Problem A. Calendar

Input file:	calendar.in
Output file:	calendar.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Recently (or in a close future) Vasya decided to learn playing darts. Unfortunately, he had no target at home, so he decided to use an old wall calendar. Vasya opened the month of his birthday on the calendar, placed it on a wall and started throwing darts to it. Certainly, each throw pierced all the twelve pages of the calendar.

Each time Vasya tried to hit his birthday and at last he succeeded. Self-satisfied, he took the calendar off the wall and found out that the successful throw hit also his mother's birthday.

Given Vasya's birth date and his mother's month of birth, find out her birth date.

The deck calendar consists of twelve pages, describing months of the year. Each page is represented with a table of five rows and seven columns. Days of the month are written in the cells of the table in the following way: the first column contains all Mondays of the month, the second column—all Tuesdays, and so on. The first day of each month is always placed in the first row, all other days are placed sequentially. In a case when there are not enough cells for some last days of the month (they might be placed in the sixth row), they will be placed in the corresponding cells of the fifth row.

Input

Input file consists of several test cases. Each test case if a single string with four numbers Y, dV, mV and mM. The first number Y ($2000 \le Y \le 2099$) is the year of Vasya's calendar. Numbers dV and mV determine day and month of Vasya's birthday. It is a correct date of year Y. The last number in a test case mM ($1 \le mM \le 12$) determines the month of Vasya's mother's birthday.

The number of test cases in a file does not exceed 10 000. The file is terminated with a string of four zeroes.

Output

For each test case write all the days of month that might be Vasya's mother's birthday in increasing order, or 0 if Vasya made a mistake. Adhere to the sample output below as close as possible.

Example

calendar.in	calendar.out
2009 22 11 1 2009 26 10 11 2009 30 08 01 0 0 0 0	Case #1: 25 Case #2: 23 30 Case #3: 0

November 2009, second sample test case:

						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23/30	24	25	26	27	28	29

Problem B. Downhill

Input file:	`downhill.in`
Output file:	`downhill.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Vasya could delete single character from string w if obtained string is lexicographic smaller than w. For example, from string "`abacaba`" Vasya could obtain strings: "`aacaba`", "`abaaba`", "`abacaa`" and "`abacab`". Suppose that Vasya could make this delete operation several times. Could he obtain given string t from given string s?

Input

Each line of the input will contain two not empty strings—s and t, separated by single space. Each string contains only alphanumerical characters. Length of each string in input is not greater than 100 000. There are at most 1 000 000 alphanumerical characters in input.

Symbols are compared using their ASCII codes.

Output

Adhere to the sample output format below as close as possible.

Example

downhill.in	downhill.out
aaa a AB 0 ddbca b abacaba aaaa ababc aac	Case #1: Yes, it is possible. Case #2: No, it is impossible. Case #3: Yes, it is possible. Case #4: Yes, it is possible. Case #5: No, it is impossible.

Problem C. Isabel

Input file:	`isabel.in`
Output file:	`isabel.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

RING (Research Institute of iNtellectual Games) announced Isabel solitaire solving contest. Anyone who could solve given solitaire, has a chance to win a prize! Vasya wants to participate in contest, but he is very lazy, so he asks you to write program for solving Isabel solitaire.

Isabel solitaire rules: There are 52 playing cards in a pack, divided into suits, and, within suits, into ranks. The suits are Clubs, Diamonds, Hearts and Spades, abbreviated C, D, H and S. The 13 ranks (or face values) are : 2, 3, 4, 5, 6, 7, 8, 9, 10 (T), Jack (J), Queen (Q), King (K) and Ace(A). Tableau consists of thirteen piles. At the beginning, deal three cards face down and one card face up on every pile. Remove cards in pairs of equal rank. Every time a card is removed, the card underneath is flipped face up and becomes playable.

Your task is to write program which removes as much cards as possible.

Input

The input file contains not more than 5 test cases. Each case consist of thirteen lines, each line describes one pile from bottom to topmost card.

Output

For every case, your program should output as much moves as possible. In case of multiple solutions anyone will do. Adhere to the sample output format below as close as possible.

Examples

isabel.in
AC AH 6S AT 2C 2H 2S KT TC 3C 3H 3S 4C 4H 4S 4T 5C 5H 5S 5T 6C 7C 6T 7S 6H 8H QS 7T 8C AS 8S 8T 9C 9H 9S 9T TH TS QT TT JC JH JS JT QC QH 7H 3T KC KH KS 2T
isabel.out

Case #1: Vasya could remove 8 card(s).
  At step 1, Vasya should remove 3S and 3T.
  At step 2, Vasya should remove 7T and 7H.
  At step 3, Vasya should remove QS and QH.
  At step 4, Vasya should remove 8H and 8T.

Problem D. Knights and Liars

Input file:	knights.in
Output file:	knights.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Vasya finished research project for RIGS (Research Institute of Given Strings) and now he wants to have a vacation on the Knights and Liars island. Inhabitants of this island can be either knights or liars. All knights always tell the truth, and liars always tell the lie.

Vasya was told a strange story about this island. Once upon a time n inhabitants of this island came to the birthday party. They set down to round table, and each of them pointed at two persons and said: "If I ask both of you, whether I am knight or liar, your answers will be the same!".

Vasya understood that situation could happen if all persons were knights. But now he want to know whether there could be any liar or not.

Input

The input file contains not more than 1 000 test cases. The first line of each test case contains one integer n ($1 \le n \le 1\,000$)—the number of inhabitants came to the party. Let us number all of them from 1 to n. Then n lines follow. On the i-th line there are two integers a_i and b_i—the numbers of persons who was pointed by i-th person. ($1 \le a_i, b_i \le n$). Notice that all numbers i, a_i and b_i can be equal.

The sum of all n in the input does not exceed 1 000.

Output

For each test case write "Yes" if there can be a liar and "No" if all persons must be knights. Adhere to the sample output format below as close as possible.

Examples

knights.in	knights.out
3 2 3 1 3 1 2 2 1 2 1 2	Case #1: Yes Case #2: No

Problem E. k-th Degree Roots

Input file:	`kroots.in`
Output file:	`kroots.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Vasya works for the RIVER (Research Institute of Variables, Exponents and Roots). He is developing an efficient algorithm of finding k-th degree roots and soon he will need some test data.

Vasya's colleague Petya has already prepared an array a of integers for testing. However, it turned out that the numbers were quite random, and Vasya's algorithm was not designed to work with numbers that didn't have integer roots of k-th degree.

Vasya decided to create an array of numbers that fit his algorithm. He wants to use all the pairs (a_i, a_j) such that $i < j$ and $a_i \cdot a_j = x^k$ for some integer x.

Help Vasya to calculate the number of such pairs.

Input

The input file contains not more than 10 000 test cases. The first line of each test case contains integers n and k ($1 \le n, k \le 100\,000$). The second line contains n integers a_i ($1 \le a_i \le 1\,000\,000$).

The sum of all n in the input does not exceed 100 000.

Output

For each test case write the only integer—the answer to the problem. Adhere to the sample output format below as close as possible.

Example

kroots.in	kroots.out
4 2 2 8 3 2 3 5 17 239 1 2 2 17 17	Case #1: There are 3 pairs. Case #2: There are 0 pairs. Case #3: There are 1 pairs.

Problem F. Ali-Baba the Martian

Input file:	martian.in
Output file:	martian.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Ali-Baba the Martian is fond of searching for treasures. Once upon a time he found a treasure map. After many sufferings he found a secret door. There was a circle with N holes near it. On the other side of his map was written that there is martian herring in each hole, and it can be either head or tail up. Ali-Baba has M hands. He can put them into holes and rotate herrings in them as he wants (Ali-Baba can determine position of herring by touch). The trouble is that after each his action the circle starts to rotate rapidly, so after the circle stops Ali-Baba can not determine where he has put hands just now.

Ali-Baba knows that one of holes is the key of the door. That means the door will open if herring in this hole would be in some definite position. Unfortunately Ali-Baba doesn't know position of this key and the position of herring in it. Also he knows that if all other herrings except the key one will be in that position, the door would open anyway.

You task is to determine, whether exists strategy for Ali-Baba to open the door, or not.

Input

Input file consists of several data sets. Each data set consists of one line. This line contains two integers N and $M (1 \le N, M \le 15)$ — the number of holes in the circle and the number of Ali-Baba's hands. Input file ends with line contains two zeroes.

Number of data sets in the input file does not exceed 100.

Output

For each data set you should print either "`Yes`" if Ali-Baba can open the door, or "`No`". Adhere to the sample output format below as close as possible.

Examples

martian.in	martian.out
3 1 3 2 0 0	Case #1: No Case #2: Yes

Problem G. Plane

Input file:	plane.in
Output file:	plane.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

It's snowing and raining in the city of Nsk, but the plane has to depart in time. Snow sticks to wings of the plane, changing its balance. So the crew should do something to make the plane as balanced as possible before the take-off.

The airport has a complicated system of runways. When the plane drives along each of them, some snow can stick either to the left wing, or to the right wing, or to none of them.

Vasya is a plane crew assistant. His task is to choose the take-off way, minimizing the difference between the amounts of snow stuck to the wings.

Input

The input file contains not more than 10 test cases. Each case starts with a line containing two integer numbers n and k—the number of junctions and runways, respectively ($2 \le n \le 10\,000$, $1 \le k \le 10\,000$). k lines follow, each describing one runway. A description of a runway consists of three integers: a, b and w ($1 \le a \ne b \le n$, $-1 \le w \le 1$). It means a runway connecting junctions a and b, where snow sticks to the left wing ($w = -1$), to the right wing ($w = 1$), or where the snow does not stick ($w = 0$).

Note, that due to ecology problems snow has some complicated structure (moreover, scientists suppose that it has mental abilities...) So the wing to be stuck by the snow does not depend on the direction of movement.

The plane starts at junction number 1 and the junction where it is possible to take off has the number n. You may assume that some way to take off exists in each test case.

Output

For each test case write the minimal possible difference. Adhere to the sample output format below as close as possible.

Example

plane.in
2 1 2 1 -1 5 5 1 2 1 2 3 0 2 4 0 5 3 -1 5 3 1
plane.out
Case #1: The plane needs to bear the difference of 1. Case #2: The plane needs to bear the difference of 0.

Problem H. Prefix function

Input file:	prefix.in
Output file:	prefix.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

RIP (Research Institute of Prefixes) studies the prefix function.

Let's describe what the prefix function is. Let's take string s. We will denote by p_i a prefix of s of length i, and by s_i a suffix of s of length i. Now, we define $\pi(s)$ as maximal integer less then i such that prefix $p_{\pi(i)}$ is equal to $s_{\pi(i)}$.

So, the prefix function of string s is a sequence $\pi(1)$, $\pi(2)$, ..., $\pi(n)$, where n is length of s. For example, prefix function of string *abacaba* is $0, 0, 1, 0, 1, 2, 3$.

Unfortunately, the computer of RIP was attacked by virus. So some data is lost. Vasya has only part of prefix function π. So, for some i Vasya knows $\pi(i)$, and for others he doesn't.

Vasya wants to restore the original string s. In case of multiple variants, Vasya wants to construct the lexicographically smallest one.

Suppose that Vasya uses Chinese alphabet with many characters. We will denote the first letter by number 1, second one by number 2, and so on. You can suppose that there are as many characters as you need.

Input

Each test case will start with number n — length of the string. The second string contains n integers — values of $\pi(p_i)$ ($-1 \le \pi(p_i) \le n$). If Vasya does not know $\pi(p_i)$, that corresponding number will be -1.

The last line of input file contains one integer 0.

The sum of all n will be not more than 2000.

Output

For each test case write the minimal string, or state that there is know such string. Adhere to the sample output format below as close as possible.

Examples

prefix.in	prefix.out
7 0 0 1 0 1 2 3 4 -1 -1 -1 3 2 1 0 0	Case #1: The minimal string is 1 2 1 3 1 2 1 Case #2: The minimal string is 1 1 1 1 Case #3: There is no such string

Problem I. Yet another stone game

Input file:	stonegame.in
Output file:	stonegame.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Vasya has friend Petya, that works in RIGS — Research Institute of Games with Stones. Yesterday, Petya was invited to play a new game, invented in RIGS. The game rules are quite simple. In the beginning of the game there are N stones on the table. The first player takes one stone from the table. Then on every move, if k stones were taken on the previous turn, than the player can take k or $k + 1$ stones. Player, that can not make a move loses. Vasya wants to win, of course. So he wants to know, who will win in the game — first player or second.

Input

Each line of the input will contain integer number N ($1 \le N \le 100000$). The last line will contain one integer — 0.

Output

For each test case write the only integer — the answer for the problem. Adhere to the sample output format below as close as possible.

Example

stonegame.in	stonegame.out
1 2 3 0	Case #1: First player wins. Case #2: Second player wins. Case #3: Second player wins.

Problem J. Three Circles

Input file:	three.in
Output file:	three.out
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Vasya works for the RICH (Research Institute of Circles Heuristics). He has just selected three points on a plane. Now he wonders, whether there are three circles with centres in that points such that each pair of them has the only point of intersection.

Write a program to find such a triple.

Input

The input file consists of several test cases. Each test case is described with a single line containing six integer numbers x_1, y_1, x_2, y_2, x_3, y_3, describing coordinates of given points. All three points are distinct. All coordinates do not exceed 10^6 by absolute value.

The input file is terminated with a line of six zeroes. The number of test cases in input file does not exceed 10 000.

Output

For each test case write three real numbers—radii of circles with centres in given points (in the same order as in the input), or three numbers -1 if the answer does not exist. The radii in your answer must be at least 0.1. It is guaranteed that in each test case having some answer there is an answer satisfying the requirement above. If there are several answers, write any of them.

Adhere to the sample output below as close as possible.

Example

three.in	three.out
0 0 2 0 0 2 0 0 0 0 0 0	Case #1: 0.5857864 1.4142136 1.4142136

Problem K. Restoration of Triangulation

Input file:	`triangulation.in`
Output file:	`triangulation.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Triangulation of a regular polygon is a decomposition of it into a set of triangles in such a way, that all triangles have vertices only at the vertices of this polygon. One can easily understand that triangulation of a regular polygon with N vertices can be represented by a set of $N - 3$ diagonals of this polygon, any two of which don't intersect inside the polygon. Let us number all vertices form 1 to N in clockwise order. Then diagonal of a polygon can be represented by numbers of vertices it connects.

Good girl Ann has drawn a triangulation of regular N-gon on a sheet of paper. Also she has written all diagonals that form this triangulation on another sheet. After that bad boy Andrew teared the first sheet into pieces and rubbed off maximal number from each notation of diagonal.

Your task is to help Ann to restore initial triangulation from remaining data.

Input

Input file consists of several data sets. Each data set consists of two lines. The first line contains the only integer $N (N \geq 4)$—the number of vertices of regular polygon that was triangulated by Ann. The second line contains $N - 3$ integers remaining on the second sheet. All input data are correct. Input file ends with line contains the only integer 0.

Number of integers in the input file does not exceed 100 000.

Output

For each data set you should print diagonals that form Ann's triangulation. Each diagonal mush be printed on the new line, and in that line the least number of vertex must be printed first. Diagonals must be sorted lexicographically. In case of many solutions print any of them. Adhere to the sample output format below as close as possible.

Examples

triangulation.in	triangulation.out
4 1 5 1 3	Case #1: 1 3 Case #2: 1 3 3 5