A. GOODS DELIVERY

"giaohangtocdo.vn" is a start-up company. Every day, the company delivers many kinds of products for n retail stores, called 1, 2...n. All stores are located on a long street: store i at position p_i on that street.

For simplicity, we assume that all kinds of products have the same size, same weight, and store i needs a fixed number n_i products each day.

The company uses many trucks to deliver products. Each truck can bring maximum k products. A truck can also deliver products for some stores in a trip. The company plans to build a grand-warehouse where all trucks pick up products and start a trip to the stores. They want to choose a location on the street to build the grand-warehouse to optimize the total distance of all trips in a day.

INPUT

The first line contains integer numbers n ($1 < n < 10^5$) and k ($0 < n < 10^9$).

There are n lines after that, where line i contains data for store i, that are p_i ($0 < p_i < 10^9$) and n_i ($0 < n_i < 10^9$) respectively.

OUTPUT

Print the total distance of all trips (one-way delivery) in a day.

EXAMPLE

Input	Output
6 5 1 7 2 2 3 6 8 9 10 11 15 13	44

B. RAILWAYS

Due to recently fatal train collision, Vietnam Railway company(VNR) decides to reconstruct its train lines. A joint group of scientists have been invited to analyze the state of railway infrastructure. They first attempt to model the main North-South railway line. In the simplest version, this line is concerned as a single line train which has only one rail line and three main stations: Hanoi, Danang and Saigon. Everyday, there are N trains on this line always departs from Hanoi Station and bound for Saigon Station in fixed order from train T_1, T_2,... T_N.

However those train won't necessarily arrive at the same order as they depart. The Railway Company may tweaking train arrival order by schedule train T_i to stop at Danang and wait for m ($0 \leq m \leq N - i$) train immediately after it, from T_{i+1}, T_{i+2}, …, T_{i+m} to pass before continue on its way.

Now this strategy of scheduling is not optimal but it is easy to implement. At Danang we build a railway switch, train that have to wait will go into the diverging track, stop and wait there, when it's time for that train to continue, it simply reverses back to the main track and continue to Saigon. This diverging track also serves as a stack of first come, last leave order, ensure that train won't have anyway to go before all the train it was supposed to wait for has passed.

But implement an easy system is not what you are here for right? As talented young scientist and engineer, your task is to calculate how many different order those N train may arrive at Saigon.

INPUT

The input consists only one positive number N ($0 < N < 37$).

OUTPUT

Number of different orders of N trains at Saigon station.

EXAMPLE

Input	Output
1	1
3	5
4	14

C. SIMPLE CACHE

Cache in computer science is a small, very fast but very expensive memory used in conjunction with a larger, cheaper but slower memory (the main memory). Cache speed up its slower companion by storing copies of frequently accessed data. There are many algorithm to manage what data going in and out of the cache and when, but that was the problem for another time. This time we only ask you to implement the basic book keeping of the cache.

Each block of data is referred to by their *address* in the main memory. There are three primary task we have to perform in the cache: Insert data into the cache, search for data in the cache and remove data from the cache.

Our cache hold data in a Direct access table (DAT) capable of holding exactly M - 1 blocks of data. To determine the position for each block of data, we have a hash function h(*address*) that return an index in the range of [0, M - 1]. Since the value for address can be much larger than the value of M, it's possible that two different can be hashed in to the same position. To avoid collision we simply find closest position after the initial hash position and wrap around the table if necessary.

Let's implement a caching system with fixed size of M, and the hash function

h(hadress) = adress % M

Starting with with an empty cache and process 2 kind of request:

1. Search for the position of an address in the DAT. If that address has not been cached in DAT and we still have empty space in DAT, insert it into DAT
2. Remove and address from DAT.

INPUT

Input has multiple test cases, each test case consists of:

- The first line contains M ($0 < M \leq 10^6$), the fixed size of DAT and N ($0 < N \leq 10^5$), the number of operations.
- N lines that follow each start with either letter 'R' or 'C' then follow by a positive integer number *address* This is the operation that have to be processed.

OUTPUT

In each test case, following:

- With operation 'R', if the block of data at the *address* has been cached print out 'A' follow by one space then he position of this address in the cache. If the block of

data has not been cached, print out letter 'S' follow by one space and its position after insertion into the cache, if cache is already full cancel the insertion and report that address with position -1.

- With operation 'C', if the block of data has been cached, delete it and print out letter ‘D’; otherwise print out letter U

EXAMPLE

Input	Output	Explanation (for test case #1 only)
5 7 R 89 R 30 R 100 R 20 R 10 C 20 C 50 10 8 R 3 R 13 R 23 C 13 R 23 C 3 R 13 R 13	S 4 S 0 S 1 S 2 S -1 D U S 3 S 4 S 5 D A 5 D S 3 A 3	89 will be inserted at 89%5 → 4 30 will be inserted at 30%5 → 0 100 should be 100%5 → 0, but collide with 30 → 1 20 20%5 → 0,collide with 100 → 1,.. → 2 10 we already have 5-1 block in the cache 20 was deleted 50 has not been cached, did not delete.

D. DANCE RECITAL

ULL dance group has been invited to perform a recital at the first UIT pageant show. This is a very big show and they will have many dancers who perform more than one routine. Each routine require a different costume and dancers will have to go backstage to change between routines.

Now it would be normal if a dancer perform in two routine that are not consecutive, he or she can change when others perform the next routine and have plenty of time waiting in backstage. But when adancer have two consecutive routine he or she will have to do a "rush change" and things may go wrong. Dancers may fail to change in time and delay the routine or worse they may damage the costume while changing in a rush.

The manager of ULL dance group want to minimize the risk for their recital. Now he has the cast of this recital at hand. Each dancer assigned an uppercase letter (ULL dance groups has never grown to have more than 26 dancers), each routine is a strings of characters defining which dancer appears in that routine, and the recital is a list of routine. The manager have the authority to change the order of routine in the recital, let's help him calculate the minimal number or "rush change" his dancers will have to do. For example given the following recital of 5 routines and 8 dancers:

ABC
ABEF
DEF
ABCDE
FGH

The first routine includes dancers {A, B, and C}, the second routine includes dancers {A, B, E, and F}, and so, dancers A and B will each require a rush change between the routines. In fact, if these five routines are scheduled in the order given above, a total of six rush changes are required. However, the schedule can be rearranged as follows:

ABEF
DEF
ABC
FGH
ABCDE

In this case, only two rush changes are required (those for E and F between the first two dances).

INPUT

Input contains multiple test cases. Each test cases have the following format:
The first line contains a single integer R, with $2 \leq R \leq 11$, that indicates the number of routines in the recital. Following that will be R additional lines, each describing the dancers for one routine in the form of a nonempty string of up to 26 non-repeating, lexicographically sorted uppercase alphabetic characters identifying the dancers who perform in that routine. Although a dancer's letter will not appear more than once in a single routine, that dancer may appear in many different routines, and it may be that two or more routines have the identical set of dancers.

OUTPUT

For each test case, output a single integer designating the minimum number of quick changes required for the recital.

EXAMPLE

Input	Output
5 ABC ABEF DEF ABCDE FGH 6 BDE FGH DEF ABC BDE ABEF 4 XYZ XYZ ABYZ Z	2 3 4

E. FIND WORD

String search is a class of problems that have been carefully investigated since the early age of computer. The most basic form of string search problem is searching for words (a typically short string) inside the haystack (a very long string). The occurrence of a word is defined as the number of characters precede it in the haystack.

We refine the problem definition further by defined words as a string of alphanumeric characters and underscore '_' only, and thus, the haystack will be a sequence of words separated by non-word character(s). For example given the haystack:

Welcome ACM_UIT 9

The word *ACM_UIT* occurs at position 8 but the word *ACM* does not occur. Let's implement an algorithm for such problem

INPUT

Input contains multiple test cases. Each test case consists of :

The first line is the haystack. You can be assured that haystack will not contain more than one million characters

Subsequent lines, each line contain one word (at most one thousand characters long) to be searched for. Test case will end with 10 equals sign (==========).

OUTPUT

In each test case, following:

For each word, output on one line all of its occurrences in the haystack, each occurrences separate by one space.

EXAMPLE

Input	Output
Welcome ACM_UIT 9 Welcome ACM ACM_UIT ========== the quick brown fox jumps over the lazy dog the quick brown Fox jumps over lazy dog ==========	0 -1 8 0 31 4 10 -1 20 26 35 40

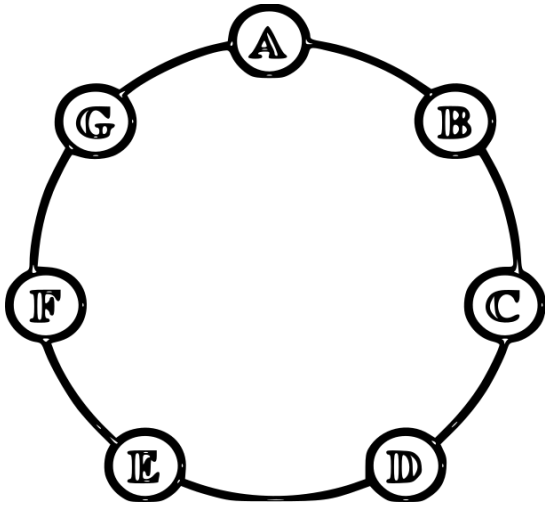
F. COUNTER TERROIST

Guerrilla warfare is a popular choice for small armed-forces when facing overwhelming enemies. Relying on mobility, terrain cover and population support. Guerilla tactics has helped many revolutionary forces liberate their country. But now, the ISIS is abusing guerilla tactics to prolong the fight and terrorizes civilians in the Middle East.

You were employed by the UN peacekeeping general's staff to help planning an attack to destroy regiment of ISIS forces in the Far East. The operation map is modeled as an undirected graph of N vertices, where ISIS is hiding. V infantry battalions will parachute into a subset of V distinct vertices with order to find and engage ISIS. If the ISIS is not hiding in those V initial vertices however, our battalions will have to move to along the edge from one vertex to another to find them. It takes exactly 1 weeks to move from one vertex to its neighbor in this terrain and you only have enough air support for 1 battalion to engage ISIS at a time. So every week only 1 battalion can move while others battalion have to stay in defensive position.

Now you don't want to underestimate our enemies, suppose that ISIS is smart, they can choose a hideout far from your dropzone and every week they too can either stay put or move to another vertex. As a planner you will have to calculate the minimum time in weeks it would take to destroy the enemy, providing that he always makes the smartest move.

Take a look at the graph below. It would be impossible for one battalion to catch ISIS force. They can stay two vertices away from the drop zone and keep moving with the same speed, staying two vertices away. But with two battalions drop in vertices A and D for example, no matter how smart the ISIS move, you can catch them in two weeks.



INPUT

Input contains many tests. Each test begins with a line contains 3 integers: V N E where $1 \le V \le 7$ is the number of battalions at your disposal, $V+1 \le N \le 15$ is the number vertices in the maps and $1 \le E \le 45$ is the number of edges connecting those vertices. Following that line is a line listing every edges in the map. Vertices are named with the first N character in the alphabet and edges are denoted by a string of exactly two characters. You can safely assume that this is a simple graph and each vertex has at most 6 incident edges.

OUTPUT

For each test case, output in a line the word "Case" follows by the case number (starting at 1) follows by a colon and a space, finally, follows by the minimal number of weeks it takes to guarantee ISIS destruction or the word "NEVER" if our battalions

EXAMPLE

Input	Output
1 7 7 AB BC CD DE EF FG GA 2 7 7 AB BC CD DE EF FG GA 1 5 6 AB AC BC BD DE EC 2 5 6 AB AC BC BD DE EC 2 10 15 AB BC CD DE EA AF BG CH DI EJ FH HJ JG GI IF 3 10 15 AB BC CD DE EA AF BG CH DI EJ FH HJ JG GI IF 3 14 10 AB BC CD EF FG GH IJ JK LM MN 4 14 10 AB BC CD EF FG GH IJ JK LM MN 0	CASE 1: NEVER CASE 2: 2 CASE 3: NEVER CASE 4: 1 CASE 5: NEVER CASE 6: 1 CASE 7: NEVER CASE 8: 2