Problem A
Air Raid

Input: standard input
Output: standard output

Consider a town where all the streets are one-way and each street leads from one intersection to another. It is also known that starting from an intersection and walking through town's streets you can never reach the same intersection i.e. the town's streets form no cycles.

With these assumptions your task is to write a program that finds the minimum number of paratroopers that can descend on the town and visit all the intersections of this town in such a way that more than one paratrooper visits no intersection. Each paratrooper lands at an intersection and can visit other intersections following the town streets. There are no restrictions about the starting intersection for each paratrooper.

Input

Your program should read sets of data. The first line of the input file contains the number of the data sets. Each data set specifies the structure of a town and has the format:

no_of_intersections
no_of_streets
S_1 E_1
S_2 E_2
……
$S_{no\_of\_streets}$ $E_{no\_of\_streets}$

The first line of each data set contains a positive integer **no_of_intersections** (greater than 0 and less or equal to 120), which is the number of intersections in the town. The second line contains a positive integer **no_of_streets,** which is the number of streets in the town. The next **no_of_streets** lines, one for each street in the town, are randomly ordered and represent the town's streets. The line corresponding to street **k** (**k ≤ no_of_streets**) consists of two positive integers, separated by one blank: $\mathbf{S_k}$ **(1 ≤** $\mathbf{S_k}$ **≤ no_of_intersections)** - the number of the intersection that is the start of the street, and $\mathbf{E_k}$ **(1 ≤** $\mathbf{E_k}$ **≤ no_of_intersections)** – the number of the intersection that is the end of the street. Intersections are represented by integers from **1** to **no_of_intersections**.

There are no blank lines between consecutive sets of data. Input data are correct.

Output

The result of the program is on standard output. For each input data set the program prints on a single line, starting from the beginning of the line, one integer: the minimum number of paratroopers required to visit all the intersections in the town.

Sample Input

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

Sample Output

```
2
1
```

Problem B
Big Number

Input: standard input
Output: standard output

In many applications very large integers numbers are required. Some of these applications are using keys for secure transmission of data, encryption, etc. In this problem you are given a number, you have to determine the number of digits in the factorial of the number.

Input

Input consists of several lines of integer numbers. The first line contains an integer n, which is the number of cases to be tested, followed by n lines, one integer $1 \leq n \leq 10^7$ on each line.

Output

The output contains the number of digits in the factorial of the integers appearing in the input.

Sample Input

```
2
10
20
```

Sample Output

```
7
19
```

Problem C
Chat Rooms

Input: standard input
Output: standard output

Suppose there are a number of good users using an Internet chat room, which is a shared text chat space. Now there are some bad guys in the Internet who would come to this room and would have no respect or interest in the topic of discussion in the group. They would start typing strings of garbage characters, start unsolicited advertisements of certain URLs or businesses, or paste same string again and again just to disturb others in the room.

When I asked about this problem to Prof Banku (naturally not as smart as Prof Shanku) he suggested the following scheme. Let us define consonants to be all letters in a-z and A-Z except a, e, i, o, u, y, A, E, I, O, U and Y.

A user should be able to send a line to the chat room unless

- the line has more than 5 consecutive consonants, or

- the line contains at least one word which has more than 4 consecutive consonants and the user sent more than 2 such lines in last 10 lines sent, or

- the user sent the same line more than once in last 10 sentences sent.

Input

Input consists of several lines. The first line contains an integer *n*, which is the number of lines in the shared text space. Then follow *n* lines each consisting of not more than 200 ASCII characters.

Output

For each line of input print y if the line is acceptable according to the criteria set above and n otherwise in separate lines. See the sample output for example.

Sample Input

```
12
hello
how r u?
where r u from?
```

kjhh kh kgkjhg jhg

where r u from?

i am from London, Ontario, Canada

how r you nxw?

now

where r u from?

kjhh kh kgkjhg jhg

very good

it is very cold here.

Sample Output

y
y
y
n
y
y
y
y
n
n
y
y

Problem D

Diamonds

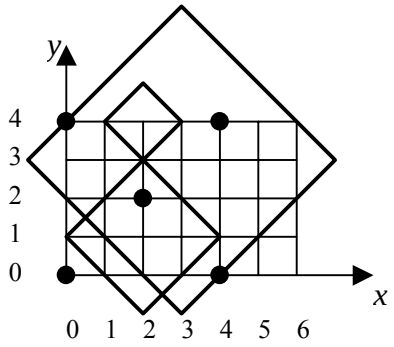
Input: standard input
Output: standard output

Consider a planer grid of M rows and N columns. This grid has exactly MN intersections, each of which is denoted by a pair of coordinates (i, j) where $i = 0, 1, \ldots, N\text{-}1$ and $j = 0, 1, \ldots, M\text{-}1$.

Suppose $K \leq MN$ points are now placed on K distinct grid intersections.

Consider a diamond shaped area $D(i, j, r)$ such that the centre of the area is at intersection (i, j) and the shape itself is a square of diagonal length $2r$, rotated 45 degree clockwise, where $i = 0, 1, \ldots, N\text{-}1$, $j = 0, 1, \ldots, M\text{-}1$, and r, the *radius of the diamond shape*, is any integer greater than 0.

Manoranjan is interested in finding the minimum radius, $R_{\min}(P_{\min})$, of such a diamond shape that would guarantee of covering at least $P_{\min}$ points no matter at which grid intersection the centre of the shape resides, where $P_{\min} = 1, 2, \ldots, K$. Let $P_{\max}(P_{\min})$ be the maximum number of points that can be covered by a diamond shape of radius $R_{\min}(P_{\min})$. Consider the example in the figure below. Five points at intersections (0,0), (4,0), (2,2), (0,4), and (4,4) are placed on a planer grid of 5 rows and 7 columns. The diamond shape $D(2,4,1)$ covers none of the points, $D(2,1,2)$ covers one point, and $D(3,3,4)$ covers four points.



Input

The number of rows, M, and columns, N, of the grid are given in line 1. K number of points are then given in the next one or more lines, where $K \leq MN$. You may assume that $1 \leq M \leq 100$ and $1 \leq N \leq 100$. A pair of integers, representing x- and y-coordinates of the point respectively, denotes each point.

All the values given in any of the input lines are separated from each other by one or more white spaces including tabs.

Output

Any given point outside the grid is discarded as "invalid" entries, e.g., the last point given in the sample input is discarded. For each possible P_{min} value, corresponding $R_{min}(P_{min})$ and $P_{max}(P_{min})$ values are written in a separate line as per the format shown in the sample output. However, a line of output should only be generated if a diamond shape of radius $R_{min}(P_{min})$ guarantees to cover exactly P_{min} points. In the sample output, no line is generated for $P_{min} = 2$ as $R_{min}(2) = R_{min}(3) = 6$.
All values are written right aligned to their column headings.

Sample Input

```
5    7
0    0 0  4
4 0
4 4      2 2
     0 5
```

Sample Output

```
Pmin⌂⌂Rmin(Pmin)⌂⌂Pmax(Pmin)
⌂⌂⌂1⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂4⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂5
⌂⌂⌂3⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂6⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂5
⌂⌂⌂4⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂8⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂5
⌂⌂⌂5⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂10⌂⌂⌂⌂⌂⌂⌂⌂⌂⌂5
```

where each ⌂ represents a single space.

Problem E
Enigmatic Travel

Input: standard input
Output: standard output

Suhan and Laina live in an **n**-dimensional city where there are **(n+1)** locations. Any two locations (consider these locations as points) are equidistant from each other and connected by only one bi-directional road. They love to roam together around the city on their favourite bi-verbal (A kind of vehicle). Kiri, a tenth generation robot also lives in the same city and wants to kill Suhan out of jealousy. That is why Suhan and Laina are very careful about keeping their thoughts and plans secret. Therefore nobody knows a) Where Suhan and Laina lives. b) What their destination location is. c) Which roads will they use?

So their journey can start from any location, ends in another location and they may use any road sequence they like. Their destination location may be same or different than the source location. For example when their tour is guaranteed to be a simple cycle their source and destination location are same.

Given the number of locations in the city **(L)** you will have to find the expected cost (often considered as average) of one of their single travelling. You can assume that the cost of travelling from one location to another through the direct (also shortest) path is 1 universal joule.

Input

The input file contains several lines of input. Each line contains a single integer **L(15>=L>2)** that indicates the number of locations in the city. Input is terminated by a line where value of **L** is zero. This line should not be processed.

Output

For each line of input produce one line of output. This line contains three floating-point numbers **F1**, **F2**, **F3**. Here **F1** is the expected cost when they travel along a path, **F2** is the expected cost when it is guaranteed that they travel along a simple path and **F3** is the expected cost when it is guaranteed that they travel along a simple cycle. All the floating-point numbers should be rounded up to four digits after the decimal point. You must assume that their travelling cost is not greater than **(L)**. Travelling cost is always expressed in universal joule.

Sample Input

```
3
4
5
0
```

Sample Output

```
2.4286 1.5000 3.0000
3.5500 2.2000 3.5000
4.6716 3.0625 4.2000
```

Problem F
Find The Multiple

Input: standard input
Output: standard output

Given a positive integer *n*, write a program to find out a nonzero multiple *m* of *n* whose decimal representation contains only the digits 0 and 1. You may assume that *n* is not greater than 200 and there is a corresponding *m* containing no more than 100 decimal digits.

Input

The input file may contain multiple test cases. Each line contains a value of *n* ($1 \le n \le 200$). A line containing a zero (0) terminates the input.

Output

For each value of *n* in the input print a line containing the corresponding value of *m*. The decimal representation of *m* must not contain more than 100 digits. If there are multiple solutions for a given value of *n*, any one of them is acceptable.

Sample Input

```
2
6
19
0
```

Sample Output

```
10
100100100100100100
111111111111111111
```

Problem G
Get Them All

Input: standard input
Output: standard output

In order to ensure that the contestants find it easy to reach the regional contest site, the authority has prepared some robot-driven vehicles. The vehicles would visit n pre-determined junctions and carry the contestants waiting there to the contest. There is a computer controlled Transportation Center (TC) that determines the number of seats of each vehicle and the time each vehicle leaves the contest site for the first time.

When a new vehicle is needed, a request is made to the TC. Every new vehicle has fewer number of seats than the last vehicle if it is more than 3: the i-th vehicle has $max(s-(i-1)*t, 3)$ seats (i= 1, 2, 3, ...). The first vehicle leaves the contest site just at 8:00am. When TC gets a request for a new vehicle, it prepares a new vehicle and right after 2 seconds of getting the request, the new vehicle leaves the contest site. If multiple requests are made at the same time, only one is considered.

At the junction j, each vehicle does the following tasks. If there are more than one vehicle at j at the same time, they perform the tasks in order of their service times: the one with the longest service time goes first. Service time of a vehicle is the difference between current time and the time the vehicle left the contest site (which is at junction 0) for the first time.

1. If $j = 0$ (the contest site), all the contestants in the vehicle gets down. Otherwise, the vehicle picks up as many contestants as it can (i.e. until the vehicle is full or there are no contestants left at junction j).

2. If, after that, there are any contestant left at junction j ($j > 0$), the vehicle sends a request for a new vehicle to the TC.

3. Finally the vehicle starts moving towards the next junction k, which is selected by the robot-driver in the following way (even at junction 0):
   If the vehicle is full, $k = 0$.
   Otherwise, if no other vehicle has left junction j yet, $k = (j+1) \bmod n$.
   Otherwise, k is $((k_0+1) \bmod n)$ if it is different from j.
   Otherwise, k is $((k_0+2) \bmod n)$.
   (Here, k_0 is the "next junction" selected by the last vehicle leaving junction j).

Vehicles do the above 3 tasks instantly (i.e. in 0 seconds). Time needed to go from each junction to any other junction is known. All the contestants reach a suitable junction by 8:00am and don't go away until they are picked up by any vehicle. Given the number of contestants waiting at each junction and a time limit, you are to determine when everyone reaches the contest or how many have reached the contest by the time limit.

Input

Input consists of several datasets. Each dataset consists of the followings:

- A line containing the name of the set (which has 2 to 20 alphanumeric characters).
- A line containing 3 positive integers *n*, *s* and *t* ($2 < n < 11$).
- Each of next *n* lines contains *n*-1 integers each. The *i*-th line (*i*= 1, 2, 3, ...) contains the time (in seconds) needed to go from the junction *i*-1 to all other junctions (except the (*i*-1)-th) in order 0, 1, 2, ... *n*.
- Next *n*-1 lines each contain a non-negative integer. The *i*-th line (*i*= 1, 2, 3, ...) contains the number of contestants waiting at the *i*-th junction.
- The last line of the dataset contains the time limit (in seconds, less than 10000000).

All integers on a single line are separated by exactly one space. Total number of contestants is at most 1000.

The end of input is marked with a line consisting of "`TheEnd`".

Output

For each set, print 2 lines. The first line contains the name of the set as it appears in the input. The second line contains the time (in seconds) needed to bring all the contestants to the contest, if the time is not more than the given time limit. Otherwise, print the number of contestants reached the contest by the time limit (See the sample output).

Sample Input

```
Dhaka2000
3 22 4
30 8
10 30
28 8
20
20
100
Dhaka2001
3 22 4
30 8
10 30
28 8
20
20
90
```

```
Dhaka2002
3 22 2
30 8
10 30
28 8
20
20
100
TheEnd
```

Sample Output

```
Dhaka2000
98 seconds needed
Dhaka2001
22 contestants reached
Dhaka2002
88 seconds needed
```

Problem H

Hermes' Colony

Input: standard input
Output: standard output

Hermes, the Greek God of Speed, has created a two-dimensional colony Massilia in space. The colony, consisting of one or more provinces, can be represented by a linear equation in 3-dimensional space. In each province there are 3 or 4 cities, each being on their convex hull. Now, citizens of each province want to create road network connecting cities of their own province. Unfortunately, materials for road construction are not available in the colony, and can only be transported from Earth. Total material required to construct the roads will be proportional to their length. This is why they are interested to build road networks of shortest length connecting different cities of a province. In order to minimize length of each network they are also ready to make junctions away from cities, if so required.

Unfortunately creatures of the colony are pretty weak in mathematics and algorithms. They have, therefore, decided to take services of *homo sapiens* of earth, who are believed to be good in mathematics and algorithms. Accordingly the chief of Massilia sent an email to Academy of Computers and Mathematics(ACM) with Head Quarters at Dhaka to solve the problem. ACM has now asked you to help out our friends in Massilia.

Input

The colony is described by an equation $ax+by+cz=d$. There are N provinces in the colony. A city in a province can be described by a point in 3-dimension like (x, y, z). All the x, y and z co-ordinates will be within –100.00 and +100.00. The first line of input will contain a, b, c and d. Then the number of provinces (N) in the colony will be given in the second line. The remaining lines will describe each province one by one.

The description of a province starts with the number of cities M $(3 \leq M \leq 4)$ in the province in a line followed by M lines. Each line contains 3 numbers describing respectively the x, y and z co-ordinates of a city in that province.

Output

You should print one line for each province in the input data as follows:

Province # p : L

Where p is the serial number of the province as they appear in the input. $1 \leq p \leq N$, and L is the minimum length of the road network for that province. L should contain two digits after the decimal point and should be exact up to 2 decimal points.

Sample Input

```
-0.126826 -0.780330 0.612372 3.000000
2
3
11.593475 -0.702393 6.405027
-43.361881 -34.677124 -48.269711
-0.380480 -2.340990 1.837117
4
-15.033179 6.549108 10.130860
-13.950171 -53.592907 -66.282234
49.017246 0.979824 16.299353
46.824024 12.971205 31.125420
```

Sample Output

```
Province # 1 : 86.43
Province # 2 : 175.15
```