

Problem E. Grammar

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 512 megabytes

Student Z took the course "Principles of Compiler" this semester, and he was very interested in the grammar.

A grammar can be represented by $G = (V_T, V_N, S, P)$.

V_T represents the terminal symbol.

V_N represents a non-terminal symbol.

S represents the start symbol, which is the initial symbol of the derivation process, and can also be regarded as a special non-terminal symbol.

P represents the production. The production is divided into left and right sides connected by " \rightarrow ". It means that the left side of the production can be replaced by the right side. For example, the production which left is S and right is Aa is $S \rightarrow Aa$, means that the symbol string Aa can be used to replace the symbol string S .

A derivation is the process of replacing a non-terminal in a symbol string with the right side of the corresponding production.

The leftmost derivation result of a grammar starts from the start symbol. Each derivation will replace the leftmost non-terminal symbol of the symbol string with the symbol string on the right side of the corresponding production. After several derivations the resulting string of symbols.

For example, for the grammar " $G[S]: S \rightarrow aaBB \ B \rightarrow bb$ ": " a " " b " are terminal symbol, " B " is non-terminal symbol, " S " is start symbol, " $S \rightarrow aaBB$ " and " $B \rightarrow bb$ " is production, then $S \rightarrow aaBB \rightarrow aabbB \rightarrow aabbbb$ is a leftmost derivation of this grammar, " $aabbbb$ " is the result.

Student Z couldn't wait to write a grammar, but the grammar he wrote was relatively simple and met the following conditions:

1. Each terminal symbol of this grammar is a lowercase English letter, so there are only 26 terminal symbols in total.
2. There are a total of n non-terminal symbols, and the i non-terminal symbol is represented by " $[i]$ ".
3. There is only one symbol on the left of each production, and it is a non-terminal symbol.
4. A non-terminal symbol can only appear on the left of a production once, that is to say, there will be a total of n productions in the grammar.
5. The first character on the right side of each production must be a terminal symbol, that is, there will be no such situation as " $[1] \rightarrow [2]a$ ".
6. Since Student Z didn't think about which symbol to use as the start symbol, any non-terminal symbol may become the start symbol of the grammar.

However, because Student Z does not have a good grasp of the rules of grammar, there is a recursive definition in the grammar (that is, it derives itself from a non-terminal symbol), then it may not eventually be possible to generate a symbol string composed entirely of terminal symbols. Therefore, for each leftmost derivation, he will perform at most 100^{100} derivations to ensure that the first 100^{100} characters of the final symbol string are terminal symbols.

Student z doesn't care about the recursive definition, He just wants to know what is the y th terminal symbol in the final string. And he will reselects a non-terminal symbol as the start symbol each time.

Input

The first line of input is a positive integer $T (T \leq 10)$ representing the number of data cases.

For each test cases, the first line inputs two positive integers $n(1 \leq n \leq 1000)$, $q(1 \leq q \leq 1000)$ represents the number of non-terminal symbols and the number of queries. The data guarantees that the total length of n productions does not exceed 100,000 characters.

The next n lines, each with a string describing a production, data guarantee that the i th production has and only has the i th non-terminal symbol on the left.

The next q line, each line has two positive integers $x, y(1 \leq x \leq n, 1 \leq y \leq 10^{18})$ represents a query, x represents the x th non-terminal symbol used as the start symbol, y represents the position of the terminal symbol of the query in this query.

Output

For each query, output a line representing the y -th terminal symbol, and output -1 if the total length of the symbol string is less than y .

Example

standard input	standard output
1	i
3 4	o
[1] ->have[1]ac	e
[2] ->myname[3]id	-1
[3] ->no	
2 9	
3 2	
1 20	
3 3	