substring. For example, the first two occurrences of `aa` above overlap with each other.

We could do a similar analysis for the string's 3-mers:

```
aaabaab


aaa
 aab
  aba
   baa
    aab
```

In this case, there are four unique 3-mers: `aaa` (appearing once), `aab` (appearing twice), `aba` (appearing once), and `baa` (appearing once).

In this problem, you will write a function `kmers` that, given a potentially large string, a value for k, and some k-mers, will determine how many times each k-mer appears in the string. The function should return a list of integers where the ith integer is the number of times the ith mer in the list of mers occurs in the sequence.

Here are some sample inputs and outputs:

Sequence	K-Mers	Results
aaabaab	ab ba aa	2 1 3

Sequence	K-Mers	Results
aaabaab	aaa aab aaa baa xyz	1 2 1 1 0

Here is the header file for this task:

```
#ifndef _Problem4_
#define _Problem4_
#include <vector>
#include <string>

/**
@class Problem4
*/
class Problem4 {
public:

    /**
     * Takes in a sequence of letters, a kmer length, and a list of k-mers and
     * returns a dictionary of k-mers and their counts in the sequence.
     *
     *  @param seq  A string of letters
     *  @param k    The length of the k-mers represented as an integer
     *  @param mers A vector of names as strings
     *
     *  @return:    A vector where the ith entry is the number
     *   of times that the ith k-mer in kmers occurs on the sequence
    **/
    static std::vector<int> kmers(std::string& seq,
                                  int k,
                                  std::vector<std::string>& mers);
};

#endif
```

And here is the skeleton code for this task:

```
#include "Problem4.h"
#include <vector>
#include <string>
#include <unordered_map>

/**
 * Takes in a sequence of letters, a kmer length, and a list of k-mers and
 * returns a dictionary of k-mers and their counts in the sequence.
 *
 *  @param seq  A string of letters
 *  @param k    The length of the k-mers represented as an integer
 *  @param mers A vector of names as strings
 *
 *  @return:    A vector where the ith entry is the number
 *   of times that the ith k-mer in kmers occurs on the sequence
**/
std::vector<int> Problem4::kmers(std::string& seq,
                                 int k,
                                 std::vector<std::string>& mers) {
    // YOUR CODE HERE
    return std::vector<int>();
}
```

When you take the placement exam, you will be expected to copy the header file and skeleton code into files and then complete the function. For the practice problems, we have provided files `Problem4.h` and `Problem4.cpp` for your convenience.