Problem A. A Coin Game

Input file:	`coingame.in`
Output file:	`coingame.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Consider n heaps of 1-cent coins, arranged in a row.

Two players make turns. During each turn, a player can takes some of the heaps from the left side of the row, at least one and at most the number of heaps taken by his competitor during the previous turn (during the first turn, the first player can take at most k heaps).

The game is over when all heaps are taken. What will be the number of coins that the first player gets if both players maximize the number of coins they get?

Input

The first and only line of the input file contains the number n of heaps, $1 \leq n \leq 180$, followed by n numbers denoting the number of coins in each heap from left to right, followed by k, $1 \leq k \leq 80$.

Number of coins in each heap is at least 1 and at most 20000.

Output

Output one integer — the number of coins that the first player will get with optimal play.

Sample input and output

coingame.in	coingame.out
3 4 9 1 3	14
4 1 2 2 7 3	5
5 3 4 8 1 7 2	18

Problem B. Die

Input file:	`die.in`
Output file:	`die.out`
Time limit:	7 seconds
Memory limit:	256 megabytes

A cube with sides marked from 1 to 6 is rolled n times. Find the probability that the sum of the numbers that appeared on the top side of each roll is equal to q.

Input

The first and only line of the input file contains two integers n and q, $1 \le n \le 500$, $1 \le q \le 3000$.

Output

Output one floating-point number. Your answer will be considered correct when it is within 1% of the correct one.

Please note that when the correct answer is very small, as in the last example, the allowed margin of error is also very small.

Sample input and output

die.in	die.out
1 6	0.16666
1 7	0
4 14	0.11265
100 100	1.53e-78

Problem C. Fences

Input file:	`fences.in`
Output file:	`fences.out`
Time limit:	3 seconds
Memory limit:	64 megabytes

Each of Farmer John's cows likes has a favorite grazing location on FJ's land. These grazing locations vary a lot because some cows like being at the center of attention while others enjoy solitude near the edge of the wilderness. A gracious farmer, FJ tries to accomodate his cows by maintaining a pasture that contains all the grazing locations. Traditionally, FJ has built a single fence around all the grazing locations so the cows can graze merrily while still feeling secure. FJ knows that it is important that a cow be able to see the entire pasture no matter where she is standing. In particular, no three cows ever stand in a line because the middle cow would be preventing the outer two from seeing each other. And obviously, no two cows can occpuy the same location (they're big creatures, you know). Maintaining the pasture is hard work, so eventually FJ realized that he could minimize the area of the pasture by finding the smallest convex polygon enclosing all the cows.

One day while visiting his cows, FJ realized that the cows liked to graze in two groups. One group liked the shady foot of a small hill and the other group liked the sunny top. FJ suddenly had a revelation: he could build two separate enclosing fences for the two groups. Of course, both fences must be convex and non-overlapping. This saves him work because he has less pasture area to maintain, and the extra fence he needs to build doesn't cost too much.

The cows decided to change their grazing positions, so it is no longer clear to FJ where to build the two fences, so of course he seeks your help. He has written down the new grazing positions of the cows. It is your task to determine the minimum total pasture area of two fences that encloses all the cows.

Input

There'll be several test cases, each representing a possible set of grazing positions.

The first line of each test case will contain a positive integer N ($2 \leq N \leq 150$), the number of cows. The subsequent N lines will contain two integers X, Y, representing the grazing location (X, Y) of a cow.

The input data is terminated by a line that contains one zero, and should not be processed.

Output

The output should contain the minimum area. Formatting should be as in the following sample output, in particular, there should be exactly one digit after the decimal point.

Sample input and output

fences.in	fences.out
6 0 0 0 1 1 0 5 5 5 4 4 5 0	Farm 1: 1.0

Problem D. Game

Input file:	`game.in`
Output file:	`game.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

A group of N MIT students decide to play the following intelligent game. They first choose three numbers A, B, and P, and then they gather together in a circle. Each MIT student wears a t-shirt with a distinct number from 1 to N on it. The students arrange themselves in a circle so that the numbers on their t-shirts are in consecutive order.

The game has N count-and-eliminate rounds, starting with round 1. In round i, the $(1 + ((Ai + B) \bmod P))$-th student is eliminated from the circle, with the counting starting from the student with the smallest number on his/her t-shirt.

The last student to be eliminated from the circle is the winner of the game.

Input

There will be several test cases, each describing an instance of a game. Each test case has a single line containing four numbers, N, A, B, and P ($1 \le N, P \le 5000$, $0 \le A, B \le 5000$). The input data is terminated by a line containing four zeroes, and should not be processed.

Output

For each test case, output the winner of the game, as in the sample output.

Sample input and output

game.in	game.out
3 1 2 5 4 3 5 2 0 0 0 0	The winner is 3. The winner is 4.

Problem E. House

Input file:	`house.in`
Output file:	`house.out`
Time limit:	3 seconds
Memory limit:	64 megabytes

In a far away land, there is a blue house with many blue rooms. The blue rooms in the house are connected by yellow corridors, such that any blue room is accessible from any other blue room through a sequence of yellow corridors.

Interesting enough, some corridors are more important than others. More precisely, some corridors are so important that if they are blocked, some rooms become unaccessible from other rooms.

The owner of the house wants to take more care of these corridors, and asks you to change their color from yellow to red.

Input

There'll be several test cases, each describing a house. The first line in each test case contains two positive integers N and M, where N ($1 \leq N \leq 50000$) is the number of rooms, and M ($1 \leq M \leq 100000$) is the number of corridors in the house. Each of the subsequent M lines contains two integers A and B, representing a corridor between rooms A and B. Note that all corridors are bidirectional. The last line of each test case contains a single dash.

The input data is terminated by a line that contains two zeroes, and should not be processed.

Output

For each test case, output $K+1$ lines, where K represents the number of corridors that need to be colored in red. Each of the first K lines should contain two integers A and B, describing one of the corridors that should be colored in red. The last line in each test case should contain two zeroes.

Sample input and output

house.in	house.out
5 5 1 2 1 3 2 3 2 4 3 5 - 0 0	3 5 2 4 0 0

Problem F. Island

Input file:	`island.in`
Output file:	`island.out`
Time limit:	3 seconds
Memory limit:	64 megabytes

In a secluded region of the Pacific Ocean, there is a small thriving island. The people on this island live in one of many squares, which is a dense area of shops, eateries, and residences. The squares on the island are connected via two-way roads. Being frugal people, they have ensured that no road has been built between two squares if it had been possible to travel between the two squares without the road. Some would say that they are overly frugal, and have failed to join all squares with each other. It might be still impossible to travel via roads from some square to another.

The Island Transportation Council is still in expanding the network of roads, but the Council wishes to expand in an orderly fashion. In particular, they want to make sure that no one has to travel too far to get from one square to another. Also, they want to make sure that no square is connected to too many roads, or else the intersection would be quite prone to accidents.

At the current moment, they only wish to survey the state of the the roads. In particular, they wish to find out R, the maximum number of roads any person must travel on to get from one square to another. Also, they want to find out S , the maximum number of roads joined at any square.

Input

There will be several test cases, each representing an island.

The first line of each test case will contain two positive integer numbers N, M , where N ($1 \leq N \leq 100000$) is the number of squares and M is the number of roads on the island. The squares are numbered from 1 to N. Each of the subsequent M lines contains two distinct integer numbers A, B, which represent a road connecting square A with square B.

The input data is terminated by a line that contains two zeros, and should not be processed.

Output

Output a single line for each test case. The output should contain the value R and any two squares that are separated by R roads. Also, output S and the square that has S roads connected to it. Formatting should be as in the following sample output.

Sample input and output

island.in
5 4 1 2 2 3 2 4 4 5 0 0
island.out
Island 1: 3 roads from 5 to 1; square 2 has 3 roads

Problem G. Matrix

Input file:	`matrix.in`
Output file:	`matrix.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given an $n \times n$ matrix of integers. A *path* of length k in this matrix is a sequence of cells (i_1, j_1), (i_2, j_2), ..., (i_k, j_k) such that:

- Any two consecutive cells in the sequence are distinct but adjacent. More formally, $|i_s - i_{s+1}| + |j_s - j_{s+1}| = 1$.
- The first cell is $(1, 1)$: $i_1 = j_1 = 1$.

Find the path of length k with the largest sum of numbers written in the cells of this path (if a cell appears several times in the path, it contributes to the sum as many times).

Input

The first line of the input file contains two integers n and k, $2 \le n \le 100$, $1 \le k \le 10000$.

The next n lines describe the matrix. All numbers in the matrix are positive and don't exceed 10000.

Output

Output one integer — the largest possible sum.

Sample input and output

matrix.in	matrix.out
5 7 1 1 1 1 1 1 1 3 1 9 1 1 6 1 1 1 1 3 1 1 1 1 1 1 1	21

Problem H. Tetris

Input file:	`tetris.in`
Output file:	`tetris.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

In a simplified game of Tetris, all pieces are rectangles of size $N \times M$, where N is the width and M is the height of the piece. The size of the board is also rectangular in shape, and has size $A \times B$. At each step in the game, pieces come from above the board and fall down until they touch another piece or the bottom of the board. If an entire row on the board is covered by pieces after a piece is placed in its final position, then that row disappears. If a piece cannot fit entirely on the board, the game is over.

You are asked to simulate a game of Tetris in which the user takes no action.

Input

There will be several test cases, each describing a Tetris game. Each test case starts with a line that contains two integers, A and B, representing the width and the height of the board, $1 \leq A, B \leq 50$. The next lines contain four integers each describing the width, the height, the color, and the column of the board above which the left corner of the piece is placed. Colors are coded with digits from 1 to 9. The last line in each test case contains the sequence "0 0 0 0"

The input data is terminated by a line that contains two zeroes, and should not be processed.

Output

For each test case, output a matrix of $A \times B$ elements representing the status of the board at the end of the game. Each element of this matrix should contain the current color of the piece at that position, or 0 if there is no piece there.

If the game was over before the last piece was placed on the board, the output should contain a single line containing the words "GAME OVER".

Write a dash after the output of each test case.

Sample input and output

tetris.in	tetris.out
5 6 2 2 1 1 3 3 2 3 3 2 3 2 1 1 4 1 0 0 0 0 2 3 1 2 1 1 1 2 2 1 0 0 0 0 0 0	00000 00000 00000 03330 03330 40222 - GAME OVER -

Problem I. Words

Input file:	`words.in`
Output file:	`words.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Farmer John recently heard that his life-long nemesis, Ranger George had successfully trained his ten thousand monkeys to type out the complete works of Shakespeare. FJ, enraged with envy, immediately brought this to the attention of Bessie, his favorite cow, and asked her to get the herd to do the same. But it is common knowledge that cows are fat and docile, big and dumb. Even given infinite time, it is unlikely that they would even be able to produce an intelligible sentence, let alone Shakespeare.

The cunning FJ decided to cheat. If the cows would only stamp out a sequence of letters in the pasture, then he could argue that the cows had "produced" every word that is a subsequence (not necessarily contiguous) of these letters. FJ realized the versatility of his brilliant plan. Without too much effort or intelligence, the cows could not only produce every English word, but also all German and French words (minus the appropriate accents on the words, but FJ isn't too particular about that). Then FJ would be able to boast about his cows having produced the entire works of Goethe and Hugo.

In short, he wants to make sure that every possible word (up to some length and using some letters drawn from some alphabet) is somewhere in the sequence. The alphabet will simply be the set of letters that appear in the sequence. But FJ ain't got all day to be hangin' out in the pasture and make sure that all the words are there, so he copied down the sequence that the cows produced and handed it to you.

Input

There will be several lines, each representing a sequence of letters 'a'..'z' that the cows produced. The total size of the input file doesn't exceed 60000 bytes.

Output

For each sequence, output the longest length such that all words with this at most this length are subsequences in the sequence. Formatting should be as in the following sample output.

Sample input and output

words.in	words.out
aaa baaba	String 1: 3 String 2: 2