Greetings From Globussoft

- Given below are 5 Programming questions, you have to solve any 3 out of 5 questions.

- These 5 questions you can attempt in any technology like C/C++, java, .Net, PHP

- To solve these 3 questions you've max. 3 hours.

- While Solving these questions you are not allowed to use any **Search Engine** like Google, Yahoo, Bing …

All the best for your test

Globussoft

QUESTION – 1

The connections between mathematics and biology are complicated. Most of the time they do not run along nice-looking links that merrily join at first glance, but they are abstract and not always easily established.

Lake Vostok - about 14000 square kilometers large, up to 650 meters deep, and covered by 3743 meters of ice - was recently discovered on the Antarctic continent. The lake remained under conditions of high pressure and no sunlight for several millions of years. It is believed that ordinary life has evolved to a more efficient form using a genetic code composed of only three bases (the current state of ignorance proclaims the four bases adenine, cytosine, guanine, and thymine). Until reasonable names are found, the three bases will be abbreviated as `N`, `O`, and `P`.

Moreover, the genome is single-stranded and directed, i.e., we may see it as a sequence over the alphabet `{N,O,P}`. Unless risking instability, it is necessary that the genome is a Thue-sequence, due to the Norwegian mathematician A. Thue (1863-1922). Define a subsegment of a sequence to be a connected subsequence, and call two subsegments adjacent if one follows immediately after the other in the sequence. A Thue-sequence is a sequence where no adjacent subsegments are equal. For example, `NOPNO` is and `NOPNPNO` is not a Thue-sequence, so that the first may be a genome whereas the second may not.

To be able to simulate experiments with the new genomes, you are asked to generate genomes of certain lengths.

Input Specification

The input contains several test cases. Each test case consists of an integer `n`. You may assume that `1<=n<=5000`. The last test case is followed by a zero.

Output Specification

For each test case specified by `n` output on a single line any genome of length `n`. If no genome of length `n` exists, output a blank line instead.

Sample Input

```
1
2
10
20
0
```

Sample Output

```
N
NO
NONPNOPNPO
```

QUESTION – 2

My birthday is coming up and traditionally I'm serving pie. Not just one pie, no, I have a number *N* of them, of various tastes and of various sizes. *F* of my friends are coming to my party and each of them gets a piece of pie. This should be one piece of one pie, not several small pieces since that looks messy. This piece can be one whole pie though.

My friends are very annoying and if one of them gets a bigger piece than the others, they start complaining. Therefore all of them should get equally sized (but not necessarily equally shaped) pieces, even if this leads to some pie getting spoiled (which is better than spoiling the party). Of course, I want a piece of pie for myself too, and that piece should also be of the same size.

What is the largest possible piece size all of us can get? All the pies are cylindrical in shape and they all have the same height 1, but the radii of the pies can be different.

Input

One line with a positive integer: the number of test cases. Then for each test case:

- One line with two integers N and F with $1 \leq N, F \leq 10000$: the number of pies and the number of friends.
- One line with N integers r_i with $1 \leq r_i \leq 10000$: the radii of the pies.

Output

For each test case, output one line with the largest possible volume V such that me and my friends can all get a pie piece of size V. The answer should be given as a floating point number with an absolute error of at most 10^{-3}.

Example

Input:

```
3
3 3
4 3 3
1 24
5
10 5
1 4 2 3 4 5 6 5 4 2
```

Output:

```
25.1327
3.1416
50.2655
```

QUESTION – 3

Project managers, such as the UNIX utility `make`, are used to maintain large software projects made up from many components. Users write a *project file* specifying which components (called *tasks*) depend on others and the project manager can automatically update the components in the correct order.

Problem

Write a program that reads a project file and outputs the order in which the tasks should be performed.

Input

For simplicity we represent each task by an integer number from $1, 2, \ldots, N$ (where N is the total number of tasks). The first line of input specifies the number N of tasks and the number M of rules, such that $N \leq 100,\ M \leq 100$.

The rest of the input consists of M *rules*, one in each line, specifying dependencies using the following syntax:

T_0 k T_1 T_2 ... T_k

This rule means that task number T_0 depends on k tasks $T_1, T_2, \ldots T_k$ (we say that task T_0 is the *target* and $T_1 \ldots T_k$ are *dependents*).

Note that tasks numbers are separated by single spaces and that rules end with a newline. Rules can appear in any order, but each task can appear as target only once.

Your program can assume that there are no circular dependencies in the rules, i.e. no task depends directly or indirectly on itself.

Output

The output should be a single line with the permutation of the tasks $1 \ldots N$ to be performed, ordered by dependencies (i.e. no task should appear before others that it depends on).

To avoid ambiguity in the output, tasks that do not depend on each other should be ordered by their number (lower numbers first).

Example

Input:

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

Output:

```
1 5 3 2 4
```

QUESTION – 4

A *palindrome* is a sequence that is the same when read forward or backward. For example, "pop" is a palindrome, as are "Poor Dan is in a droop" (ignoring spaces and case), and "12321".

In this problem, you are to find the "cheapest" way to transform a sequence of decimal digits into a palindrome. There are only two types of modifications you may make to the sequence, but each of these may be repeated as many times as necessary. You may delete a digit from either end of the sequence, or you may add a digit to either end of the sequence. Each of these operations incurs a "cost" of 1. For each input sequence, determine the smallest cost of transforming the sequence into a palindrome, and the length of the resulting palindrome. If two palindromes can be produced with the same cost, the length of the longer palindrome (the one with more digits) is to be reported.

For example, suppose the initial sequence was "911". This can be transformed into a palindrome by deleting the leading "9" (yielding "11") or by adding an additional "9" to the right end of the sequence (yielding "9119"). Since both of these transformations have a cost of 1, and the second transformation yields a longer palindrome, it is this one which would be reported as your result.

Note that the particular palindrome produced by the cheapest sequence of transformations is not necessarily unique, but since you are not required to report the resulting palindrome, any of these will suffice.

Input

There will be multiple cases to consider. Each case has a single line of input that contains one or more decimal digits followed by the end of line. The maximum number of digits in a sequence will be 6. The last case is followed by an empty line (that is, only an end of line).

Output

For each input case, display the case number (1, 2, …), the input sequence, the cost of the cheapest transformation, and the length of the resulting palindrome. Your output should follow the format shown in the examples below.

Example

```
Input:
911
9118
11234
<-- This line is blank

Output:
Case 1, sequence = 911, cost = 1, length = 4
Case 2, sequence = 9118, cost = 2, length = 4
Case 3, sequence = 11234, cost = 3, length = 8
```

QUESTION – 5

You are given a tree (a connected, acyclic graph) along with a set of **commodities**, i.e. pairs of vertices, $(s_1,t_1),...,(s_m,t_m)$ $(s_i \neq t_i)$. A **multicut** is a set of edges that when removed disconnects s_i from t_i for all i. There is a unique path $P_{u,v}$ between every pair of vertices u,v in a tree, and the **max-cost** of a multicut S is $\max_i |S \cap P_{si,\,ti}|$. You will be given a rooted tree of height 1 and a set of commodities and must return the minimum possible max-cost over all multicuts.

Input

The first line of the input is "$N\ M$" ($1 \leq N, M \leq 100000$), where N is the number of vertices in the tree and M is the number of commodities. All vertices are numbered $0, ...,N-1$, and the root has label $N - 1$. M lines then follow, where the ith line is "$s_i\ t_i$", representing a commodity (s_i, t_i) where $s_i \neq t_i$. Commodities are distinct: neither $(s_i, t_i) = (s_j, t_j)$ nor $(s_i, t_i) = (t_j, s_j)$ will hold when $i \neq j$.

Output

Your output should consist of a single number, the minimum possible max-cost of a multicut, followed by a newline.

Example

Input:

```
10 2
0 5
4 8
```

Output:

```
1
```