Competition 06

Due: October 12

GOOD LUCK, EVERYBODY!

Problem A. After realizing that there is much money to be made in software development, Farmer John has launched a small side-business writing short programs for clients in the local farming industry.

Farmer John's first programming task seems quite simple to him —almost too simple: his client wants him to write a program that takes a number N as input, and prints 17 times N as output. Farmer John has just finished writing this simple program when the client calls him up in a panic and informs him that the input and output both must be expressed as binary numbers, and that these might be quite large.

Please help Farmer John complete his programming task. Given an input number N, written in binary with at most 1000 digits, please write out the binary representation of 17 times N.

input format: The input consists of $\rho \geq 1$ *challenges*, each in the format:

- Line 1: The binary representation of N (at most 1000 digits).

No extra lines intervene between two successive challenges. The last challenge is followed by a line containing a single copy of the character `#`.

sample challenge:

```
10110111
```

sample details: The input is the binary representation of the number 183.

output format: The output should consist of ρ *responses*, each in the format:

- Line 1: The binary representation of N times 17.

No extra lines should intervene between two successive responses.

sample response:

```
110000100111
```

sample details: The binary representation of the number $183 \times 17 = 3111$ is 110000100111.

Problem B. Farmer John has secret message that he wants to hide from his cows; the message is a string of length at least 2 containing only the characters `A`...`Z`.

To encrypt his message, FJ applies a sequence of "operations" to it, where an operation applied to a string S first shortens S by removing either its first or last character, after which the original string S is attached either at the beginning or at the end. For example, a single operation to the string `ABCD` could result in four possible strings:

```
BCDABCD
ABCABCD
ABCDABC
ABCDBCD
```

Given the final encrypted string, please count the number of possible ways FJ could have produced this string using one or more repeated operations applied to some source string. Operations are treated as being distinct even if they give the same encryption of FJ's message. E.g., there are four distinct separate ways to obtain `AAA` from `AA`, corresponding to the four possible operations above.

input format: The input consists of $\rho \geq 1$ *challenges*, each in the format:

- Line 1: A string of length at most 100.

No extra lines intervene between two successive challenges. The last challenge is followed by a line containing a single copy of the character `#`.

sample challenge:

```
ABABA
```

sample details: The result of FJ's operations is the string `ABABA`.

output format: The output should consist of ρ *responses*, each in the format:

- Line 1: The number of different ways FJ could have produced this string by applying one or more successive operations to some source string of length at least 2. If there are no such ways, output zero.

No extra lines should intervene between two successive responses.

sample response:

```
6
```

sample details: Here are the different ways FJ could have produced `ABABA`:

1. Start with `ABA --> AB+ABA`.
2. Start with `ABA --> ABA+BA`.
3. Start with `AB --> AB+A --> AB+ABA`.
4. Start with `AB --> AB+A --> ABA+BA`.
5. Start with `BA --> A+BA --> AB+ABA`.
6. Start with `BA --> A+BA --> ABA+BA`.

Problem C. Farmer John has N cows ($2 \leq N \leq 15$), each of which is one of three different breeds: Holsteins, Jerseys, or Guernseys.

Unfortunately, FJ cannot remember the exact breeds of his cows! He does, however, remember a list of K ($1 \leq K \leq 50$) relationships between pairs of cows; for example, he might remember that cows 1 and 2 have the same breed, or that cows 1 and 5 have different breeds.

Given FJ's list of relationships between pairs of cows, please help him compute the number of different possible assignments of breeds to his cows (this number could be zero if his list contains contradictory information).

input format: The input consists of $\rho \geq 1$ *challenges*, each in the format:

- Line 1: Two space-separated integers: N and K.
- Lines $2 \ldots 1{+}K$: Each line describes the relationship between a pair of cows x and y ($1 \leq x, y \leq N$, $x \neq y$). It is either of the form "`S x y`", meaning that x and y have the same breed, or "`D x y`", meaning that x and y have different breeds.

No extra lines intervene between two successive challenges. The last challenge is followed by a line containing a single copy of the character `#`.

sample challenge:

```
4 2
S 1 2
D 1 3
```

sample details: There are 4 cows. Cows 1 and 2 have the same breed, and cows 1 and 3 have different breeds.

output format: The output should consist of ρ *responses*, each in the format:

- Line 1: The number of possible breed assignments.

No extra lines should intervene between two successive responses.

sample response:

```
18
```

sample details: The following six breed assignments are possible for the first 3 cows: `HHG`, `HHJ`, `GGH`, `GGJ`, `JJH`, `JJG`. For each of these, we can have 3 possible breed assignments for the 4th cow, giving a total of 18 different assignments consistent with FJ's list.