Practice Problem: Sum

Source file: `X.c, X.cpp, X.pas`

Introduction

Adding many numbers is a very boring task. Your task is to write a program that can do this automatically.

Input

The input contains several blocks of test cases. Each case consists of a line containing two integers $1 \leq a \leq b \leq 1000$. The input is terminated by a test case with $a = b = 0$.

Output

For each test case, you have to output a line containing a single number, the sum of the integers between a and b (including a and b).

Sample Input

```
1 5
10 10
0 0
```

Sample Output

```
15
10
```

Problem A: The Road

Source file: `A.c, A.cpp, A.pas`

Introduction

During the Grear Civil War of Ruritania, General Bandu, the leader of the 3rd Infantry Division of the Bandulu army had to move his soldiers from the city of Bulu to Ndulu. However, Colonel Tuluu, the leader of the 8th Parachute Battalion of the Tuluvu army, has set up a lookout spot near the road. Fortunately, there are buildings, hills, forests etc. beside the road, they block some parts of the road. Therefore there are safe sections of the road (that cannot be seen from the lookout) and dangerous sections of the road (that can be seen from the lookout). Your task is to count the number of safe sections on the road.

The road is a straight line segment, the buildings etc. are represented by polygons. The road on the figure has two safe segments, marked by bold lines. Note that the second section is interrupted by an infinitesimally small dangerous section, but these 'dangerous points' are not counted.

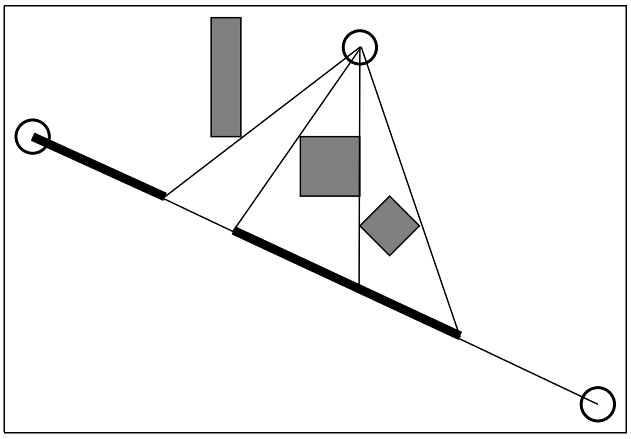
Input

The input contains several blocks of test cases. Each case begins with a line containing an integer $1 \leq n \leq 1000$, the number of polygons. The next line contains two integers, the x and y coordinates of the Tuluvu lookout spot. The next line contains the coordinates of the cities Bulu and Ndulu. This is followed by the descriptions of the n polygons. Each polygon is described as follows. The first line contains an integer $3 \leq m \leq 50$, the number of vertices of the polygon. The next m lines contain the coordinates of the m vertices in clockwise or counter-clockwise direction. It can be assumed that the polygons do not touch each other, the road, or the lookout spot. The polygons can be either convex or concave, but a polygon does not intersect itself. The lookout spot and the two cities are not on the same line. The coordinates in the input are integers in the range -20000 to 20000. The total number of points of the polygons in a test case is at most 20000.

The input is terminated by a block with $n = 0$.

Output

For each test case, you have to output a line containing a single integer, the number of safe sections on the road.



Sample Input

```
4
10 0
1 2 1 28
3
20 21
21 20
21 21
3
-10 -10
-10 -11
-11 -10
4
2 2
2 3
3 3
3 2
4
2 10
2 11
3 11
3 10
3
10 0
0 -5 0 12
4
2 0
3 -1
2 -2
1 -1
4
2 0
3 1
2 2
1 1
4
9 -1
11 -1
11 -3
9 -3
0
```

Sample Output

```
2
1
```

Problem B: Cable Master

Source file: `B.c, B.cpp, B.pas`

Introduction

Inhabitants of the Wonderland have decided to hold a regional programming contest. The Judging Committee has volunteered and has promised to organize the most honest contest ever. It was decided to connect computers for the contestants using a “star” topology—i.e., connect them all to a single central hub. To organize a truly honest contest, the Head of the Judging Committee has decreed to place all contestants evenly around the hub on an equal distance from it.

To buy network cables, the Judging Committee has contacted a local network solutions provider with a request to sell for them a specified number of cables with equal lengths. The Judging Committee wants the cables to be as long as possible to sit contestants as far from each other as possible.

The Cable Master of the company was assigned to the task. He knows the length of each cable in the stock up to a centimeter, and he can cut them with a centimeter precision being told the length of the pieces he must cut. However, this time, the length is not known and the Cable Master is completely puzzled.

You are to help the Cable Master, by writing a program that will determine the maximal possible length of a cable piece that can be cut from the cables in the stock, to get the specified number of pieces.

Input

The input file contains several blocks of test cases. The first line of each test case contains two integer numbers N and K, separated by a space. N ($1 \le N \le 10000$) is the number of cables in the stock, and K ($1 \le K \le 10000$) is the number of requested pieces. The first line is followed by N lines with one number per line, that specify the length of each cable in the stock in meters. All cables are at least 1 meter and at most 100 kilometers in length. All lengths in the input file are written with a centimeter precision, with exactly two digits after a decimal point.

The input is terminated by a block with $K = N = 0$.

Output

For each test case, write to the output file the maximal length (in meters) of the pieces that Cable Master may cut from the cables in the stock to get the requested number of pieces. The number must be written with a centimeter precision, with exactly two digits after a decimal point.

If it is not possible to cut the requested number of pieces each one being at least one centimeter long, then the output file must contain the single number `'0.00'` (without quotes).

Sample Input

```
4 11
8.02
7.43
4.57
5.39
0 0
```

Sample Output

```
2.00
```

Problem C: Necklace

Source file: `C.c, C.cpp, C.pas`

Introduction

In a long and bloody fight, the great hero Megalinuxiad finally defeated the evil dragon Xlup. Hidden deep in the cave of the dragon, Megalinuxiad found an old chest covered in cobweb. After disarming the deadly traps protecting the chest, he found a very long and beautiful necklace containing black and white pearls. Pearls are very valuable in the tribe of Megalinuxiad, since they are used for all sorts magical ceremonies.

Unfortunately, there is a very old law that a great hero cannot posses a pearl necklace, therefore he has to offer the pearls to the leaders of the tribe. According to the law, first he has to give 3 white pearls and 2 black pearls to the Chieftain. The remaining white pearls have to be divided evenly among the Four Chief Shamans in the tribe. The remaining black pearls has to be divided evenly among the Five Taxmasters. For example, if he has a necklace with 11 white and 17 black pearls, then the Chieftain gets 3 white and 2 black, each shaman gets 2 white, and each taxmaster gets 3 black pearls. It is extremely important that every shaman and every taxmaster gets exactly the same number of pearls. If a shaman gets fewer pearls than some other shaman, then he will get angry at Megalinuxiad, and turns it into a toad. Even worse things can happen if a taxmaster becomes angry. For example, if the necklace contains 10 white pearls, then one of the shamans will be angry. Similarly, if there are 18 black pearls, then the 16 remaining pearls cannot be divided evenly among the taxmasters. If there are only one black pearl, or there are no black pearls, then the Chieftain cannot get his share, which also very bad.

Megalinuxiad wants to avoid these complications, therefore he decides to throw away part of the necklace such that the remaining pearls can be divided without problems. Obviously, he wants to throw away as few pearls as possible. However, he can cut off only the two ends of the necklace, he cannot remove a pearls from the middle. For example, the necklace

●●○○○○●○○○○

is not good, but after cutting a black pearl from the left and a white pearl from the right the pearls can be shared: the Chieftain gets 3 white and 2 black pearls, each Shaman gets 1 white pearl. The taxmasters don't get anything, but this is acceptable, since all of them get the same number (zero) of perls. The necklace

●○○●

is not good either, but it cannot be corrected, since there are not enough white pearls for the chieftain. Therefore all the pearls have to be thrown away.

Input

The input file begins with a line containing a single integer, the number of test cases to follow. Each test case is on separate line, describing a necklace. The line contains as many characters as the number of pearls in the necklace, each character is either `'W'` (white) or `'B'` (black). The length of each line is at most 100000 characters.

Output

For each test case you have to output a line containing a single integer, the number of pearls that have to be cut off from the necklace. If the necklace cannot be corrected, then output the length of the necklace (indicating that all the pearls have to be thrown away).

Sample Input

```
3
BBWWWWBWWWW
BWWB
BBWWWWBBBBBWW
```

Sample Output

```
2
4
8
```

Problem D: Billboard

Source file: `D.c, D.cpp, D.pas`

Introduction

The Department of Recreation has decided that it must be more profitable, and it wants to sell advertising space along a popular jogging path at a local park. They have built a number of billboards (special signs for advertisements) along the path and have decided to sell advertising space on these billboards. Billboards are situated evenly along the jogging path, and they are given consecutive integer numbers corresponding to their order along the path. At most one advertisement can be placed on each billboard.

A particular client wishes to purchase advertising space on these billboards but needs guarantees that every jogger will see it's advertisement at least K times while running along the path. However, different joggers run along different parts of the path.

Interviews with joggers revealed that each of them has chosen a section of the path which he/she likes to run along every day. Since advertisers care only about billboards seen by joggers, each jogger's personal path can be identified by the sequence of billboards viewed during a run. Taking into account that billboards are numbered consecutively, it is sufficient to record the first and the last billboard numbers seen by each jogger.

Unfortunately, interviews with joggers also showed that some joggers don't run far enough to see K billboards. Some of them are in such bad shape that they get to see only one billboard (here, the first and last billboard numbers for their path will be identical). Since out-of-shape joggers won't get to see K billboards, the client requires that they see an advertisement on every billboard along their section of the path. Although this is not as good as them seeing K advertisements, this is the best that can be done and it's enough to satisfy the client.

In order to reduce advertising costs, the client hires you to figure out how to minimize the number of billboards they need to pay for and, at the same time, satisfy stated requirements.

Input

The input file contains several blocks of test cases. The first line of each case contains two integers K and N ($1 \leq K, N \leq 1000$) separated by a space. K is the minimal number of advertisements that every jogger must see, and N is the total number of joggers.

The following N lines describe the path of each jogger. Each line contains two integers A_i and B_i (both numbers are not greater than 10000 by absolute value). A_i represents the first billboard number seen by jogger number i and B_i gives the last billboard number seen by that jogger. During a run, jogger i will see billboards A_i, B_i and all billboards between them.

The input is terminated by a block with $K = N = 0$.

Output

For each test case, write a single integer M on a separate line. This number gives the minimal number of advertisements that should be placed on billboards in order to fulfill the client's requirements.

Sample Input

```
5 10
1 10
20 27
0 -3
15 15
8 2
7 30
-1 -10
27 20
2 9
14 21
0 0
```

Sample Output

```
19
```

Problem E: Tennis

Source file: `E.c, E.cpp, E.pas`

Introduction

A very sad event shocked the Kingdom of Yrrgarr: a popular tennis player committed suicide because he had not won a single match during his long carrier. The King of Yrrgarr does not want this to happen again, therefore he orders that next year every tennis player will win at least one match (a king can order that!)

The king gives you the list of the matches scheduled for next year. Your job is to decide in advance for each match who will win the game. You have to make the decisions in such a way that every player will win at least one match. Deciding the results of the games in advance makes the tennis competitions a bit boring, but at least no one will commit suicide.

Input

The input contains several blocks of test cases. Each case begins with a line containing two integers $1 \le n \le 10000$ and $1 \le m \le 100000$: the number n of tennis players and the number m of tennis matches scheduled for next year. This line is followed by m lines describing the m matches. Each such line contains two integers $1 \le i, j \le n$, which has the meaning that players i and j will play a match. It can be assumed that every player will play at least one match.

The input is terminated by a block with $n = m = 0$.

Output

For each test case, you have to select a winner for each match (there is no draw) such that every player wins at least one match. If this is possible, then output a single line containing `'Suicides can be avoided.'`, otherwise write `'Suicide is possible.'` (without the quotes).

Sample Input

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

Sample Output

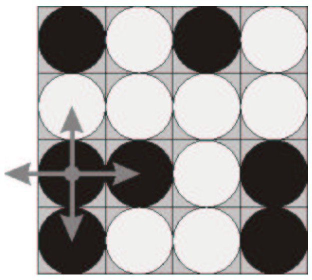
```
Suicide is possible.
Suicides can be avoided.
Suicides can be avoided.
```

Problem F: Flip Game

Source file: `F.c, F.cpp, F.pas`

Introduction

Flip game is played on a rectangular 4 × 4 field with two-sided pieces placed on each of its 16 squares. One side of each piece is white and the other one is black and each piece is lying either it's black or white side up. Each round you flip 3 to 5 pieces, thus changing the color of their upper side from black to white and vice versa. The pieces to be flipped are chosen every round according to the following rules:



- Choose any one of the 16 pieces.
- Flip the chosen piece and also all adjacent pieces to the left, to the right, to the top, and to the bottom of the chosen piece (if there are any).

Consider the following position as an example:

●○●○
○○○○
●●○●
●○○●

Here ● denotes pieces lying their black side up and ○ denotes pieces lying their white side up. If we choose to flip the 1st piece from the 3rd row (this choice is shown at the picture), then the field will become:

●○●○
●○○○
○○○●
○○○●

The goal of the game is to flip either all pieces white side up or all pieces black side up. You are to write a program that will search for the minimum number of rounds needed to achieve this goal.

Input

The input file contains several test cases. The first line contains a single integer, the number of test cases in the input. Each test case consists of 4 lines with 4 characters `'w'` (white side up) or `'b'` (black side up) each that denote a game field position. There are *no* empty lines separating the test cases.

Output

For each test case, you have to write to the output a single integer number: the minimum number of rounds needed to achieve the goal of the game from the given position. If the goal is initially achieved, then write 0. If it's impossible to achieve the goal, then write the word `'Impossible'` (without quotes).

Sample Input

```
2
bwbw
wwww
bbwb
bwwb
bwwb
bbwb
bwwb
bwww
```

Sample Output

```
Impossible
4
```

Problem G: Joseph

Source file: `G.c, G.cpp, G.pas`

Introduction

The Joseph's problem is notoriously known. For those who are not familiar with the original problem: from among n people, numbered $1, 2, \ldots, n$, standing in circle every mth is going to be executed and only the life of the last remaining person will be saved. Joseph was smart enough to choose the position of the last remaining person, thus saving his life to give us the message about the incident. For example when $n = 6$ and $m = 5$ then the people will be executed in the order $5, 4, 6, 2, 3$ and 1 will be saved.

Suppose that there are k good guys and k bad guys. In the circle the first k are good guys and the last k bad guys. You have to determine such minimal m that all the bad guys will be executed before the first good guy.

Input

The input file consists of separate lines containing k. The last line in the input file contains 0. You can suppose that $0 < k < 14$.

Output

The output file will consist of separate lines containing m corresponding to k in the input file.

Sample Input

```
3
4
0
```

Sample Output

```
5
30
```