0 SimpleSum

0.1 Problem

Given a sequence of n integers $a_1, \ldots, a_n$, compute the sum $s = \sum_{i=1}^{n} a_i$.

0.2 Input

The first line contains a single integer n ($1 \leq n \leq 1000$). n lines follow, with the ith line containing a single integer representing a_i ($0 \leq a_i \leq 100$).

0.3 Output

Output the sum s followed by a newline character.

0.4 Sample Data

Input	**Output**
`3`	`45`
`1`	
`2`	
`42`	

1 The 3n + 1 problem

1.1 Problem

Problems in Computer Science are often classified as belonging to a certain class of problems (e.g. NP, Unsolvable). In this problem you will be analyzing a property of an algorithm whose classification is not known for all possible inputs.

Consider the following algorithm:

```
1. input $n$
2. print $n$
3. if $n = 1$ then STOP
4. if $n$ is odd, then $n := 3n + 1$
5. else $n := n/2$
6. goto 2
```

Given the input 22, the following sequence of numbers will be printed:

$$22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1$$

It is conjectured that the algorithm above will terminate (when a 1 is printed) for any integral input value. Despite the simplicity of the algorithm, it is unknown whether this conjecture is true. It has been verified, however, for all integers in $\{1, \ldots, 1000000\}$ (and, in fact, for many more numbers than this).

Given an input n, it is possible to determine the length of the sequence of printed numbers (including the 1). For a given n this is called the Collatz-length of n. In the example above, the Collatz-length of 22 is 16. For two given numbers i and j you are to determine the maximum Collatz-length of numbers in $[i, j]$.

1.2 Input

The input consists of a pair of integers i and j $(1 \leq i \leq j < 1000)$.

1.3 Output

Output the maximum Collatz length for starting values in $\{i, i+1, \ldots, j-1, j\}$, followed by a newline character.

1.4 Sample Data

Input	Output
`1 10`	`20`
`100 200`	`125`
`201 210`	`89`
`900 999`	`174`

2 Speeding

2.1 Problem

Always the troublemaker, Bessie the cow has stolen Farmer John's tractor and taken off down the road!

The road is exactly 100 miles long, and Bessie drives the entire length of the road before ultimately being pulled over by a police officer, who gives Bessie a ticket for exceeding the speed limit, for having an expired license, and for operating a motor vehicle while being a cow. While Bessie concedes that the last two tickets are probably valid, she questions whether the police officer was correct in issuing the speeding ticket, and she wants to determine for herself if she has indeed driven faster than the speed limit for part of her journey.

The road is divided into N segments, each described by a positive integer length in miles, as well as an integer speed limit in the range $1 \ldots 100$ miles per hour. As the road is 100 miles long, the lengths of all N segments add up to 100. For example, the road might start with a segment of length 45 miles, with speed limit 70, and then it might end with a segment of length 55 miles, with speed limit 60.

Bessie's journey can also be described by a series of segments, M of them. During each segment, she travels for a certain positive integer number of miles, at a certain integer speed. For example, she might begin by traveling 50 miles at a speed of 65, then another 50 miles at a speed of 55. The lengths of all M segments add to 100 total miles. Farmer John's tractor can drive 100 miles per hour at its fastest.

Given the information above, please determine the maximum amount over the speed limit that Bessie travels during any part of her journey.

2.2 Input

The first line of the input contains N and M, separated by a space.
The next N lines each contain two integers describing a road segment, giving its length and speed limit.
The next M lines each contain two integers describing a segment in Bessie's journey, giving the length and also the speed at which Bessie was driving.

2.3 Output

Please output a single line containing the maximum amount over the speed limit Bessie drove during any part of her journey. If she never exceeds the speed limit, please output 0.

2.4 Sample Data

Input	Output
3 3	5
40 75	
50 35	
10 45	
40 76	
20 30	
40 40	

3 Bicoloring

3.1 Problem

In 1976 the “Four Color Map Theorem" was proven with the assistance of a computer. This theorem states that every map can be colored using only four colors in such a way that no region is colored using the same color as a neighbor region. Here you are asked to solve a simpler similar problem. You have to decide whether a given arbitrary connected graph can be bicolored. That is, if one can assign colors (from a palette of two) to the nodes in such a way that no two adjacent nodes have the same color. To simplify the problem you can assume:

- There are no loops, i.e. no node has an edge to itself.
- There are no multi-edges, i.e. from any node to any other node there is at most one edge.
- The graph is undirected.
- The graph is strongly connected. That is, for each pair a, b of nodes, there exists a path from a to b in the graph.

The nodes of the graphs are identified by the integers 0 to $n-1$ for some n.

3.2 Input

The first line contains an integer n, specifying the number n of nodes in the graph ($1 < n \leq 200$). The second line contains the number m of edges ($1 \leq m \leq \frac{n(n-1)}{2}$). m lines follow, each containing two integers a and b ($0 \leq a, b < n$), specifying that there is an (undirected) edge between the nodes a and b.

3.3 Output

Output a single line containing the literal string *“BICOLORABLE.”* or *“NOT BICOLORABLE.”* (without the quotes), specifying whether the graph is bicolorable or not. Terminate the output with a single newline character.

3.4 Sample Data

Input	Output
3	NOT BICOLORABLE.
3	
0 1	
1 2	
2 0	
9	BICOLORABLE.
8	
0 1	
0 2	
0 3	
0 4	
0 5	
0 6	
0 7	
0 8	

4 Oil Deposits

4.1 Problem

The GeoSurvComp geologic survey company is responsible for detecting underground oil deposits. GeoSurvComp works with one large rectangular region of land at a time, and creates a grid that divides the land into numerous square plots. It then analyzes each plot separately, using sensing equipment to determine whether or not the plot contains oil.

A plot containing oil is called a pocket. If two pockets are adjacent, they are part of the same oil deposit. Oil deposits can be quite large and may contain numerous pockets. Your job is to determine how many different oil deposits are contained in a grid.

4.2 Input

The input file contains one or more grids. Each grid begins with a line containing m and n, the number of rows and columns in the grid, separated by a single space. If $m = 0$ it signals the end of the input; otherwise $1 \leq m \leq 100$ and $1 \leq n \leq 100$.

Following this are m lines of n characters each (not counting the end-of-line characters). Each character corresponds to one plot, and is either `*`, representing the absence of oil, or `@`, representing an oil pocket.

4.3 Output

For each grid, output the number of distinct oil deposits. Two different pockets are part of the same oil deposit if they are adjacent horizontally, vertically, or diagonally.

4.4 Sample Data

Input	Output
1 1 * 3 5 *@*@* **@** *@*@* 1 8 @@****@* 5 5 ****@ *@@*@ *@**@ @@@*@ @@**@ 0 0	0 1 2 2

5 Sum2D

5.1 Problem

You are given a rectangular field of integers with a width W and a height H. These integers are all in the range [-1000,1000]. Your task is to write a programm that computes the sum of all integers in a rectangular subfield. You are therefore additionally given a list of Q subrectangles. For each of these subrectangles you should compute the sum.

5.2 Input

The first line contains W, H and Q in that order.

The next H lines contain each W integers seperated by a space. This is the rectangular field.

After that another Q lines follow with 4 integers x_1, y_1, x_2, y_2 in that order. (x_1, y_1) and (x_2, y_2) form two opposing edges of a subrectangle. Coordinates are in the range [1,W] or [1,H]. The edges should be included in the subrectangle.

H and W will not exceed 1000.

5.3 Output

For each subrectangle print the sum of the integers in it on a line of its own.

5.4 Sample Data

Input	Output
`5 4 3`	`42`
`1 5 -3 8 -4`	`1`
`2 4 -1 -5 9`	`-9`
`10 1 0 -8 5`	
`3 5 9 0 1`	
`1 1 5 4`	
`2 3 2 3`	
`4 3 2 2`	

6 Fish

6.1 Problem

In a small coastal country, all towns are situated on a long coastline (which we will model as a straight line). A long straight road runs along the coast, connecting the towns. The position of each town can be described by a single non-negative integer, the distance (in kilometers) from the start of the road.

Most of the citizens are fishermen, and they catch huge amounts of fish. After the fishing season is over and before the tourist season starts, the fish can be transported between different towns. A town can accommodate x tourists if it has x tons of fish available. The goal is to accommodate the largest possible number of tourists while distributing them evenly between towns. In other words, we want to find the largest integer y for which it is possible to distribute fish so that each town can accommodate at least y tourists.

In one shipment, an integral number of tons of fish is sent from one town to another. During transportation, one ton of fish per kilometer traveled is lost to hungry pillagers descending from the mountains. More formally, if a town ships f tons of fish to another town that is d kilometers away, then $f - d$ tons will arrive at the destination; if f is less than d, then the entire shipment is lost.

It is possible to arbitrarily repackage and combine shipments in intermediate towns. For example, we can send shipments from towns a and b to town c, combine half of the remaining fish from both shipments with the fish originating in c and send it in a single large shipment from town c to town d.

Write a program that, given the positions of all towns and the amount of fish each town produces, determines the largest number of tourists that can be accommodated by each city after the fish have been distributed.

6.2 Input

The first line contains a single integer n, $1 \leq n \leq 10^5$, the number of towns. Each of the following n lines contains two integers p and f, $0 \leq p$, $f \leq 10^{12}$, the position of a town (in kilometers) and the amount of fish it produces (in tons). The towns will be sorted in ascending order of position. The positions of all towns will be distinct.

6.3 Output

Output the largest number of tourists y from the task description followed by a newline character.

6.4 Sample Data

Input	Output
5	6
1 0	
2 21	
4 0	
5 7	
7 6	

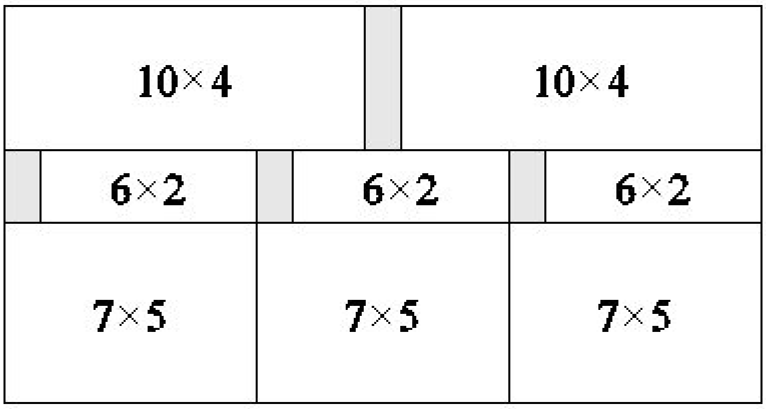
7 Lost Marbles

7.1 Problem

You have been given a large rectangular slab of marble, which you are supposed to cut to rectangular marble plates of sizes $W_1 \times H_1, W_2 \times H_2, \ldots, W_n \times H_n$.

Any piece of marble (the slab or the plates cut from it) can be cut either horizontally or vertically into two rectangular plates with integral widths and heights, cutting completely through that piece. This is the only way to cut pieces and pieces cannot be joined together. Since the marble has a pattern on it, the plates cannot be rotated: if you cut a plate of size $A \times B$ then it cannot be used as a plate of size $B \times A$ unless $A = B$. A marble plate is wasted if it is not of any of the desired sizes after all cuts are completed.

You can make zero or more plates of each desired size as long as you minimize the amount of wasted marble. Because no one wants you losing your marbles.



As an example, assume that the size of the original slab is 21×11, and the desired plate sizes are 10×4, 6×2, 7×5, and 15×10. The minimum possible area wasted is 10, and the figure shows one sequence of cuts with total waste area of size 10.

7.2 Input

The first line of input contains two integers W and H ($1 \leq W, H \leq 600$), the size of the original slab. The second line contains one integer n ($1 \leq n \leq 200$): the number of desired plate sizes. The following n lines each contain two integers W_i and H_i ($1 \leq W_i \leq W, 1 \leq H_i \leq H$), the desired plate sizes.

7.3 Output

Print a single integer: the minimum total area of the original slab that must be wasted.

7.4 Sample Data

Input	Output
`21 11`	`10`
`4`	
`10 4`	
`6 2`	
`7 5`	
`15 10`	

8 Modulo Data Structures

8.1 Problem

You have an array, $Arr[1 \ldots N]$. Initially, all values of $Arr[i] = 0$. The array supports the following two types of querries:

1. Increase all $Arr[k]$ by C for all $k \equiv A (\bmod B)$
2. Output $Arr[D]$ for a given D.

8.2 Input

The first two lines of input consists of two integers, N and Q, representing the length of the array and the number of queries, respectively. Note that $1 \leq N, Q \leq 200000$.

The following Q lines all begin with an integer T representing the type of query. If $T = 1$, then it will be followed by three integers representing A, B and C respectively. Note that $1 \leq C \leq 10000, 0 \leq A \leq B \leq N$. Else, T will be 2, and it will be followed by one integer representing D, subjected to $1 \leq D \leq N$.

8.3 Output

For each query of type 2, output the integer value of $Arr[D]$ on a separate line.

8.4 Sample Data

Input	Output
`5 10`	`7`
`1 1 2 7`	`3`
`1 4 5 6`	`7`
`1 2 3 3`	`10`
`2 1`	`6`
`2 2`	`10`
`2 3`	
`1 3 4 3`	
`2 3`	
`2 4`	
`2 5`	