PA

The government of Nova Mareterrania requires that various legal documents have stamps attached to them so that the government can derive revenue from them. In terms of recent legislation, each class of document is limited in the number of stamps that may be attached to it. The government wishes to know how many different stamps, and of what values, they need to print to allow the widest choice of values to be made up under these conditions. Stamps are always valued in units of \$1.

This has been analysed by government mathematicians who have derived a formula for $n(h,k)$, where h is the number of stamps that may be attached to a document, k is the number of denominations of stamps available, and n is the largest attainable value in a continuous sequence starting from \$1. For instance, if $h = 3$, $k = 2$ and the denominations are \$1 and \$4, we can make all the values from \$1 to \$6 (as well as \$8, \$9 and \$12). However with the same values of h and k, but using \$1 and \$3 stamps we can make all the values from \$1 to \$7 (as well as \$9). This is maximal, so $n(3,2) = 7$.

Unfortunately the formula relating $n(h,k)$ to h, k and the values of the stamps has been lost — it was published in one of the government reports but no-one can remember which one, and of the three researchers who started to search for the formula, two died of boredom and the third took a job as a lighthouse keeper because it provided more social stimulation.

The task has now been passed on to you. You doubt the existence of a formula in the first place so you decide to write a program that, for given values of h and k, will determine an optimum set of stamps and the value of $n(h,k)$.

Input

Input will consist of several lines, each containing a value for h and k. The file will be terminated by two zeroes (0 0). For technical reasons the sum of h and k is limited to 9. (The President lost his little finger in a shooting accident and cannot count past 9).

Output

Output will consist of a line for each value of h and k consisting of the k stamp values in ascending order right justified in fields 3 characters wide, followed by a space and an arrow (`->`) and the value of $n(h,k)$ right justified in a field 3 characters wide.

Sample Input

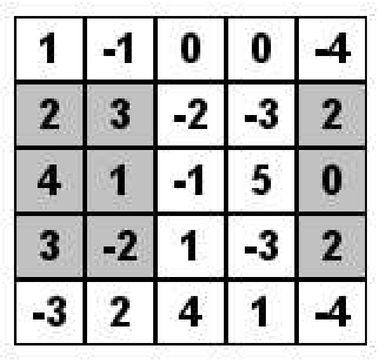
```
3 2
0 0
```

Sample Output

```
  1  3 ->  7
```

PB

A grid that wraps both horizontally and vertically is called a torus. Given a torus where each cell contains an integer, determine the sub-rectangle with the largest sum. The sum of a sub-rectangle is the sum of all the elements in that rectangle. The grid below shows a torus where the maximum sub-rectangle has been shaded.



1	-1	0	0	-4
2	3	-2	-3	2
4	1	-1	5	0
3	-2	1	-3	2
-3	2	4	1	-4

Input

The first line in the input contains the number of test cases (at most 18). Each case starts with an integer N ($1 \leq N \leq 75$) specifying the size of the torus (always square). Then follows N lines describing the torus, each line containing N integers between -100 and 100, inclusive.

Output

For each test case, output a line containing a single integer: the maximum sum of a sub-rectangle within the torus.

Sample Input

```
2
5
1 -1 0 0 -4
2 3 -2 -3 2
4 1 -1 5 0
3 -2 1 -3 2
-3 2 4 1 -4
3
1 2 3
4 5 6
7 8 9
```

Sample Output

```
15
45
```

PC

Running a taxi station is not all that simple. Apart from the obvious demand for a centralised coordination of the cabs in order to pick up the customers calling to get a cab as soon as possible, there is also a need to schedule all the taxi rides which have been booked in advance. Given a list of all booked taxi rides for the next day, you want to minimise the number of cabs needed to carry out all of the rides.

For the sake of simplicity, we model a city as a rectangular grid. An address in the city is denoted by two integers: the street and avenue number. The time needed to get from the address a, b to c, d by taxi is $|a - c| + |b - d|$ minutes. A cab may carry out a booked ride if it is its first ride of the day, or if it can get to the source address of the new ride from its latest, at least one minute before the new ride's scheduled departure. Note that some rides may end after midnight.

Input

On the first line of the input is a single positive integer N, telling the number of test scenarios to follow. Each scenario begins with a line containing an integer M, $0 < M < 500$, being the number of booked taxi rides. The following M lines contain the rides. Each ride is described by a departure time on the format $hh : mm$ (ranging from `00:00` to `23:59`), two integers a b that are the coordinates of the source address and two integers c d that are the coordinates of the destination address. All coordinates are at least 0 and strictly smaller than 200. The booked rides in each scenario are sorted in order of increasing departure time.

Output

For each scenario, output one line containing the minimum number of cabs required to carry out all the booked taxi rides.

Sample Input

```
2
2
08:00 10 11 9 16
08:07 9 16 10 11
2
08:00 10 11 9 16
08:06 9 16 10 11
```

Sample Output

```
1
2
```

PD

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 oating point number with an absolute error of at most 10^{-3}.

Sample Input

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

Sample Output

```
25.1327
3.1416
50.2655
```

PE

The **ACM (Asian Cultural Museum)** authority is planning to install fire exits in its galleries in order to handle the emergency situation arising in case of a sudden fire. The museum is a collection of numerous interconnected galleries. The galleries are connected by corridors in such a way that from any gallery there is exactly one path to reach any other gallery without visiting any intermediate gallery (a gallery that is on that path) more than once.

However, in order to reduce installation cost, it has been decided that not every gallery will have a fire exit. Fire exits will be installed in such a way that if any gallery does not have a fire exit then at least one of its adjacent galleries must have one and for each corridor at least one of the two galleries it connects must have a fire exit. You are hired to determine where to put the fire exits under this constraint.

However, as a first step, you are expected to determine the minimum number of fire exits required.

Input

The input file may contain multiple test cases. The first line of each test case contains an integer N $(1 \leq N \leq 1,000)$ indicating the number of galleries in this test case. Then follow N lines where the i-th $(1 \leq i \leq N)$ line is the adjacency list of the i-th gallery (Each gallery is given a unique identification number from 1 to N for convenience). The adjacency list for gallery i starts with an integer n_i $(1 \leq n_i \leq N-1)$ indicating the number of galleries adjacent to this gallery, followed by n_i integers giving the identification numbers of those galleries.

A test case containing a zero for N terminates the input.

Output

For each test case in the input file print a line containing the minimum number of fire exits required to meet the given constraint.

Sample Input

```
4
3 2 3 4
1 1
1 1
1 1
16
4 6 12 15 16
3 3 8 10
4 2 4 6 9
1 3
1 6
3 1 3 5
1 15
1 2
1 3
1 2
1 16
```

```
1 1
1 15
1 15
4 1 7 13 14
2 1 11
0
```

Sample Output

```
1
6
```