B. Guess That Car!

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A widely known among some people Belarusian sport programmer Yura possesses lots of information about cars. That is why he has been invited to participate in a game show called "Guess That Car!".

The game show takes place on a giant parking lot, which is $4n$ meters long from north to south and $4m$ meters wide from west to east. The lot has $n + 1$ dividing lines drawn from west to east and $m + 1$ dividing lines drawn from north to south, which divide the parking lot into $n·m$ 4 by 4 meter squares. There is a car parked strictly inside each square. The dividing lines are numbered from 0 to n from north to south and from 0 to m from west to east. Each square has coordinates (i, j) so that the square in the north-west corner has coordinates $(1, 1)$ and the square in the south-east corner has coordinates (n, m). See the picture in the notes for clarifications.

Before the game show the organizers offer Yura to occupy any of the $(n + 1)·(m + 1)$ intersection points of the dividing lines. After that he can start guessing the cars. After Yura chooses a point, he will be prohibited to move along the parking lot before the end of the game show. As Yura is a car expert, he will always guess all cars he is offered, it's just a matter of time. Yura knows that to guess each car he needs to spend time equal to the square of the euclidean distance between his point and the center of the square with this car, multiplied by some coefficient characterizing the machine's "rarity" (the rarer the car is, the harder it is to guess it). More formally, guessing a car with "rarity" c placed in a square whose center is at distance d from Yura takes $c·d^2$ seconds. The time Yura spends on turning his head can be neglected.

It just so happened that Yura knows the "rarity" of each car on the parking lot in advance. Help him choose his point so that the total time of guessing all cars is the smallest possible.

Input

The first line contains two integers n and m $(1 \le n, m \le 1000)$ — the sizes of the parking lot. Each of the next n lines contains m integers: the j-th number in the i-th line describes the "rarity" c_{ij} $(0 \le c_{ij} \le 100000)$ of the car that is located in the square with coordinates (i, j).

Output

In the first line print the minimum total time Yura needs to guess all offered cars. In the second line print two numbers l_i and l_j $(0 \le l_i \le n, 0 \le l_j \le m)$ — the numbers of dividing lines that form a junction that Yura should choose to stand on at the beginning of the game show. If there are multiple optimal starting points, print the point with smaller l_i. If there are still multiple such points, print the point with smaller l_j.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input

```
2 3
3 4 5
3 9 1
```

output

```
392
1 1
```

input

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

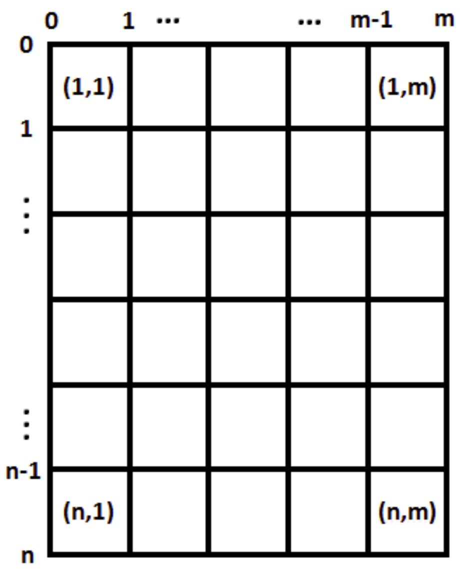
output

```
240
2 3
```

Note

In the first test case the total time of guessing all cars is equal to $3 \cdot 8 + 3 \cdot 8 + 4 \cdot 8 + 9 \cdot 8 + 5 \cdot 40 + 1 \cdot 40 = 392$.

The coordinate system of the field:



D. Ice Sculptures

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Berland University is preparing to celebrate the 256-th anniversary of its founding! A specially appointed Vice Rector for the celebration prepares to decorate the campus. In the center of the campus n ice sculptures were erected. The sculptures are arranged in a circle at equal distances from each other, so they form a regular n-gon. They are numbered in clockwise order with numbers from 1 to n.

The site of the University has already conducted a voting that estimated each sculpture's characteristic of t_i — the degree of the sculpture's attractiveness. The values of t_i can be positive, negative or zero.

When the university rector came to evaluate the work, he said that this might be not the perfect arrangement. He suggested to melt some of the sculptures so that:

- the remaining sculptures form a regular polygon (the number of vertices should be between 3 and n),
- the sum of the t_i values of the remaining sculptures is maximized.

Help the Vice Rector to analyze the criticism — find the maximum value of t_i sum which can be obtained in this way. It is allowed not to melt any sculptures at all. The sculptures can not be moved.

Input

The first input line contains an integer n $(3 \le n \le 20000)$ — the initial number of sculptures. The second line contains a sequence of integers $t_1, t_2, ..., t_n$, t_i — the degree of the i-th sculpture's attractiveness $(-1000 \le t_i \le 1000)$. The numbers on the line are separated by spaces.

Output

Print the required maximum sum of the sculptures' attractiveness.

Sample test(s)

input

```
8
1 2 -3 4 -5 5 2 3
```

output

```
14
```

input

```
6
1 -2 3 -4 5 -6
```

output

```
9
```

input

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

output

```
21
```

Note

In the first sample it is best to leave every second sculpture, that is, leave sculptures with attractivenesses: 2, 4, 5 и 3.

SPOJ Problem Set (classical)

12368. Emma s Domino

Problem code: TAP2012E

[The original version of this problem (in Spanish) can be found at http://www.dc.uba.ar/events/icpc/download/problems/taip2012-problems.pdf]

The *domino effect* is a phenomenon that occurs when in a line of domino pieces, each standing on its smallest face, the first piece from one of the line's ends falls in the direction of the next piece. In turn, this second piece falls over the third one in the line, and so on until the other end of the line is reached, at which point every piece has fallen. Note that in order to produce this effect, the distance between consecutive pieces in the line must be lower or equal to their height.

Emma has very recently found out about the domino effect, and she was immediately amazed by it. She spent all morning forming a line with the **N** domino pieces that her brother Ezequiel gave her, but just before she was going to make the first piece fall, her grandma came to her home and took her to play in the park. Ezequiel knows Emma has not taken into account the distance between consecutive pieces when she formed her domino line, and doesn't want to see her frustrated if all the pieces do not fall after she pushes the first one. Thus, Ezequiel wants to move some pieces from inside the line so that the distance between consecutive pieces is always lower or equal to their height **H**. Because he doesn't want Emma to find out that he has moved some of the pieces, he will leave the first and last pieces where they are, and he would also like to move as few pieces as possible from inside the line. What is the minimum number of pieces he must move?

Input

Each test case is described using two lines. The first line contains two integer numbers **N** and **H**, indicating respectively the number of pieces in the line (**$3 \le N \le 1000$**) and their height (**$1 \le H \le 50$**). The second line contains **N-1** integers **D_i**, representing the distances between pairs of consecutive domino pieces, in the order given by the line (**$1 \le D_i \le 100$** for **$i = 1, 2, ..., N-1$**). The end of the input is signalled by a line containing two times the number **-1**.

Output

For each test case, you should print a line containing a single integer number, representing the minimum number of pieces that must be moved in order to have the distance between consecutive pieces always lower or equal to **H**. Note that the first and last pieces cannot be moved, and that the relative order between the the pieces cannot be changed. If it is impossible to achieve the desired result, print the number **-1**.

Example

```
Input:
8 3
2 4 4 1 4 3 2
10 2
1 2 2 2 2 2 2 2 3
5 2
2 2 2 2
5 3
1 6 2 4
-1 -1

Output:
3
8
0
-1
```

A. k-String

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A string is called a k-string if it can be represented as k concatenated copies of some string. For example, the string "`aabaabaabaab`" is at the same time a 1-string, a 2-string and a 4-string, but it is not a 3-string, a 5-string, or a 6-string and so on. Obviously any string is a 1-string.

You are given a string s, consisting of lowercase English letters and a positive integer k. Your task is to reorder the letters in the string s in such a way that the resulting string is a k-string.

Input

The first input line contains integer k ($1 \le k \le 1000$). The second line contains s, all characters in s are lowercase English letters. The string length s satisfies the inequality $1 \le |s| \le 1000$, where $|s|$ is the length of string s.

Output

Rearrange the letters in string s in such a way that the result is a k-string. Print the result on a single output line. If there are multiple solutions, print any of them.

If the solution doesn't exist, print "`-1`" (without quotes).

Sample test(s)

input

```
2
aazz
```

output

```
azaz
```

input

```
3
abcabcabz
```

output

```
-1
```

D. Ring Road 2

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

It is well known that Berland has n cities, which form the Silver ring — cities i and $i + 1$ ($1 \leq i < n$) are connected by a road, as well as the cities n and 1. The goverment have decided to build m new roads. The list of the roads to build was prepared. Each road will connect two cities. Each road should be a curve which lies inside or outside the ring. New roads will have no common points with the ring (except the endpoints of the road).

Now the designers of the constructing plan wonder if it is possible to build the roads in such a way that no two roads intersect (note that the roads may intersect at their endpoints). If it is possible to do, which roads should be inside the ring, and which should be outside?

Input

The first line contains two integers n and m ($4 \leq n \leq 100, 1 \leq m \leq 100$). Each of the following m lines contains two integers a_i and b_i ($1 \leq a_i, b_i \leq n, a_i \neq b_i$). No two cities will be connected by more than one road in the list. The list will not contain the roads which exist in the Silver ring.

Output

If it is impossible to build the roads in such a way that no two roads intersect, output `Impossible`. Otherwise print m characters. i-th character should be `i`, if the road should be inside the ring, and `o` if the road should be outside the ring. If there are several solutions, output any of them.

Sample test(s)

input

```
4 2
1 3
2 4
```

output

```
io
```

input

```
6 3
1 3
3 5
5 1
```

output

```
ooo
```

ZOJ Problem Set - 2016

Play on Words

Time Limit: 5 Seconds   Memory Limit: 32768 KB

Some of the secret doors contain a very interesting word puzzle. The team of archaeologists has to solve it to open that doors. Because there is no other way to open the doors, the puzzle is very important for us.

There is a large number of magnetic plates on every door. Every plate has one word written on it. The plates must be arranged into a sequence in such a way that every word begins with the same letter as the previous word ends. For example, the word "acm" can be followed by the word "motorola". Your task is to write a computer program that will read the list of words and determine whether it is possible to arrange all of the plates in a sequence (according to the given rule) and consequently to open the door.

Input

The input consists of T test cases. The number of them (T) is given on the first line of the input. Each test case begins with a line containing a single integer number Nthat indicates the number of plates (1 <= N <= 100000). Then exactly Nlines follow, each containing a single word. Each word contains at least two and at most 1000 lowercase characters, that means only letters 'a' through 'z' will appear in the word. The same word may appear several times in the list.

Output

Your program has to determine whether it is possible to arrange all the plates in a sequence such that the first letter of each word is equal to the last letter of the previous word. All the plates from the list must be used, each exactly once. The words mentioned several times must be used that number of times.

If there exists such an ordering of plates, your program should print the sentence "Ordering is possible.". Otherwise, output the sentence "The door cannot be opened.".

Sample Input

```
3
2
acm
ibm
3
acm
malform
mouse
2
ok
ok
```

Sample Output

```
The door cannot be opened.
Ordering is possible.
The door cannot be opened.
```

Source: **Central Europe 1999**

Submit   Status

SPOJ Problem Set (classical)

12365. Ball of Reconciliation

Problem code: TAP2012B

[The original version of this problem (in Spanish) can be found at http://www.dc.uba.ar/events/icpc/download/problems/taip2012-problems.pdf]

Every year the kingdoms of Cubiconia, Quadradonia and Nlogonia organize a ball to commemorate the end of the war that ravaged the region for many years. A certain number of noblemen of each kingdom is invited to participate in this event, and it is expected that every pair of guests coming from different kingdoms will dance together exactly once. That is, each of the guests from Cubiconia shall dance once with every guest from Quadradonia and Nlogonia; in turn, each of the guests from Quadradonia shall also dance once with every guest from Nlogonia. However, guests coming from the same kingdom are not allowed to dance with one another.

To help organize the ball, the total number of dances that will take place is determined beforehand, so that care must be taken when choosing the number of noblemen that shall be invited from each kingdom. For example, if it is decided that the ball must have **N = 20** dances, one possibility is to invite **6** noblemen from Cubiconia, **2** from Quadradonia and **1** from Nlogonia, which can be represented by the expression **(6, 2, 1)**. This is a valid option because the total number of dances would then be **6*2 + 6*1 + 2*1 = 20**.

Traditions whose origins nobody can now remember indicate that the number of invited noblemen from Cubiconia must be greater or equal to the number of those coming from Quadradonia, and at the same time the number of invited noblemen from Quadradonia must be greater or equal to those coming from Nlogonia. Thus, for **N = 20** dances there are exactly **5** possible ways to choose the number of guests from each kingdom, namely **(5, 4, 0)**, **(4, 2, 2)**, **(10, 2, 0)** and **(20, 1, 0)** as well as the aforementioned **(6, 2, 1)**.

With so many restrictions, the organizing committee has problems finding the ideal way to choose the number of guests from each kingdom. Your task is to help this committee by counting the different ways in which the number of guests can be chosen for a ball with **N** dances. Two of these ways are considered different if they differ in the number of invited noblemen from at least on of the three kingdoms.

Input

Each test case is described using a single line, containing an integer **N** representing the total number of dances that the ball must have (**$1 \le N \le 10^4$**). The end of the input is signalled by a line containing the number **-1**.

Output

For each test case, print a single line containing the number of different ways in which the number of guests from each kingdom can be chosen in order to have a ball where there are exactly **N** dances, with all the restrictions mentioned in the problem statement.

Example

```
Input:
20
1
9747
-1

Output:
5
1
57
```

show comments

16185. Mining your own business

Problem code: BUSINESS

John Digger is the owner of a large illudium phosdex mine. The mine is made up of a series of tunnels that meet at various large junctions. Unlike some owners, Digger actually cares about the welfare of his workers and has a concern about the layout of the mine. Specifically, he worries that there may a junction which, in case of collapse, will cut off workers in one section of the mine from other workers (illudium phosdex, as you know, is highly unstable). To counter this, he wants to install special escape shafts from the junctions to the surface. He could install one escape shaft at each junction, but Digger doesn't care about his workers that much. Instead, he wants to install the minimum number of escape shafts so that if any of the junctions collapses, all the workers who survive the junction collapse will have a path to the surface.

Write a program to calculate the minimum number of escape shafts and the total number of ways in which this minimum number of escape shafts can be installed.

Input

The input consists of several test cases. The first line of each case contains a positive integer N ($N \leq 5 \cdot 10^4$) indicating the number of mine tunnels. Following this are N lines each containing two distinct integers s and t, where s and t are junction numbers. Junctions are numbered consecutively starting at 1. Each pair of junctions is joined by at most a single tunnel. Each set of mine tunnels forms one connected unit (that is, you can get from any one junction to any other).

The last test case is followed by a line containing a single zero.

Output

For each test case, display its case number followed by the minimum number of escape shafts needed for the system of mine tunnels and the total number of ways these escape shafts can be installed. You may assume that the result fits in a signed 64-bit integer.

Follow the format of the sample output.

Example

Input:

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

Output:

```
Case 1: 2 4
Case 2: 4 1
```

B. Cthulhu

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

...Once upon a time a man came to the sea. The sea was stormy and dark. The man started to call for the little mermaid to appear but alas, he only woke up Cthulhu...

Whereas on the other end of the world Pentagon is actively collecting information trying to predict the monster's behavior and preparing the secret super weapon. Due to high seismic activity and poor weather conditions the satellites haven't yet been able to make clear shots of the monster. The analysis of the first shot resulted in an undirected graph with n vertices and m edges. Now the world's best minds are about to determine whether this graph can be regarded as Cthulhu or not.

To add simplicity, let's suppose that Cthulhu looks from the space like some spherical body with tentacles attached to it. Formally, we shall regard as Cthulhu such an undirected graph that can be represented as a set of three or more rooted trees, whose roots are connected by a simple cycle.

It is guaranteed that the graph contains no multiple edges and self-loops.

Input

The first line contains two integers — the number of vertices n and the number of edges m of the graph ($1 \le n \le 100$, $0 \le m \le \frac{n \cdot (n-1)}{2}$).

Each of the following m lines contains a pair of integers x and y, that show that an edge exists between vertices x and y ($1 \le x, y \le n, x \ne y$). For each pair of vertices there will be at most one edge between them, no edge connects a vertex to itself.

Output

Print "NO", if the graph is not Cthulhu and "FHTAGN!" if it is.

Sample test(s)

input

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

output

```
FHTAGN!
```

input

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

output

```
NO
```

Note

Let us denote as a simple cycle a set of v vertices that can be numbered so that the edges will only exist between vertices number 1 and 2, 2 and 3, ..., v - 1 and v, v and 1.

A tree is a connected undirected graph consisting of n vertices and n - 1 edges ($n > 0$).

A rooted tree is a tree where one vertex is selected to be the root.

SPOJ Problem Set (classical)

18170. Takeover Wars

Problem code: TKV1000

You are studying a takeover war between two large corporations, Takeover Incorporated and Buyout Limited. Each of these corporations controls a number of subsidiaries. The aim in this war is simply to drive the competition out of the market. There are N subsidiaries of Takeover Incorporated and M subsidiaries of Buyout Limited, and you know the market value of each subsidiary.

Each company can designate one of its subsidiaries to perform a takeover. The takeover can either be friendly or hostile. A friendly takeover means a subsidiary of a corporation merges with a different subsidiary of the same corporation. The market value of the merged subsidiary is the sum of the market values of the constituent subsidiaries. There is no constraint on the relative sizes of the subsidiaries participating in a friendly takeover.

A hostile takeover means a subsidiary A of a corporation attempts to take over a subsidiary B of the other corporation. For this to succeed, the market value of A has to be greater than the market value of B. After this move, B disappears from the market. The market value of A does not change (the gain of incorporating B's assets is offset by the monetary cost of the takeover). For simplicity we assume that no sequence of moves leads to two subsidiaries of different corporations having the same market value. The companies take turns making moves in this takeover war, with Takeover Incorporated going first. A company will do nothing on its turn only if it cannot make a takeover. A company loses the takeover war if all its subsidiaries are taken over.

Your aim is to learn which company can guarantee a victory from this war. In the first case of the sample data, Takeover Incorporated can simply take over one of the companies of Buyout Limited in its first move with the 7-value subsidiary. Then it will lose one of its small (1-value) subsidiaries to a hostile takeover, and then it will take over the second subsidiary of Buyout Limited. In the second case, Takeover has to make a friendly takeover in its first move. Buyout Limited will join its two subsidiaries into a single company with market value 10. Takeover will have to make a friendly takeover again (as again it will not have a large enough subsidiary to take over Buyout's giant). Now Takeover will have two subsidiaries, valued either 9 and 3 or 6 and 6. In either case, Buyout takes over one of these subsidiaries, Takeover has to pass, and Buyout takes over the other one.

Input

Each test case is described by three lines of input. The first line contains two numbers $1 \le N \le 10^5$ and $1 \le M \le 10^5$ denoting respectively the number of subsidiaries of Takeover Incorporated and Buyout Limited. The next line lists the N sizes a_i of the subsidiaries of Takeover Incorporated ($1 \le a_i \le 10^{12}$), and the third line lists the M sizes b_j of the subsidiaries of Buyout Limited ($1 \le b_j \le 10^{12}$).

Output

For each test case, display the case number and either the phrase Takeover Incorporated or the phrase Buyout Limited depending on who wins the takeover war if both corporations act optimally.

Example

Input:
```
3 2

7 1 1

5 5

4 2

3 3 3 3

5 5
```

Output:
```
Case 1: Takeover Incorporated

Case 2: Buyout Limited
```

show comments