Element Reaction

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You now have m types of elements. The name of the first element is `a`, the second one is `b`, and so on. You will use these elements to attack monsters.

Specifically, you have a sequence of length n for using elements, and you will use these elements to attack monsters in order. Assuming the current type of element is x, if the monster does not have an element attached to it, then the element x will be attached to the monster. If the monster already has a certain element y, then the two elements will react, causing damage of $a_{y,x}$. After the reaction, only the element x will remain on the monster, and reactions will also occur between elements of the same type.

However, now some types of elements may have become inert, meaning that these elements will not react or attach to the monster when used to attack, and will be ignored directly.

But you do not know which types of elements have become inert, so you want to know for each type of element, in the 2^m possible scenarios of whether it is inert or not, the total damage caused by the element sequence.

Input

The first line contains two integers m, n ($1 \le n \le 10^5, 1 \le m \le 17$).

The next m lines each contain m integers, where the j-th number in the i-th line represents the reaction damage $a_{i,j}$ when the i-th type of element is used before the j-th type of element ($0 \le a_{i,j} \le 10^8$).

The next line contains a string of length n, representing the given element sequence. It is guaranteed that only the first m lowercase English letters appear in the string.

Output

Output a line of 2^m integers, where the i-th number represents the total damage caused by the element sequence when all elements are considered inert as 1 and non-inert as 0, and the binary number obtained by arranging the element types in ascending order from the least significant bit to the most significant bit is $i - 1$.

Examples

standard input	standard output
3 4 1 2 3 4 5 6 7 8 9 abca	15 6 10 0 6 0 1 0
3 10 1 2 3 4 5 6 7 8 9 acbabccbac	47 42 32 27 17 10 2 0