1 Phonetic Matching

You are playing the role of a newspaper reporter who wishes to build a program that will help flag press releases that come across the wire related to a political candidate that you are covering. Press releases come quite frequently from all of the candidates, so automating the process of identifying those from your candidate will reduce the amount of reading you need to do in order to keep on top of the race. Unfortunately, the candidate you are covering comes from a family that immigrated into the country from a place that didn’t speak English, so the transliterated spelling of their name to English is non-unique from a phonetic point of view. Their name is easy to pronounce but difficult to spell correctly, so press releases frequently contain the name of the candidate spelled slightly incorrectly. Clearly an approach that searches on exact string matches will miss important press releases. Something more creative must be employed.

In researching the problem of identifying misspelled articles, you discover an algorithm from the early 1900s known as "Soundex". It is a basic algorithm in which words are given signatures based on their pronunciation. Words that are misspelled but are similar in pronunciation will be given identical signatures, allowing one to match based on signatures instead of exact string matching. You would like to build a tool that processes text files to identify press releases from your candidate using this algorithm. The algorithm for constructing the signature of a word is as follows:

1. Retain the first letter of the string.

2. Remove all occurrences of the following letters, unless it is the first letter:
   a, e, h, i, o, u, w, y.

3. Assign numbers to the remaining letters (after the first) as follows:
   - b, f, p, v = 1
   - c, g, j, k, q, s, x, z = 2
   - d, t = 3
   - l = 4
   - m, n = 5
   - r = 6

4. If two or more letters with the same number were adjacent in the original name (before step 1), or adjacent except for any intervening h and w then omit all but the first.

5. Return the first four characters, right-padding with zeroes if there are fewer than four.

As an example, both Rupert and Robert yield Soundex signatures “R163”, while Rubin produces “R150”.

As an example, both Rupert and Robert yield Soundex signatures “R163”, while Rubin produces “R150”.

As an example, both Rupert and Robert yield Soundex signatures “R163”, while Rubin produces “R150”.
2 Input specification

The text will have already been tokenized into a stream of words. The input will be fed into STDIN. The first line will contain two integers, the first being the number of candidates \((k)\) and the second being the number of press releases \((p)\). The next \(k\) lines will each contain the proper spelling of each candidate name. This will be followed by \(p\) blocks of words, each of which starts with the number of words \((n)\) within the block, followed by \(n\) lines each containing a single word to process.

3 Output specification

The output of the algorithm will be a table with \(p\) columns and \(k\) rows, with columns separated by a single space. For each document that contains a candidate name via a Soundex match, an X will be placed in the corresponding column of the candidate row. If there is no match, an O (oh, not zero) will be placed instead.

4 Sample input

2 3
knuth
luks
4
knuth
likes
writing
algorithms
3
luks
grades
homework
5
knooth
and
lux
are
misspelled

5 Output for the sample input

X O X
O X X