Problem A: Apple Trees

Input File: A.in

Apple trees in a garden form a rectangular grid. A group of monkeys lives on these trees. Not more than one monkey lives in each tree. A monkey feels happy to believe that it is the lord of all apple trees it views from the top of the tree it lives in. However, as the trees are of different heights a monkey can view trees that are not obstructed from its view by other trees. A monkey that views the maximum number T of trees is a lord of lords. The undisputed monarch is the lord of lords that lives in the highest tree. In case two or more lord of lords have the claim to be the monarch then monarchy is disputed.

Assume that the ground of the garden is plane; vertical straight lines represent trees; trees are at a distance of unity row-wise or column-wise; and the height of each tree is an integer in the unit of the distance. A tree of height zero in a position indicates absence of a tree on that position. Visibility of a tree R from the top of a tree P depends on heights and locations of P, R and other trees. A tree R is invisible from the top of a tree P if and only if there exists a visible tree Q that lies on the vertical plane containing P and R and is located between P and R so that the top of R is either on or below, the line joining tops of P and Q.

Write a program to locate the undisputed monarch, given heights of all trees.

Input

The input may contain multiple test cases.

For each test case, the first line gives two integers m and n representing respectively the total number of rows and columns of trees in the garden. Each of the next m lines contains n integers representing heights of trees in a row. The j^{th} integer in the i^{th} row represents the height of the tree in i^{th} row and j^{th} column of the garden. Assume that the garden contains no more than 200 trees.

The input terminates with an input 0 as the first input for a test case.

Output

For each test case output three integers r, c and T in one line. Integers r and c represent the location of the undisputed monarch, where r is the row number and c is the column number of the tree on which the monarch lives. In case monarchy is disputed, both r and c are zero. The integer T represents the total number of trees visible to a lord of lords.

Sample Input

```
3 3
1 2 3
4 5 6
7 0 9
3 4
4 4 4 4
4 4 4 4
4 4 4 4
1 10
5 3 2 1 6 4 2 8 7 1
0
```

Sample Output

```
2 2 7
0 0 10
1 5 6
```

Problem B: Bio-HRQ-Comparator
Input File: B.in

Bio-HRQ-Comparator is a fully automatic computer based neurological scanning system that scans brains of three persons simultaneously and ranks them 1, 2, 3 with respect to their HRQ (Human Resource Quotient) for a given activity. Bio-HRQ-Comparator has three specially designed chairs each fitted with brain scanning devise. Exactly three persons are to sit on the chairs and just think independently and simultaneously on the best possible solution of a given problem related to an activity for which HRQ is tested. The thinking process continues for a specified duration of time that is dependent on the complexity of the given problem. Bio-HRQ-Comparator captures the thinking process and ranks them with respect to their HRQ without any further scrutiny. Through rigorous testing and analysis, ranking by Bio-HRQ-comparator has been found to be so precise that no two ranked persons are identified to have different rank order or found to have the same HRQ.

The system has the potential to replace the traditional method of selection of candidates through interview, where the basic problem is to arrive at total ordering and ranking of a given set of n (assume $n \leq 20$) candidates with respect to their HRQ for a specific job. It is proposed to select arbitrarily k sets of three candidates each and rank candidates in each set using Bio-HRQ-comparator hoping that total ordering and ranking of candidates can be done successfully. However it is not always possible to arrive at total ordering and ranking of all n candidates using arbitrarily selected k sets of candidates.

You are required to write a program that either ranks all n candidates, if possible, using the k results obtained so far or determines the minimum number m of additional Bio-HRQ-comparator testing required for determining the total ordering and ranking of all n candidates. In case m additional testing are required, the program should identify m sets of three candidates each, for additional Bio-HRQ-comparator testing. Assume for simplicity that a candidate is not required to appear more than twice for additional testing.

Input

Input may contain multiple test cases. Each test case contains two lines.

The 1st line gives n and a string of n letters. Each letter in the string identifies a candidate and letters appear in an arbitrary order.

The 2nd line gives k results of Bio-HRQ-comparator testing. The first field is the integer k and it is followed by k strings of three letters each representing k results. The three letters in a string appear in order of ranks 1, 2 and 3 of candidates represented by the letters.

Input terminates with a line that contains 0 (zero) as the first and the only character.

Output

For each test case, print output in one line.

The first field in the line gives the integer m. If m is equal to 0 then a string of n letters follows it; the string represents the total rank order of all n candidates.

Otherwise m strings each of length three follow the integer m. Each string contains either three letters or two letters followed by an asterisk (*). Each letter identifies a specified candidate while an asterisk represents an option to have any candidate. Letters in a string appear in alphabetic order and strings appear in lexicographic order.

Sample Input

```
5 axdpf
3 adf xdp axp
5 xapfd
3 afd xdp axf
7 adgbnem
4 aem egn dgm emb
0
```

Sample Output

```
1 fp*
0 axfdp
2 ade bmn
```

Problem C: Chain shopping

Input: C.in

Chain shopping is a new scheme introduced at a Mall to promote sale. Mall offers normal discount on every item. Under the scheme a customer makes a list of distinct items to be purchased. The list is called a purchase chain. For the k^{th} item in the chain Mall offers k times the normal discount. This policy attracts customers to form long judicious chains. Assume that an item under sale is identified by an integer ID and normal discount D offered for an item is an amount in whole number.

Chain shopping is permitted under certain conditions. Any one of the available items may be included as the first item in the purchase chain. However for any other item in the purchase chain option is dependent on the previous selection. For each item ID Mall displays prominently a list P(ID) of next potential items. If ID is in the chain then the next item could only be any one of the items listed in P(ID). For example if Mall has four items with ID 24 29 61 81 and next potential items P(24), P(29), P(61) and P(81) are respectively 24: 29 61 29: 24 61 81 61: 24 81 29 81: 24 61 then the chain of three items 61 81 24 is valid while the chain 61 24 81 is invalid.

Write a program to print purchase chain(s) of specified length L, for which the total discount MD is maximum, given ID and D for each item under sale and all lists of next potential items.

Input

Input contains multiple test cases. Each case contains three lines.

The 1^{st} line gives an integer specifying L.

The 2^{nd} line gives a certain number (assume 20 or less) of pairs of integers in an arbitrary order. Each pair represents ID and D in order, for an item under sale.

The 3^{rd} line gives all lists of potential items in the form of a string of integers, space and colon (:). An integer n followed by colon identifies the beginning of a list. Integers separated by space that follow colon, identify items in the list. Either the end of the line or the beginning of another list indicates the end of a list.

Input terminates with an input 0 as the first input for a test case.

Output

For each test case the 1^{st} line contains two integers N and MD. Integer N is the total number of purchase chains for each one of which the total discount is MD. Each of the following N lines contains one chain and chains appear in lexicographic order.

Sample Input

```
3
24 30 81 20 29 40 61 50
24: 81 29 81: 24 29 61 29: 81
61: 81 24
4
24 20 29 15 61 10 81 5
24: 29 61 29: 24 61 81 61: 24
81 29 81: 24 61
0
```

Sample Output

```
2 230
29 81 61
61 24 29
1 150
81 61 29 24
```

Problem D: Development
Input File : D.in

Development of unused land is an essential first step for creation of infrastructure. An entrepreneur ventures into a project for creation of a state-of-the-art health-care centre in a city. Government is prepared to allocate under certain conditions, a part of an unused block of land for the project. The block exists in the form of an MxN grid of square plots each of unit area. The entrepreneur has estimated for each plot, the cost of development in certain unit and rounded it to an integer. He requires for the project, a total number of K connected plots. However he is allowed to select only a rectangular/ square block of plots so that at least one side of the selected block either coincides with or is a part of a side of the existing block. In addition, on removal of the selected block, plots in the existing block should remain connected.

Given the cost of development for each plot in the grid and the total number K of plots to be selected, write a program to select all feasible blocks (say b in number), for each one of which the total cost of development C is the least.

Tables below illustrate the selection of all 3 feasible blocks (b=3), each containing 4 plots (K=4) with least total cost of development (C=47). Integers on the grid represent cost of development of plots and shaded plots identify selected blocks.

03	20	29	06
21	09	06	11
07	10	25	05

03	20	29	06
21	09	06	11
07	10	25	05

03	20	29	06
21	09	06	11
07	10	25	05

Input

Input may contain multiple test cases.

For each test case the first line gives three integers M, N and K as defined above. Each of the next M lines contains N integers; the j^{th} integer in the i^{th} line represents the cost of development of the plot in i^{th} row and j^{th} column of the grid.

Input terminates with a line containing 0 as the first input for a test case.

Output

For each test case, print in the first line, the least total cost of development C and the total number of feasible blocks b.

The first line is followed by b more lines; each line contains two pairs of integers identifying a feasible block. The 1^{st} pair identifies the first and the last row of the selected block, while the 2^{nd} pair identifies the first and the last columns of the block. The lines appear in lexicographic order.

Sample Input

```
3 4 4
3 20 29 6
21 9 6 11
7 10 25 5
3 4 3
3 20 29 6
21 9 6 11
7 10 25 5
0
```

Sample Output

```
47 3
2 3 1 2
2 3 3 4
3 3 1 4
22 1
1 3 4 4
```

Problem E: Editing
Input File: E.in

Editing of original version of a document produces the final version after certain corrections, insertions, deletions and/or reorganization of text. The original as well as the final version of the document may be considered as a string of only case-sensitive letters, digits, space and standard punctuation symbols: comma, full stop, semicolon and colon. In order to avoid confusion the character # is used to represent a space character in a string. Often the two versions contain common sub-strings of characters intermixed with uncommon sub-strings scattered throughout the document.

A publishing house wants to have a computer program that will identify the difference between the two versions of a document, given the two versions as input. You are required to write a program for the publishing house.

The difference between the two versions is considered to be a single string containing reduced forms of original and edited versions, separated by the underscore (_) character. The reduced forms of two versions are obtained by deleting successively the longest common sub-strings of length two or more from the two versions simultaneously until no more common sub-strings exist in the two versions. In case there exist two or more longest common sub-strings, the rightmost longest sub-string in the original version is selected first for deletion. If the selected sub-string in the original version occurs more than once in the edited version then the right most sub-string is selected for deletion.

Input

Input may contain multiple test cases.

Each test case contains two lines. The first line contains the original version while the second line contains the edited version of the document. Assume that each version contains not more than 250 characters.

Input terminates with a line containing # as the first input for a test case.

Output

For each test case, print output in one line. The line contains two fields separated by a space. The first field is an integer representing the total number of common sub-strings deleted; the second field is the string representing the difference between the two versions.

Sample Input

```
docu#giori#nal#,#ment.
Original#,#document#.
ready#and#explore
explore#and#ready
gty#frsirheir:sig
eir:sigtyfr#ssirh
#
```

Sample Output

```
5 o#._O.
3 _
4 g_s
```

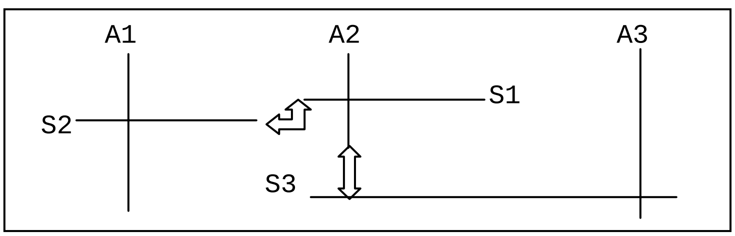
Problem F: Full Connectivity
Input File: F.in

Full connectivity of network of roads in a new developing industrial city does not exist. Due to existing partial connectivity it is not always possible to reach a location on a road from another location on another road. It is considered desirable to connect all unconnected existing network of roads with minimum cost. You are required to write a program to determine the total minimum cost required for connecting all unconnected roads.

At the time of development of each sector, axes parallel roads are constructed in the sector, parallel to either of the two mutually perpendicular axes passing through the city center. The x-axis extends from west to east while the y-axis extends from south to north. The roads extending from west to east are called Streets while the roads extending from south to north are called avenues. A pair of points with the same y-coordinate identifies the extremities of each street. Likewise a pair of points with the same x-coordinate identifies the extremities of each avenue. Assume for simplicity that integer coordinates represent each extremity and the cost of construction of roads is equal, in certain unit, to the length of the road constructed.

For illustration consider the connectivity of the network of roads shown in the figure below, with three streets S1, S2, S3 and three avenues A1, A2, A3. The network is not fully connected. However, connection with minimum cost can be established by construction of additional roads at locations indicated by block arrows:

1. Extend either S1 or S2 and join the two streets constructing a new avenue.
2. Extend the avenue A2 and connect it to S3.



Input

Input may contain multiple test cases. Each case contains two lines.

The first line identifies all streets in an arbitrary order while the second line identifies all avenues, again in an arbitrary order. Each street is represented by three integers: the x-coordinate x0 of the extremity towards west, the x-coordinate x1 of the extremity towards east and the common y-coordinate yc. Likewise, each avenue is represented by three integers: the common x-coordinate xc, the y-coordinate y0 of the extremity towards south and the y-coordinate y1 of the extremity towards north.

Input terminates with a line containing 0 as the first input for a test case.

Output

For each test case, print the total minimum cost required for connecting all unconnected roads.

Sample Input

```
6 20 0 1 15 10 -9 -2 8
-7 2 10 8 3 11 19 -4 2
-6 20 10 -16 12 -6 -12 12 4
-8 -4 8 8 -12 8
0
```

Sample Output

```
8
2
```

Problem G: Game U-turn
Input File: G.in

Game U-turn is a card game of patience, played by a single player. Given a sequence of cards with some cards face up and others face down at arbitrary positions, a player is required to perform a series of operation called U-turn to make all cards face up without altering positions of cards in the sequence. A U-turn operation is performed on a sub-string of cards of any length to u-turn the face of each card in the sub-string, i.e., if the face of a card is up then put it down and if it is down then put it up, without altering positions of cards in the sub-string. The effort of a U-turn operation is the length of the sub-string on which the operation is performed. The series of U-turns should be such that the effort of each U-turn operation is distinct and the total effort of the series of operations is minimum.

Write a program to find the total minimum effort required to make all cards of a given sequence face up using a series of U-turn operation. Assume that the sequence of cards is represented by a string of 0's and 1's where a zero represents a card with face up and a one represents a card with face down.

Input

Input may contain multiple test cases.

Each test case has a single input line containing a string of 0's and 1's. The length of the string is fifteen or less.

Input terminates with a line containing 0 as the input for a test case.

Output

For each test case, print the total minimum effort required.

Sample Input	Sample Output
1000101	9
001100010	3
1010100000	7
0	

Problem H: Homework

Input File: H.in

Homework to primary school children often becomes primary home engagement for parents. In order to lessen the burden of parents a software firm proposes to develop a package that includes solutions to problems on Arithmetic, which are commonly given as homework. As a programmer of the software firm you are required to write a program to solve such a problem.

The problem is to add triplets of integers from a given set S of n ($\leq$30) distinct positive integers and determine all K subsets {a b c d e f} of six distinct elements in S so that the sum of three elements in the subset, say (a+b+c), is equal to the sum (d+e+f) of the other three.

For example, given the set S={1001 2001 3001 4001 5001 6001 7001} there exists exactly three subsets (K=3): {1001 2001 3001 4001 5001 7001}, {1001 2001 3001 5001 6001 7001} and {1001 3001 4001 5001 6001 7001} each one of which satisfies the stated condition.

Input

Input may contain multiple test cases. Each test case contains two lines.

The first line gives two integers: n, the total number of elements in S and I, an indicator that indicates the output format. The indicator I, is either 0 or 1.

The second line gives n elements of S in an arbitrary order.

Input terminates with an input 0 as the first input for a test case.

Output

For each test case, print output in either of two formats.

If the indicator I is 0 then print only the total number K of subsets in one line.

Otherwise the line is followed by K more lines each containing elements of a subset. The elements of each subset appear in ascending order and the subsets appear in lexicographic order.

Sample Input

```
6 0
50 22 87 180 65 115
7 1
1001 2001 3001 4001 5001 6001
7001
8 0
12 20 35 38 46 23 18 58
0
```

Sample Output

```
0
3
1001 2001 3001 4001 5001 7001
1001 2001 3001 5001 6001 7001
1001 3001 4001 5001 6001 7001
4
```