Problem A

Miles 2 Km

Source file name: `miles2km.c`, `miles2km.cpp` *or* `miles2km.java`

It is a well known fact that you can use Fibonacci numbers to convert distances from miles 2 kilometers. Well, it's actually not a well known fact. It is, however, a well known fact that one mile is 1.6 km. Now let me explain how you can use Fibonacci numbers in order to convert (approximately) miles to kilometers. Lets first write down the first numbers of the Fibonacci sequence:

$$0,\ 1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34,\ 55,\ 89,\ 144,\ \ldots$$

So let's suppose you want to convert 5 miles into kms. Well, in the Fibonacci sequence the number following 5 is 8 so 5 miles is 8 kms. You don't believe me? ok, just do the maths: multiply 5 by 1.6... $5 \times 1.6 = 8$! Amazing right?. The bad thing is that it doesn't work all the time, for example, how do you convert 89 miles into kms?. According to the previous procedure it would be 144 km, but it is really $89 \times 1.6 = 142.4$, so there is an error of 1.6. Still amazing no?. Anyway, what would you do if you want to convert 6 miles into kilometers?. You could, for example, say $6 = 5 + 1$, so we convert 5 and 1 to kms and then we add up. 5 miles is 8 kms and 1 miles is 2 kms, adding up we obtain $8 + 2 = 10$ so 6 miles is approximately 10 kms. Of course, we could also say $6 = 2 + 2 + 2$, so we convert 2 miles to kms and obtain 3. So 6 miles is approximately $3 + 3 + 3 = 9$ kms. Which answer is closer to the real value?.

It is your task to write a program which finds the best conversion from miles to kilometers using the previous Fibonacci method. The best conversion is the one with the lowest error.

Input

The input will consist of several test cases. Each case will be a single line containing one integer number $0 < N \leq 1000$. The end of input is indicated by a test case with $N = 0$.

The input must be read from standard input.

Output

For each test case in the input, your program must print the lower error e for converting the corresponding input to kilometers (rounding up to 2 decimal digits).

The output must be written to standard output.

Sample input	**Output for the sample input**
`6`	`0.40`
`5`	`0.00`
`12`	`0.20`
`0`	

Problem B
Books

Source file name: `books.c`, `books.cpp` *or* `books.java`

Suppose you are a professor who just moved to a new house. All your stuff is packed in well-identified boxes, and a lot of them contain books of every kind and size. You just have chosen a room for your new library and you have unpacked all your books, but you are still missing the bookshelves.

Now suppose you went to the hardware store and that they couldn't offer you a full pre-built personal library, but only a series of identical rectangular bookshelves that can be stacked one over another, or placed next to each other. You have decided to go for this option, but you don't know how many rectangular bookshelves are you going to need. The problem lies in all your personal rules to accommodate books in shelves. First, you don't like to waste space or resources, so you want each bookshelf filled as much as possible, but using as few books as possible. Second, you don't like to stack books over each other or occlude a book by putting another book in front of it. Third, you like your books placed vertically, in straight position, not with the front-cover facing the bottom or the top of the shelf, but facing either the left or the right side of the shelf. These are some of the things you need in your orderly life.

Now, you want to know how to organize your books in the shelves, and since you don't like to put books in front of others, you have decided to ignore the depth of the books and the shelves. Consequently, you have actually annotated the height and width of all your books. Given the height and width of an empty bookshelf, you can actually compute how much area of the shelf would each of your books take. This way you can also compute how much area of the shelf is being wasted when no more books fit in it.

Then, given the number of bookshelves you are going to buy along with their height and width, you have decided to write a program that, for a given number of bookshelves, will compute the best way to organize your books in the shelves according to your personal rules, and report the total amount of space you are wasting among all shelves. Your plan is to input to this program several different number of shelves, with different sizes, and then decide what's the best configuration for you to buy in the hardware store.

Input

The input can contain several problems. Each problem starts with four integers N, H, W, B, separated of each other by a blank space. $0 < N \leq 10$ represents the number of bookshelves you are going to buy. $0 < H, W \leq 30$ are the height and width of each bookshelf, respectively. $0 < B \leq 100$ represents the number of books you own. Then, B lines follow. Each of these lines contain two integers $0 < Bh, Bw \leq 30$ separated of each other by a blank space, representing the height and width of each book, respectively. Assume you measured your books while they were oriented vertically. This is how you want them to be placed in the shelves!. Their orientation must not be changed!.

The end of input is represented by a case with $N = 0, H = 0, W = 0$, and $B = 0$.

The input must be read from standard input.

Output

For each problem in the input, the output should be a single line containing an integer representing the total area of wasted space among all the bookshelves specified in the input problem.

The output must be written to standard output.

Sample input

```
5 5 4 2
4 6
5 4
1 10 10 3
10 10
10 10
10 10
3 10 10 3
10 10
10 10
10 11
0 0 0 0
```

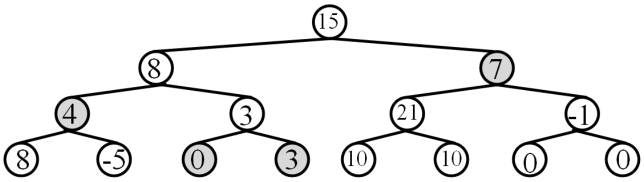
Output for the sample input

```
80
0
100
```

Problem C
Optimal Cut

Source file name: `optimal.c`, `optimal.cpp` *or* `optimal.java`

A *cut* of a binary tree is a set of tree nodes such as for each possible path from the root node to any leaf node, just one node in the cut belongs to the path. In a weighted binary tree, one can compute a *weighted* cut, because it would include weighted nodes. The total weight of such cut can be computed by adding up the weights of the nodes in it. The figure below shows a weighted binary tree. The gray nodes represent a weighted cut, and its total weight is 14.



Now, given a weighted binary tree, you have to write a program that finds the weighted cut with the maximal total weight value among all possible weighted cuts in the tree. This is hereafter called the *Optimal Cut* of the input binary tree. However, to make it a bit more interesting, your program must find the optimal cut that includes no more than K nodes, and report its weight. In the figure above, for instance, the nodes of the optimal cut sums 28 when $K = 3$, and 15 when $K = 2$.

Input

The input can contain several problems. Each problem contains the description of a complete binary tree in a single line. This description corresponds to sequence of integer numbers, separated of each other by a blank space. The description starts with an integer $0 \leq H < 20$ representing the height of the tree, followed by another integer $1 \leq K \leq 20$ representing the maximum number of nodes in the optimal cut to be found. Then, the weights $-10^3 \leq Wi \leq 10^3$ of the nodes follow, listed in pre-order.

The end of input is indicated by a single line containing only an integer value of -1.

The input must be read from standard input.

Output

For each problem in the input, the output should be a single line with a single integer number, representing the total weight of the optimal cut found.

The output must be written to standard output.

Sample input	**Output for the sample input**

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

Problem D

Nails

Source file name: `nails.c`, `nails.cpp` *or* `nails.java`

The World Of Nails (WON) is a famous hardware store specialized in selling hundreds of different kinds of nails. Some of them very old, rare and expensive. However, the store also offers some standard hand tools and wood. Rado, the store's manager, has two kids who usually go to the store after school and spend the afternoon playing with the tools. During all this time, the kids play with hammers, nails, screws, wood, etc. (quite dangerous isn't it?).

One day, the kids were playing in the wood deposit. By accident, they hit a huge pile of wood sticks which then collapsed. A lot of wood sticks fell to the floor, but did not hit any of the kids. The kids did not get scare and so they continued playing. They took a bag of very expensive nails and a hammer and decided to hammer the nails on the wood sticks, just for fun. If a wood stick were laying in top of another, they joined these two pieces together by hammering exactly one nail where they crossed, see Figure 1. No more than two stick are crossed in the same point such as shown on third configuration in figure below. If a wood stick was laying on the floor all alone, with no other stick on top of it, they hammered the stick on to the floor using two nails, one for each end of the stick.

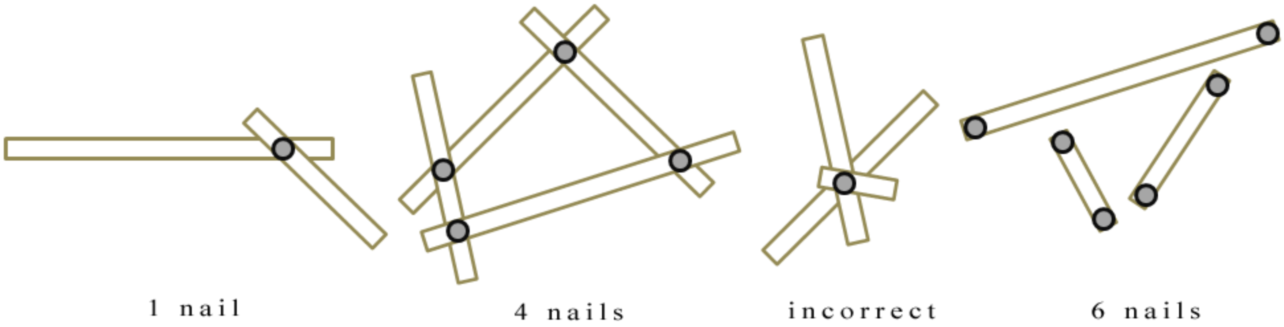


Figure 1: Different configurations

After a couple of hours, Rado discovered the disaster in the deposit. When he noticed what kind of nails were the kids using, he was shocked. A lot of money wasted on a little game. Rado wants you to help him calculate how many nails did they kids use, to calculate how much money he has lost.

Input

The input consists in several test cases. Each case corresponds to a set of wooden sticks. The first line of a test case contains one integer ($0 \leq N \leq 1000$), indicating the number of sticks. Next, N lines follow, each one corresponding to a stick. Each line contains four integer values ($0 \leq X_1, Y_1, X_2, Y_2 \leq 1000$) separated of each other by exactly one blank space. The first two integers correspond to the Cartesian coordinates of one end of the wood stick. The later two integers correspond to the Cartesian coordinate of the other end of the wood stick. Assume no stick is completely superposed by another.

The end of input is indicated by a test case with $N = 0$.

The input must be read from standard input.

Output

For each case specified in the input, the output is a single line containing the number of nails used by the kids.

The output must be written to standard output.

Sample input

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

Output for the sample input

```
3
2
6
```

Problem E

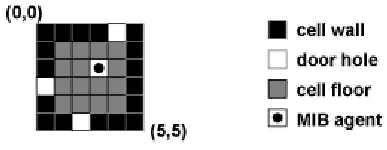
Escape

Source file name: `escape.c`, `escape.cpp` *or* `escape.java`

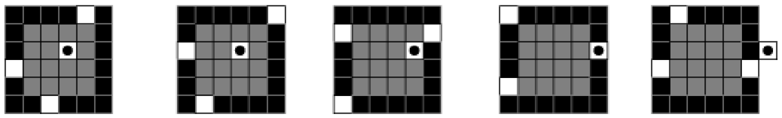
Suppose you are a MIB (Men in Black) agent who has been just abducted by aliens. They have locked you in a very strange prison cell. The cell is completely rectangular and it seems to have no roof. You are located somewhere inside the cell, and for some reason you cannot walk around the cell, you are simply stuck at that position. A very strange detail is that the walls are moving constantly around you, in a periodic fashion, at a regular speed. Even stranger is that the walls have actually door holes which you could use to get out of the cell, but the holes are moving constantly with the walls and, anyway, you cannot move from where you are. It's all part of some bizarre alien torturing method.

Your fellow MIB agents are aware of your adduction and are trying to help you, using a secret communication device implanted in your skin. They have informed you that you cannot move because the floor has some kind of gravity control mechanism to hold you still. Your fellow MIB agents can disable the device remotely just for a little time, so you have only one chance to escape. In addition, they can only disable the device to allow you to move either horizontally or vertically. Once the device is disabled, you would have to run to the door and try to exit the cell through a door hole. Therefore, you have to calculate the exact moment a hole will pass in front of you. Once you start moving you cannot change your direction, or it might be too late to reach the door hole.

Since you only have one chance to escape, you have decided to run a simulation first. To simulate this process you are assuming the cell looks like a matrix when seem from the sky, and each matrix position could only be: part of the cell wall, part of the cell floor, a door hole or yourself. You are assuming the walls begin the simulation stalled, and then start moving clockwise, at one matrix position per second. You also move at one matrix position per second. The following image depicts a sample matrix of your simulation and the whole escaping process.



sample 6x6 matrix



0 seconds
Walls stalled

1 second
Walls starts moving

2 seconds
Agent starts moving towards (2,5)

3 seconds

4 seconds
Agent exits cell

Given the size of the cell, the position of the door holes and your position inside the cell, your task is to write a program that simulates the situation and reports the minimum amount of time that would take you to exit the cell, counted from the moment the simulation starts, and the distance you have to run during your escape. This distance is represented by the number of matrix positions you have to pass through to reach the door hole, plus the one last step you need to actually exit the cell.

If you find two or more possible solutions with the same timing, then choose the solution with the lowest running distance.

Input

The input contains several test cases, each one corresponding to one simulation. The first line of a test case contains three integers M, N and H. The first two integers indicate the size of the cell ($3 \leq M, N \leq 1000$), corresponding to the number of rows and columns of the cell, respectively. The last integer indicates the number of door holes ($0 < H < (2 \times M) + (2 \times N) - 4$). Then, H lines follow, each one containing two integer values, corresponding to the matrix position of a door hole. These two values will be separated from each other by exactly one blank space. All door-hole-positions will be different from each other. Finally, there will be a single line with two integers, separated by a blank space, indicating your initial matrix position inside the cell.

The end of input is indicated by a test case with $M = 0$, $N = 0$ and $H = 0$.

The input must be read from standard input.

Output

For each test case specified in the input, your program must print a single line containing two integers, separated from each other by one blank space. The first integer should be the minimum amount of time that would take you to exit the cell, counted from the moment the simulation starts. The other integer should be the distance you have to run during your escape.

The output must be written to standard output.

Sample input	**Output for the sample input**

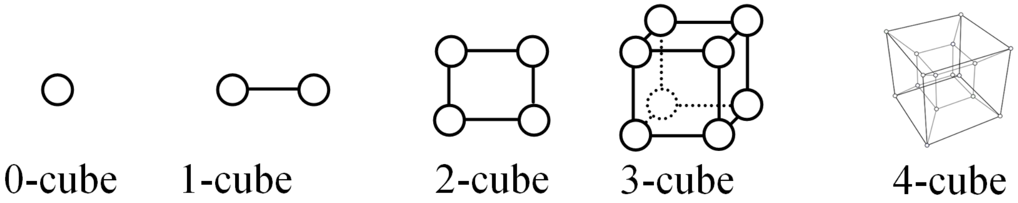
6 6 3
0 4
3 0
5 2
2 3
3 3 4
0 0
0 2
2 0
2 2
1 1
0 0 0
	4 3
	2 2

Problem F

Hypercube

Source file name: `hypercube.c`, `hypercube.cpp` *or* `hypercube.java`

In geometry, a hypercube is an $n-$dimensional analogue of a square ($n = 2$) and a cube ($n = 3$). It consists in groups of opposite parallel line segments aligned in each of the space's dimensions, at right angles to each other and of the same length. An $n-$dimensional hypercube is also called an $n-$cube.



In parallel computing, the vertexes are processors, and the line segments (edges) represent connections. The $n-$cube architecture has the following properties:

- Each node has n connections with different processors.
- Each processor has a unique identifier, between 0 and $2^n - 1$.
- Two processors are directly connected if and only if their identifiers differ in just one bit. For instance, in a $3-$cube, processors 3 (011 in binary) and 7 (111 in binary) are directly connected.
- The number of processors is 2^n

The new company WEFAIL is designing hypercubes, but they are always contracting new people, whose do not know all the hypercube properties, and sometimes they fail; thus these properties are not satisfied in all cases.

Given an arbitrary graph, your task is to write a program that determines whether the graph is a hypercube or not.

Input

The input consists in several problem instances. Each instance contains one graph, which starts with a line with two positive integers: K and M, representing the number of vertexes ($0 < K \leq 1024$) and the number of edges respectively. It follows ($0 \leq M \leq 5130$) lines, representing the edges. Each edge is given by two 32 bits integers, representing the processors connected by the edge.

The end of input is indicated by a test case with $K = 0$.

The input must be read from standard input.

Output

For each problem instance, the output is a single line, with the word “YES” if the corresponding graph is a hypercube, and “NO” otherwise (quotes for clarity).

The output must be written to standard output.

Sample input

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

Output for the sample input

```
YES
NO
NO
```