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

In the two-player game "Two Ends", an even number of cards is laid out in a row. On each card, face up, is written a positive integer. Players take turns removing a card from either end of the row and placing the card in their pile. The player whose cards add up to the highest number wins the game. Now one strategy is to simply pick the card at the end that is the largest — we'll call this the greedy strategy. However, this is not always optimal, as the following example shows: (The first player would win if she would first pick the 3 instead of the 4.)

3 2 10 4

You are to determine exactly how bad the greedy strategy is for different games when the second player uses it but the first player is free to use any strategy she wishes.

Input

There will be multiple test cases. Each test case will be contained on one line. Each line will start with an even integer n followed by n positive integers. A value of $n = 0$ indicates end of input. You may assume that n is no more than 1000. Furthermore, you may assume that the sum of the numbers in the list does not exceed 1,000,000.

Output

For each test case you should print one line of output of the form:

In game m, the greedy strategy might lose by as many as p points.

where m is the number of the game (starting at game 1) and p is the maximum possible difference between the first player's score and second player's score when the second player uses the greedy strategy. When employing the greedy strategy, always take the larger end. If there is a tie, remove the left end.

Example

Input:

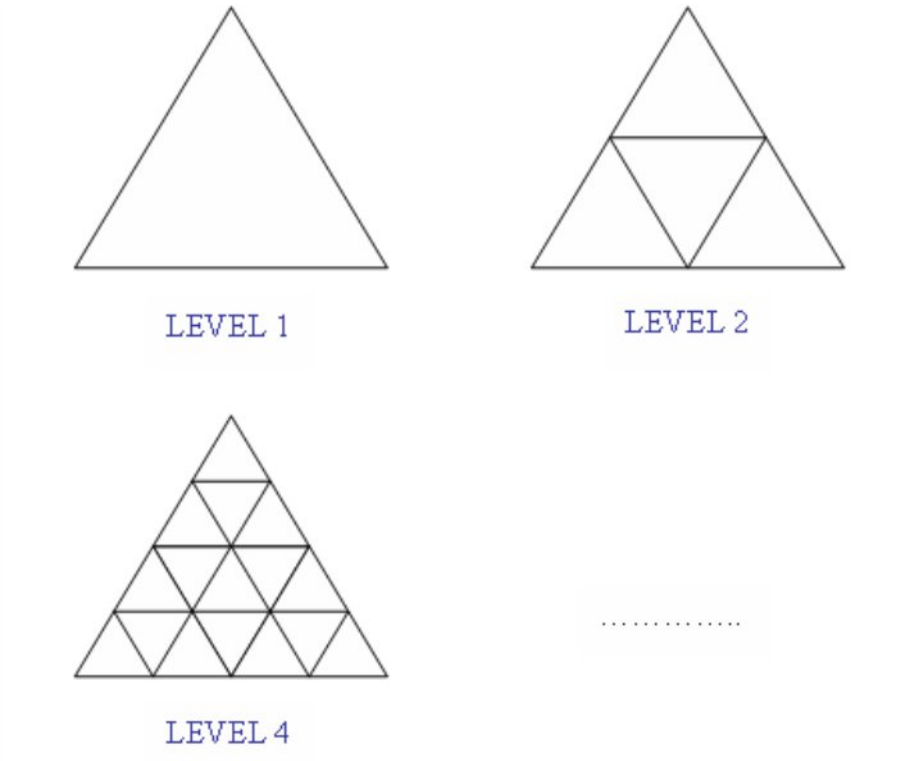
```
4 3 2 10 4
8 1 2 3 4 5 6 7 8
8 2 2 1 5 3 8 7 3
0
```

Output:

```
In game 1, the greedy strategy might lose by as many as 7 points.
In game 2, the greedy strategy might lose by as many as 4 points.
In game 3, the greedy strategy might lose by as many as 5 points.
```

QUESTION – 2

We define the LEVEL of a triangle as in the following illustrative image:



Task: Your task is very easy. All you have to do is to count all triangles in the biggest one (Level N).

Input

The first line of the input contains an integer T ($T \leq 10000$) - the number of test cases and T lines follow. Each line contains an integer N ($1 \leq N \leq 10^6$) which is the level of the triangle in that test case.

Output

For each test case, you should write a seperate line: the number of triangles in the biggest one (Level N). (All answers will fit within the range of a 64-bit integer)

Example

```
Input:
3
1
2
3
```

Output:
```
1
5
13
```

QUESTION – 3

Given two strings of lowercase letters, ***a*** and ***b***, print the longest string ***x*** of lowercase letters such that there is a permutation of ***x*** that is a subsequence of ***a*** and there is a permutation of ***x*** that is a subsequence of ***b***.

Input

Input file contains several lines of input. Consecutive two lines make a set of input. That means in the input file line **1** and **2** is a set of input, line **3** and **4** is a set of input and so on. The first line of a pair contains ***a*** and the second contains ***b***. Each string is on a separate line and consists of at most **1000** lowercase letters.

Output

For each set of input, output a line containing ***x***. If several ***x*** satisfy the criteria above, choose the first one in alphabetical order.

Example

Sample input:
```
pretty
women
walking
down
the
street
```

Sample output:
```
e
nw
et
```

QUESTION – 4

You're in charge of designing a campus network between buildings and are very worried about its reliability and its cost. So, you've decided to build some redundancy into your network while

keeping it as inexpensive as possible. Specifically, you want to build the cheapest network so that if any one line is broken, all buildings can still communicate. We'll call this a *minimal reliable net*.

Input

There will be multiple test cases for this problem. Each test case will start with a pair of integers n (<=15) and m (<= 20) on a line indicating the number of buildings (numbered 1 through n) and the number of potential inter-building connections, respectively. (Values of n = m = 0 indicate the end of the problem.) The following m lines are of the form b_1 b_2 c (all positive integers) indicating that it costs c to connect building b_1 and b_2. All connections are bidirectional.

Output

For each test case you should print one line giving the cost of a minimal reliable net. If there is a minimal reliable net, the output line should be of the form:

The minimal cost for test case p is c.

where *p* is the number of the test case (starting at 1) and *c* is the cost. If there is no reliable net possible, output a line of the form:

There is no reliable net possible for test case p.

Example

Input:

```
4 5
1 2 1
1 3 2
2 4 2
3 4 1
2 3 1
2 1
1 2 5
0 0
```

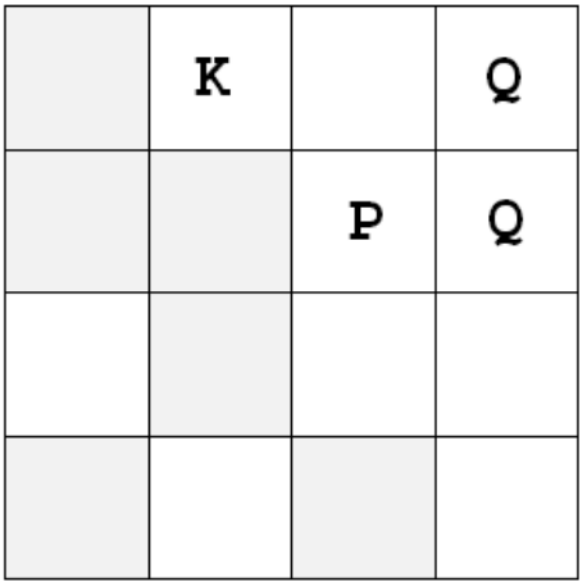
Output:

```
The minimal cost for test case 1 is 6.
There is no reliable net possible for test case 2.
```

QUESTION – 5

You all are familiar with the famous 8-queens problem which asks you to place 8 queens on a chess board so no two attack each other. In this problem, you will be given locations of queens

and knights and pawns and asked to find how many of the unoccupied squares on the board are not under attack from either a queen or a knight (or both). We'll call such squares "safe" squares. Here, pawns will only serve as blockers and have no capturing ability. The board below has 6 safe squares. (The shaded squares are safe.)



Recall that a knight moves to any unoccupied square that is on the opposite corner of a 2x3 rectangle from its current position; a queen moves to any square that is visible in any of the eight horizontal, vertical, and diagonal directions from the current position. Note that the movement of a queen can be blocked by another piece, while a knight's movement can not.

Input

There will be multiple test cases. Each test case will consist of 4 lines. The first line will contain two integers n and m, indicating the dimensions of the board, giving rows and columns, respectively. Neither integer will exceed 1000. The next three lines will each be of the form

$k\ r_1\ c_1\ r_2\ c_2 \cdots r_k\ c_k$

indicating the location of the queens, knights and pawns, respectively. The numbering of the rows and columns will start at one. There will be no more than 100 of any one piece. Values of n = m = 0 indicate end of input.

Output

Each test case should generate one line of the form

Board b has s safe squares.

where *b* is the number of the board (starting at one) and you supply the correct value for *s*.

Example

```
4 4
2 1 4 2 4
1 1 2
1 2 3
2 3
1 1 2
1 1 1
0
1000 1000
1 3 3
0
0
0 0
```

Output:

```
Board 1 has 6 safe squares.
Board 2 has 0 safe squares.
Board 3 has 996998 safe squares.
```