Arithmetic

by Akaki Mamageishvili

Manao loves integer numbers very much. For the last few months he spent all his time inventing and solving different, sometimes senseless, equations. He is not that good in mathematics to solve all the equations he invented, though. He therefore has a long list of unsolved equations. Fortunately, he heard that students who participate in ACM competitions are perfect in mathematics, and decided to ask help from you.

Your task is to help Manao find a positive integer x such that $a^x + b^x$ is divisible by p^k; a, b, k, and x are positive integers; p is an odd prime; $1 \leq a, b, p \leq 10'000$; $0 < k < 38$; and $x, p^k \leq 10^{18}$.

Input

The input contains several testcases. Each testcase contains a single line with 4 integers a, b, p, k. The input ends with a line containing four zeros, which must not be processed.

Output

For each testcase your program should output a single line with an integer x, the solution to Manao's equation. If there is no such x, you should output -1 instead.

Sample input

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

Sample output

```
2
-1
```

Explanation: $3^2 + 4^2 = 25$ is divisible by $5^2 = 25$; $1^x + 1^x$ is always equal to 2, so there is no solution.

Computing using big integers

In C/C++, you may use the GNU Multiple Precision Arithmetic Library for all problems. Documentation is available at `http://doc.hc2.ch`. There is also a sample code on how to use this library on page 9 of the team manual, which is also available there.
In Java, we recommend the use of `java.math.BigInteger`. Documentation is available at `http://doc.hc2.ch`.

Cantonal Courier

by Jan Hązła

To earn money for the journey of your dreams (Valencia and Saint Petersburg) you come up with an idea for a shipping company. You are going to have an employee (a courier) in each canton of Algoland. The caveat that is going to let you offer competitive rates while promoting sustainability is that your couriers are going to use public transport. The distinctive feature of Algoland's public transport system is that each canton consists of several zones – depending on your route you need to buy the tickets for some subset of those zones.

For each canton you are given a list of possible assignments and the reward for each of them that a customer will pay if you agree to take it. For each assignment you are also given the list of zones for which you need a ticket in case you want to do it. Finally, you are given the price of the ticket for each zone. All the tickets are day passes and your couriers are truly excellent, so you can be sure that once bought, a single ticket can be reused for several jobs.

Find the optimal profit (payment from jobs minus costs of tickets) you can achieve for each canton.

Input

The input file consists of several testcases. Each testcase describes a single canton and starts with a line holding two integers: $1 \leq Z, J \leq 100$, where Z is a number of zones and J is a number of jobs.
In the next line there are Z integers $c_1, \ldots, c_Z, 1 \leq c_i \leq 5000$, where c_i is the cost of the ticket for zone i. The third line contains J integers $p_1, \ldots, p_J, 1 \leq p_i \leq 5000$, where p_i is the reward for job i.
J lines follow – i-th of those lines describes the tickets needed for job i. Each of those lines starts with $0 \leq N_i \leq Z$, followed by a strictly increasing sequence of N_i zones (1-based) for which the tickets are needed.
All consecutive numbers in a single line are single-space separated. The input terminates with a line containing two zeros.

Output

For each testcase output a single line with an integer: your profit given optimum choice of jobs.

Sample input

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

Sample output

```
1
3
```

Merlin's Orb

by Andrei Giurgiu

Old Merlin owns a powerful crystal ball, which allows him to chat with his fellow wizards wirelessly. Unfortunately for our friend, not so long ago the crystal orb started to exhibit a very strange behaviour. Instead of showing a smooth crystal-clear image as usual, the ball now displays some annoying greenish circles on its surface. To make matters worse, at any intersection point of two circles a potentially destructive high-energy beam of light spews out of the ball. Understandably, Merlin is terribly worried about these leaks, as too many of them would cause his prized orb to disintegrate in an Earth-shattering explosion (an event highly undesirable).

There is a glimmer of hope, however: after carefully studying the problem, he is now able to pre-determine the size of the circles, but not their positions - so the circles will now have a known size each, but they will appear uniformly and independently at random on the surface of the crystal sphere.

Merlin now wants to devote all his time repairing his device and needs some help from you. He would like to have an idea about how much more damage the orb will sustain, so he wants you to evaluate the expected number of energy releases that will occur during the time he is working on the orb. It's also good to know that in the extremely unlikely event that two circles overlap completely, nothing happens (Merlin tried this).

Merlin is a nice wizard and wants to save you some time, so he provides you with one of his many theorems: Suppose you have a sphere, and that the equator of this sphere has latitude 0, and latitude increases towards the north pole, which has latitude 90°. Then the area of the surface that lies north of the circle of latitude ϕ is $2\pi R^2(1-\sin\phi)$. He also gives you an old book of trigonometry: after you open it on a random page, you see

$$\cos(\alpha-\beta)-\cos(\alpha+\beta) = 2\sin\alpha\sin\beta.$$

Input

Each test case takes up two lines of input:

- The first line contains two numbers: the number $2 \le N \le 3 \cdot 10^5$ of circles that will appear, and the radius of the crystal ball, given as an integer R, where $0 < R \le 10^8$.
- The second line contains N integers $0 < r_i \le R$, $1 \le i \le N$, representing the radii of the circles.

The succession of test cases ends with a line containing two zeros.

Output

For each test case, output one line containing just one real number, representing the expected number of points of circle intersections.

Your result is guaranteed to be accepted if either the relative error or the absolute error is less than 10^{-7}. Please make sure to supply enough decimals so that you do not run into trouble! Also, do not process the line containing the two final zeros (it is not a valid input, anyway).

Sample input

```
2 5
5 5
```

```
3 7
1 2 4
0 0
```

Sample output

```
2.000000000
0.571428571
```

New Year

by Rajko Nenadov

New Year is just around the corner, and wizards at the Unseen University of Zurich want their hallways to be as colorful as possible for this occasion. They have spells for K different colors (numbered from 1 to K) and there is a total of N chambers in the University, $\lceil \frac{N}{2} \rceil$ in the west and $\lfloor \frac{N}{2} \rfloor$ in the east wing. Each hallway connects a chamber in the west wing with a chamber in the east wing and no two hallways connect the same pair of chambers.

Since colors are magical (as everything else in that place), they tend to become unstable and possibly explode if two adjacent hallways have the same color. As you are an expert in colors and hallways at the Adventure consulting company, wizards have hired you to find out the largest number of hallways they can color such that no two colors are too close to each other, that is, no two hallways which leave the same chamber have the same color. Since this number alone is not very helpful for them, they also asked you to provide them with one such coloring.

All hallways that are not colored will be completely dark and depressing, and wizards want their University to be as little depressing as possible for the New Year! If you fail to find the answer, or you provide them with a wrong one, they will turn you into a frog!

Input

Each test case starts with a line holding three integers: N ($1 \leq N \leq 200$) – the number of chambers; M ($2 \leq M \leq 10000$) – the number of hallways; and K ($1 \leq K \leq 20$) – the number of different colors. Chambers are numbered from 1 to N. In each of the next M lines there are two integers a ($1 \leq a \leq \lfloor \frac{N}{2} \rfloor$) and b ($\lfloor \frac{N}{2} \rfloor + 1 \leq b \leq N$), specifying there is a hallway running from chamber a to chamber b.

The input terminates with a line containing three zeros. This line must not be processed.

Output

For each test case, first output a number L, the largest number of hallways that can be colored such that the condition from the statement holds, and in the next L lines output numbers h and c – meaning that the h-th edge from that test case should be colored with a color c. Any valid coloring will be accepted!

Sample input

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

Sample output

```
7
1 1
2 1
3 1
4 1
5 2
6 2
7 2
```

Packing T-Shirts

by Robert R. Enderlein

You are currently packing your luggage for your trip to Valencia, and like any self-respecting programmer you must face the difficult task of choosing which T-shirts to take with you. After having packed all other bare essentials (laptop, mouse, keyboard, socks, etc.) and having reserved some space for all the paraphernalia you will carry back from SWERC (medal, diploma, your SWERC T-shirt, prizes), you are left with a capacity of c in your luggage.

While the weights of your T-shirts look random to your teammates, they in fact obey an arcane rule: you have chosen a superincreasing sequence $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_N)$ (superincreasing means that each element is larger than the sum of all the previous elements $\alpha_i > \sum_{j=1}^{i-1} \alpha_j$); a prime number q larger than the sum of all α_i; and an integer r that is coprime to q. The weight of your ith T-shirt in your collection is equal to $r \cdot \alpha_i \pmod q$.

You would like to choose a set of T-shirts to take with you, such that their combined weight is exactly equal to c. This will impress your teammates, who will think that you can solve NP-complete problems.

Input

Each test-case starts with a line holding three integers: N ($1 \leq N \leq 61$); q ($2 \leq q \leq 2^{62} - 57$; q is prime); and r ($2 \leq r < q$). On the next line there are N integers $w_1, \ldots, w_N$ (such that $w_i = r \cdot \alpha_i \pmod q$ for some α_i; the sequence $(\alpha_1, \ldots, \alpha_N)$ is superincreasing; and $\sum_{i=1}^{N} \alpha_i < q$), the weights of your T-shirts. On the next line there is an integer c ($1 \leq c \leq 2^{62} - 1$), the remaining capacity of your luggage.

The input terminates with a line containing three zeros. This line must not be processed.

Output

For each test case, output a space-separated list of integers $(S_1, S_2, \ldots)$ on a single line, such that $(S_1, S_2, \ldots)$ is a subset of $(w_1, \ldots, w_N)$, the sum of that list equals the capacity $\sum_{i=1}^{|S|} S_i = c$, and such that the list is sorted, i.e., $S_i < S_{i+1}$. We guarantee you that there is always a solution.

Sample input

```
8 881 588
295 592 301 14 28 353 120 236
1129
0 0 0
```

Note: in this testcase, we have that $\alpha = (2, 7, 11, 21, 42, 89, 180, 354)$.

Sample output

```
236 301 592
```

Fancy Runway

by Christian Kauth

You are approaching Valencia airport, as you glance out of the window and spot the curious illumination of the runway: a line of N red or yellow lamps indicates that you are probably in the right place. Each second, the colours of the lamps change. Lamp i takes the colour that lamp $i+1$ had just before, giving the impression of a sliding ground! Although it looks random, the colour of the N^{th} bulb is determined in a deterministic way and depends on a subset S of lamps, which always includes lamp 1. If an odd number of those subset lamps currently shine red, then the N^{th} lamp will be red in the next state and yellow otherwise. Your coach, who knows lots of mazy stories, tells you that Spanish pilots read this colour code as instructions for landing approaches! Given the current state and the state at which your plane shall touch down, can you compute how long you will still be hovering over Valencia?

Input

Each test-case consists of three lines: The first line gives N ($2 \leq N \leq 32$), the number of lamps, followed by S, the number of lamps in the subset ($2 \leq S \leq N$) steering the N^{th} lamp. Line two holds S integers, indicating which lamps are part of that subset. Line three provides two N-character-long strings with characters r (for red) and y (for yellow), describing the current and final states. The first character of the string describes lamp 1, the last one lamp N. The input terminates with a line containing two zeros. This line must not be processed.

Output

For each test case, output a single integer, the minimum amount of time (in seconds) it will take your plane to touch down if this is possible given the current and final states. Otherwise, print a line containing the single character '*'.

Sample input

```
2 2
1 2
rr ry
4 2
1 2
ryry rryy
7 2
1 3
yryrryy yrrrryy
0 0
```

Sample output

```
1
6
*
```

Scouting Camp

by Jonas Wagner

Jonas and his friends are organizing a scouting camp. A friendly farmer gave them permission to construct their tents on his meadow. This they did. To protect the meadow during the camp, they now want to lay out a road network that connects all the tents using Passareco elements.

Figure 1: In a scouting camp, there are many tents, and roads consisting of Passareco elements connect them.

These Passareco elements are wooden paths that can be laid down in either east-west or north-south direction on the camping ground. For example, to connect the tent at (4, 7) with the tent at (19, 3), one would need 15 meters of Passareco going east followed by 4 meters going north (or the other way around), for a total of 19 meters.

The problem is, it sometimes rains in scouting camps. Thus, Jonas' friends would like to construct the Passareco network in a way that minimizes the length of the longest Passareco segment. This means they can move from any tent in the camping ground to another, taking shelter at tents on their way, without staying in the rain for too long.

Input

Each testcase consists of $N+1$ lines:

The first line contains N, the number of tents ($1 \leq N \leq 20000$). The following N lines each contain two integers x, y which denote the position of the tent. You may assume that $|x|, |y| \leq 2 \cdot 10^7$. No two tents are placed in the same point.

The input file ends with a testcase with $N = 0$, which should not be processed.

Output

For every testcase, print a single number on a line by itself: the largest distance between two tents connected by an uninterrupted Passareco way.

Sample input

```
5
1 3
4 1
5 3
8 4
4 6
0
```

Sample output

```
4
```

Notes

The sample input corresponds to the following layout:

```
      2
      |
      .
      |
1-.-.-.-3
        |
        .-.-.-4
        |
        .
        |
      5-.
```

The longest way out in the rain is 4 meters long (e.g. when going from tent 1 to tent 3). Note that to go from 4 to 5, one can make a detour to 3, in order to minimize the time in the rain.

The Sliding Puzzle

by Sandro Feuz

You may be familiar with the so called sliding puzzle. If not, don't worry, here is a short description:

- There are sliding puzzles of different sizes. The puzzle of size N is played on an $N \times N$ square and it will hold that $N \in \mathbb{N}, N \geq 2$.
- The fields are filled with the numbers 1 through $N^2 - 1$ in some given starting configuration. Each number appears exactly once, hence there is exactly one free field.
- A valid move consists of moving one of the neighbors of the free field to the free field itself. That is, in each configuration you have between 2 and 4 valid moves and a move can be specified by the number, which will end up in the free field.[1]

In this task you will be given a starting configuration for a sliding puzzle of size N, as well as M moves. You have to simulate all the moves and output the resulting configuration. Invalid moves should be ignored.

Input

The input consists of multiple testcases.
Each testcase starts with a line of two integers N ($2 \leq N \leq 100$), the size of the sliding puzzle, and M ($0 \leq M \leq 100\,000$), the number of moves to simulate.
The next line contains a permutation of the integers 0 through $N^2 - 1$, separated by spaces. It describes the starting configuration. The number on field (x, y), $1 \leq x, y \leq N$, is given as the $((y-1) \cdot N + x)$-th element of the permutation. The number 0 represents the free field. The top-left corner has coordinates $(1, 1)$.
Each of the following M lines contains an integer X ($1 \leq X \leq N^2 - 1$) describing a move as explained above. Remember that invalid moves should be ignored.

The input terminates with a line containing two zeros. This line must not be processed.

Output

For each test case, output a single line with the sliding puzzle configuration after simulating the M moves, in the same format we used in the input. That is, you should output a single line containing a permutation of the numbers 0 through $N^2 - 1$.

Sample input

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

[1] A valid move can alternatively be seen as moving the free field to one of its neighbors.

```
4
3
1
3
7
1
0 0
```

Sample output

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

Explanation of the sample

The starting configuration, as well as the configurations after every move are shown in figures (a) through (g). Note that the second move was invalid, therefore the configuration did not change.

7	4	
3	1	6
8	5	2

(a) Start configuration

7		4
3	1	6
8	5	2

(b) After first move

7		4
3	1	6
8	5	2

(c) After second move

7	1	4
3		6
8	5	2

(d) After third move

7	1	4
	3	6
8	5	2

(e) After fourth move

	1	4
7	3	6
8	5	2

(f) After fifth move

1		4
7	3	6
8	5	2

(g) After sixth move