

Problem A Child's Play

Input: /users/acm/data/integers.in

Output: standard output

Anu is a little girl who has an extraordinary skill in playing with numbers. You give her a set of nonzero positive integers with or without repetitions and tell her to generate all possible distinct nonzero positive integers by retaining / adding / subtracting any combination of the given integers. She can do so instantly.

For example if you give her the three integers 2, 8, 3 then in a moment she will tell you that a total of 11 integers can be generated. In fact, the 11 integers are the following: 1, 2, 3, 5, 6, 7, 8, 9, 10, 11 and 13.

You are required to write a program that equals the skill of Anu.

Input

The input may contain multiple test cases.

The first line of each test case gives two integers: the case number c and the total number k (< 15) of integers given to Anu. The next line gives k integers in any order. A blank character separates two consecutive integers in input.

The input terminates with an input 0 for c .

Output

For each test case print output in one line, giving the case number c and the total number of integers that can be generated.

Sample Input

```
1 2
2 3
2 3
2 3 8
3 4
2 5 5 4
0
```

Sample Output

```
1 4
2 11
3 14
```

Problem B

Mouli Search

Input : /users/acm/data/preference.in

Output : standard output

Mouli.com plans to implement a search engine that can list web sites/pages (SP) in order of anticipated preference of a user. It needs your service to write a program for this purpose.

When a visitor to Mouli.com registers as a user, the server creates on visitor's hard disk, a small data file named Moulicookie. The file contains personal information and preferences of the user. The server exchanges information automatically between the server and the browser and tracks user's actions and preferences. Using all these information the server categorizes a user into k categories, k being an integer. The number k may be different for different user. The categories of a user are identified by integers 1, 2, .. k .

The server has a large database file named Moulicookbook that contains statistical information on visits to SP. The file also includes a grade A, B, C or D for anticipated preference of each SP with respect to each possible category of a user. The grade A is the highest possible grade while B, C and D are the other grades in decreasing order of preferences. The server updates the file after every visit to an SP.

When a user makes a search request the server selects all SP in Moulicookbook that correspond to the request. Let there be n selected SP. Using statistical information, the server assigns a distinct identifying rank to each selected SP. It constructs a two-dimensional array P , where the element $P(i, j)$, $i=1, 2, \dots, n$, $j=1, 2, \dots, k$ contains the grade of the selected SP having rank i with respect to category j in Moulicookie.

Given two SP s_1 and s_2 , s_1 has higher preference than s_2 if the total number of categories having higher grades in s_1 is greater than that in s_2 . In case of a tie s_1 is considered to have higher preference than s_2 if $s_1 < s_2$. It is possible to have three SP: s_1, s_2, s_3 where s_1 has higher preference than s_2 , s_2 has higher preference than s_3 but s_3 has higher preference than s_1 . In such a case, the set of SP: $\{s_1, s_2, s_3\}$ forms an equivalent set. If an SP is equivalent to two or more SP in an equivalent set then it is equivalent to all SP belonging to the set. All SP in an equivalent set have the same anticipated preference.

Given P corresponding to a search request as input, you are required to write a program to list all SP in order of anticipated preference.

Input

The input may contain multiple test cases.

The first line of each test case contains the case number c , the total number n (<100) of selected SP from Moulicookbook and the total number k (<20) of categories in Moulicookie.

The next n lines give elements of P . The i -th line gives the i -th row of P . A blank character separates two consecutive elements in a row.

The input terminates with an input 0 for c .

Output

For each test case, first print in one line, the test case number c .

In the following lines, print identifying rank of SP in order of anticipated preference. Each line contains either rank of one SP or ranks of all SP belonging to an equivalent set in increasing order of ranks.

Print a blank line between two successive test cases.

Sample Input

1 5 3
B C A
A C D
C A C
A B C
B A C
2 6 3
A C D
C A C
B A C
A B C
B C A
A B B
0

Sample Output

1
1 3 4 5
2

2
6
3
1 2 4 5

Problem C Shunting Satabdi Trains

Input: /users/acm/data/assembly.in

Output: standard output

Satabdi trains offer catering services to all its passengers. For convenience of catering services as well as for hygienic considerations, these trains use specially designed railway carriages. Each of these carriages has exclusive facilities for Catering services (C) at one end and Toilets (T) at the other end. Carriages are linked end to end so that ends on both sides of a link have the same kind of facilities, either C or T.

In factory carriages are linked end to end in random order to form assembly of carriages. An assembly of carriages has a front and a rear. Let $\rightarrow A \rightarrow$ or $\leftarrow A \leftarrow$ denote an assembly A where an arrow to A indicates the front and an arrow from A indicates the rear of the assembly. A carriage in an assembly has an orientation; it is the order of the two ends with respect to the front of the assembly viz., either CT or TC. It should be noted that two carriages linked to each other in a Satabdi train have opposite orientations, either CT followed by TC or TC followed by CT. An assembly $\rightarrow A \rightarrow$ may be reversed changing front to rear and rear to front without altering any link. If $\leftarrow r(A) \leftarrow$ denotes the reverse of $\rightarrow A \rightarrow$ then the orientation of each carriage in $\leftarrow r(A) \leftarrow$ is opposite to that of the same carriage in $\rightarrow A \rightarrow$; this is illustrated below.

$\rightarrow A \rightarrow$: $\rightarrow CT \rightarrow TC \rightarrow CT \rightarrow CT \rightarrow CT \rightarrow$ $\leftarrow r(A) \leftarrow$: $\leftarrow CT \leftarrow TC \leftarrow CT \leftarrow CT \leftarrow CT \leftarrow$

An assembly $\rightarrow A \rightarrow$ may be considered to be an assembly of two or more sub-assemblies, e.g., $\rightarrow A_1 \rightarrow A_2 \rightarrow$ or $\rightarrow A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow$.

In railway yards an assembly of carriages is reassembled as per the requirement of Satabdi trains using certain shunting operations. The operations are based on de-linking a subassembly at one or two links, shunting a sub-assembly and attaching it to front/rear of another subassembly either with or without reversing. A list of feasible shunting operations performed at a railway yard is given below:

OP	Operations	Assembly before	Assembly after
1	Delink & Attach	$\rightarrow A_1 \rightarrow A_2 \rightarrow$	$\rightarrow A_2 \rightarrow A_1 \rightarrow$
2	Double Delink & Attach	$\rightarrow A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow$	$\rightarrow A_2 \rightarrow A_1 \rightarrow A_3 \rightarrow$ or $\rightarrow A_1 \rightarrow A_3 \rightarrow A_2 \rightarrow$
3	Delink, Reverse & Attach	$\rightarrow A_1 \rightarrow A_2 \rightarrow$	$\rightarrow r(A_1) \rightarrow A_2 \rightarrow$ or $\rightarrow A_2 \rightarrow r(A_1) \rightarrow$ or $\rightarrow A_1 \rightarrow r(A_2) \rightarrow$ or $\rightarrow r(A_2) \rightarrow A_1 \rightarrow$

Let c_1 , c_2 and c_3 be the cost of operations (OP) 1, 2 and 3 respectively, where $c_1 < c_2 < c_3$. You are required to write a program that will compute the minimum cost c of reassembling a factory-assembled assembly of carriages into an assembly of carriages suitable for Satabdi trains.

Input

The input may contain multiple test cases.

The first line of each test case gives an integer k denoting the case number and three real numbers c_1 , c_2 and c_3 .

The next line gives the orientation of carriages in a factory-assembled assembly of carriages in the order in which they appear from front to rear in the assembly. A blank character separates orientations of two neighbouring carriages.

The input terminates with an input 0 for k .

Output

For each test case in input, print output in one line, giving the test case number k and the minimum cost c of reassembling. In case an input does not conform to the requirements of Satabdi trains then print the comment "invalid" after case number k.

Sample Input

```
1 1.5 3.5 7.5
CT TC TC CT CT TC TC
2 2.0 1.5 4.5
CT TC TC CT TT TC
3 1.0 4.0 8.0
TC TC TC TC TC CT TC
4 1.0 2.0 3.0
TC CT TC CT TC CT
0
```

Sample Output

```
1 5.0
2 invalid
3 13.0
4 0.0
```

Problem D
Smallest Rectangle

Input: /users/acm/data/points.in

Output: standard output

You have a plain and simple rectangle to find. Given a set of points (x_i, y_i) , $i=1, 2, \dots, n$, you are required to write a program that finds the smallest rectangle, smallest in area, so that all given points are either inside or on the boundary of the rectangle and at least two of the points lie on any one of the sides of the rectangle. If there is more than one smallest rectangle then find any one of them.

Input

The input may contain multiple test cases. For each test case, the first line contains the case number c and the total number n (<50) of given points. Each of the next n lines gives the x -coordinate and the y -coordinate of a given point. A blank character separates two input values in a line.

The input terminates with an input 0 for c .

Output

For each test case first print in one line, the case number c and the area of the smallest rectangle.

The next four lines contain coordinates of four vertices of any one of the possible smallest rectangles. The vertices are arranged in ascending order of x -coordinates. In case x -coordinates of any two vertices are equal then the two vertices are arranged in ascending order of y -coordinates.

Retain two digits after the decimal point in each computed real value. Keep a blank line between outputs of two test cases.

Sample Input

```
1 6
1.0 1.2
1.0 1.5
1.3 1.0
1.5 3.0
5.0 1.0
5.0 2.9
2 12
1.0 1.5
1.0 2.5
1.5 1.0
2.0 3.0
3.0 1.0
4.0 7.0
4.5 -1.0
5.0 6.5
5.5 9.0
6.0 1.0
6.5 8.5
8.0 8.0
0
```

Sample Output

```
1 8.00
1.00 1.00
1.00 3.00
5.00 1.00
5.00 3.00

2 47.08
0.46 1.69
4.62 -1.08
5.69 9.54
9.85 6.77
```

Problem E
Least Imprecise Pair

Input: /users/acm/data/string.in

Output: standard output

Consider a string S of n nonnegative integers and two sub-strings X, Y in S , each of length m . Let a string or a sub-string T of length k be represented by an array T with elements t_1, t_2, \dots, t_k . If the starting position of a sub-string Z in T is i then $t_i = z_1$.

Sub-strings X and Y match each other precisely, if $x_i = y_i$ for $i=1, 2, \dots, m$. They match each other imprecisely with imprecision index α , where

$$\alpha = \frac{1}{\sqrt{m}}(h + \delta), \quad \delta = \frac{1}{m} \sum_{i=1}^m (|x_i - y_i|)$$

and h is a predefined real number. The least imprecise pair of sub-strings in S is defined to be the pair of sub-strings X, Y that has least value of α . If there is more than one pair of X, Y for which α is least then the least imprecise pair satisfies the following conditions in order of priority:

1. The length m is as large as possible,
2. The starting point of X is closest to that of S .
3. The starting point of Y is furthest from that of X .

You are required to write a program that finds the least imprecise pair of sub-strings X, Y for a given string S and a given value of h .

Input

The input may contain multiple test cases. The first line of each test case gives two integers and a real number: the case number c , the length n (<100) of S and the value of h respectively. The next line gives elements of S separated by a blank character. The input terminates with an input 0 for c .

Output

For each test case print output in one line, giving the test case number c , the length m of the least imprecise pair of sub-strings X and Y , the starting position i of X in S , the starting position j of Y in S and the least value of α . Retain four digits after the decimal point in the least value of α .

Sample Input

```
1 8 0.0
5 8 7 6 8 6 9 6
2 8 1.0
5 8 7 6 8 6 9 6
3 8 2.0
5 8 7 6 8 6 9 6
0
```

Sample Output

```
1 3 4 6 0.1925
2 4 3 5 0.7500
3 6 1 3 1.2247
```

Problem F
Hidden Digits Multiplication

Input: /users/acm/data/steps.in

Output: standard output

Consider steps of multiplication of an unsigned decimal integer X by another unsigned decimal integer Y. If Y is of m digits and the digit in position i of Y is y_i , $i=0, 1, \dots, (m-1)$ then steps of multiplication consist in finding m intermediate products $P_i = X \cdot y_i$, $i=0, 1, \dots, (m-1)$, positioning them appropriately and adding them to get the product $Z=X \cdot Y$. The steps are illustrated below for a simple example.

Steps with all digits visible	Steps in general	Steps with some digits hidden	Steps with all digits hidden
<pre style="margin: 0;"> 5 3 9 2 4 3 ----- 1 6 1 7 6 2 1 5 6 8 ----- 2 3 1 8 5 6</pre>	<pre style="margin: 0;"> X Y ----- P₀ P_(m-1) ----- Z</pre>	<pre style="margin: 0;"> c 3 9 b 4 3 ----- a d a 7 d b a c d 8 ----- b 3 a 8 c d</pre>	<pre style="margin: 0;"> a b c d e b ----- f g f h g d f a g i ----- d b f i a g</pre>

Given steps of a multiplication with some or all digits hidden using letters, you are required to write a program that finds X, Y and Z. Assume that each letter represents a hidden digit only and no visible digit is hidden in a letter. Assume further that no two letters represent the same digit.

Input

The input may contain multiple test cases.

For each test case, the first line contains the test case number c and the number m ($1 \leq m \leq 10$), of digits in Y.

The next (m+3) lines give X, Y, P_i , $i=0, 1, \dots, (m-1)$ and Z respectively, with some or all digits hidden.

The input terminates with an input 0 for case number c.

Output

For each test case, print in one line, the test case number c and the total number k of possible sets of values of X, Y, Z.

Each of next k lines gives one set of values of X, Y and Z. A blank character separates two values in a set. The k lines are arranged in lexicographic order of X and Y.

Print a blank line between two successive test cases.

Sample Input

```
1 2
9
a3
bc
de
dfc
2 3
9
a2x
x
bc
de
dfcx
3 2
c39b
43
ada7d
bacd8
b3a8cd
0
```

Sample Output

```
1 0

2 2
9 420 3780
9 720 6480

3 1
5392 43 231856
```

Problem G Weapons of Mass Destruction

Input: /users/acm/data/wmdcodes.in

Output: standard output

Destruction of Weapons of Mass Destruction (WMD) was the only justification of war against a country Q by coalition forces of countries K & S. However even after taking full control of country Q, coalition forces failed to uncover any significant WMD. The question is: where are the WMD?

A highly reliable intelligence source has the information that the previous Government of country Q has hidden the code of locations of WMD so cunningly that best efforts of the allies have failed to locate them. The same intelligence source has discovered the secret code and the clue to decipher them.

The code consists of a set of strings and a square grid, both of capital letters. Each letter used in the code represents a known geographical location in country Q. A string is said to appear on the grid if it is formed by reading letters on the grid, horizontally (row-wise, left to right), vertically (column-wise, top to bottom) or diagonally (upper left to lower right), starting from any position on the grid.

Locations of WMD are identified by the unique and minimal set of distinct letters on the grid satisfying the following conditions:

1. Each letter in the set is required to be replaced at a position on the grid by another letter so that all strings appear on the grid after the replacement.
2. A string appears on the grid with at most one changed letter in it.

If conditions stated above do not lead to identification of a unique and minimal set of distinct letters on the grid then there is no existence of WMD.

You are required to save the world from mass destruction by writing a program that deciphers the secret code and identifies the locations of WMD.

As an illustration, consider the grid of letters shown on the right. Strings BDSC and KDMC appear on the grid when each of two distinct letters B and A is replaced by C; DMK, MMS and IMD all appear with the unique and minimal change: N to M; BDND appears in two ways, either vertically or diagonally, and the change required in each case is a unique one viz., K to D at a position on the grid. However both BDNK and NIND appear on the grid without any change of letters; to read BDND and BSND both, there exists no distinct set to change; both NIND and SMKS cannot be read making any change; and there exists no unique change to read IMD.

B	D	S	B
K	D	M	A
N	I	N	D
D	H	S	K

Input

The input may contain multiple test cases.

The first line of each test case gives three integers: the case number c , the total number n (<50) of strings and the number m (<15) that defines the size m^2 of the square grid.

The second line gives n strings separated by blank character.

The next m lines contain letters on the grid: the i^{th} line gives letters in the i^{th} row. A blank character separates two consecutive letters in a row.

The input terminates with an input 0 for c .

Output

For each test case, print in one line, the test case number c and the total number k , of locations of WMD. If k is equal to zero then there is no more output.

If k is nonzero then print in one line, all k letters representing locations of WMD in alphabetic order. Use blank character to separate two letters in the line.

Print a blank line between outputs of two test cases.

Sample Input

1 5 4

BDSC KDMC DMK MMS IMD

B D S B

K D M A

N I N D

D H S K

2 5 4

BDNK NIND BDND BSND SMKS

B D S B

K D M A

N I N D

D H S K

0

Sample Output

1 3

A B N

2 0

Problem H Invaders from Space

Input: /users/acm/data/positions.in

Output: standard output

After 50 years from now sociopolitical scenario on the earth is likely to be much different from what it is today. It is likely that UN will have to defend the earth against invaders from space.

Assume that a set of m invaders has taken up relatively stationary strategic positions in space with the intention to invade the earth. They have placed themselves in m distinct positions in a relatively stationary 3D cubic mesh (3DCM) of size n^3 . Let a position p in 3DCM be identified by a 3D index (i, j, k) where each of the indices i, j, k assumes values $0, 1, \dots, (n-1)$. Armed with sophisticated laser beams an invader at a position p in 3DCM is capable of destroying a target at another position q if two of the three indices of q are identical to the corresponding indices of p . For example an invader at $(0, 5, 4)$ can destroy a target at $(0, 1, 4)$. In such a case the position q is exposed to an invader at p .

For defense UN uses its advanced Satellite Robots System (SRS). Each robot in SRS is equipped with advanced nuclear missile technology and is capable of destroying targets from relatively stationary positions with respect to a set of relatively stationary objects in space. When placed at a position p in 3DCM mentioned above a robot can destroy a target at another position q if any of the following conditions is true:

1. At least one index of q is identical to the corresponding index of p , e.g., $p=(0, 5, 4)$, $q=(1, 5, 2)$ or $(0, 1, 4)$.
2. Each index of q differs from the corresponding index of p by the same magnitude e.g., $p=(0, 5, 4)$, $q=(3, 2, 1)$.

In such a case the position q is exposed to a robot at p .

For defense UN decides to deploy maximum number of robots according to the following strategy:

1. Robots are deployed in groups of n robots,
2. Each robot in a group occupies a distinct defensive position in 3DCM. The position is different from positions occupied by either invaders or other robots.
3. A robot is exposed neither to an invader nor to another robot of the same group. However robots belonging to different groups may be exposed to each other.

Given m positions of invaders in 3DCM you are required to write a program that finds the maximum number of robots that can be deployed. The program should also identify distinct defensive positions for robots in each group.

Input

The input may contain multiple test cases.

For each test case, the first line contains the case number c , the size parameter n (<10) of 3DCM and the total number m of invaders. Each of the next m lines gives three integers i, j, k identifying the position (i, j, k) of an invader in 3DCM. The integers are separated by blank character.

The input terminates with an input 0 for c .

Output

For each test case, the first line contains the test case number c and the total number g , of groups of robots that can be deployed. If g is equal to zero then there is no more output.

In case g is not equal to zero then print n more lines as output. Each of these lines contains g positions of robots. As in input a position (i, j, k) in 3DCM is identified by integers i, j, k . The r^{th} position in the i^{th} line corresponds to the position (i, j, k) of the robot belonging to the r^{th} group for which the first index is i , for $r=1, 2, \dots, g$ and $i=0, 1, \dots, (n-1)$. One blank character separates integers identifying a position and two or more blank characters separate positions in a line.

Print a blank line between two successive test cases.

Sample Input

1 3 2
0 2 1
1 0 1
2 4 6
0 2 0
0 3 3
2 0 3
2 2 2
3 0 0
3 1 1
0

Sample Output

1 0

2 2
0 0 1 0 0 2
1 2 3 1 1 0
2 1 0 2 3 1
3 3 2 3 2 3