A program that will create a nonogram puzzle (https://en.wikipedia.org/wiki/Nonogram) from a binary image. The image is given as a series of values between 0 and 255 from standard input (stdin). You will need to translate this input and extract the runs for each **column** and **row** from the image. Each row and column has a collection of runs and the length of each run is listed in sequence from last run to first run.

- Width and Height are > 0
- You need to support numbers between **0** or **255** in the image
- The image width and height needs to match the given width and height
- If no width and height are given the program should not attempt to read in the data
- If an error occurs reading input from stdin the program should terminate with 1.
- Depending on what property is associated with causing the error the relavent error message should be display. Check invalid input examples
- In the case of invalid input it is based on first seen, So if there is two or more input that are invalid, the first one seen is what is considered the cause.
- When the end of a file is reached an EOF character is provided. Your program should not hang after the file has been read in.

You will be given a width and height on the first line of standard input denoted like so:

```
6 7
```

Which is width and height respectively.

If we were to consider the runs on row 0 (x[0])

```
0 0 1 1 0 1 0 1
```

Output:

```
2 1 1
```

If a row or column has no runs then it has the value of 0

Example:

```
0 0 0 0 0 0 0
```

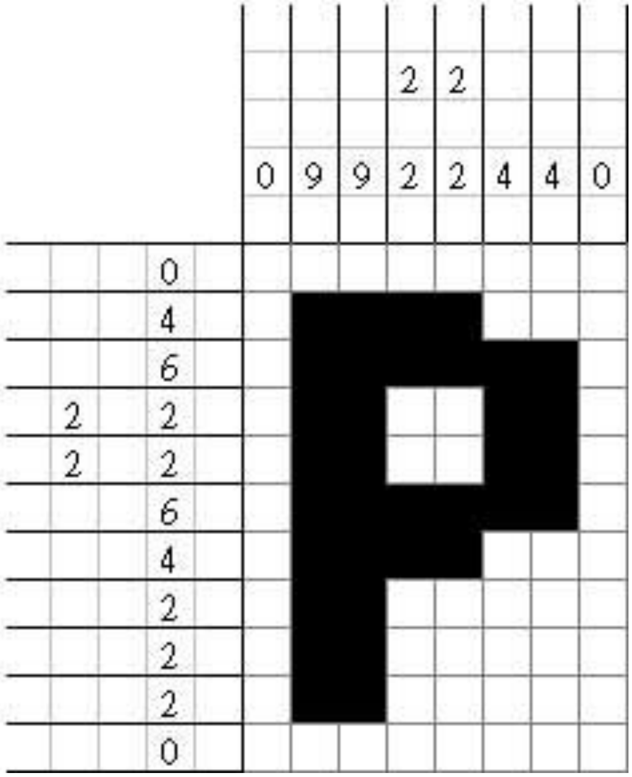
Output:

```
0
```

Sample Nonogram Input:

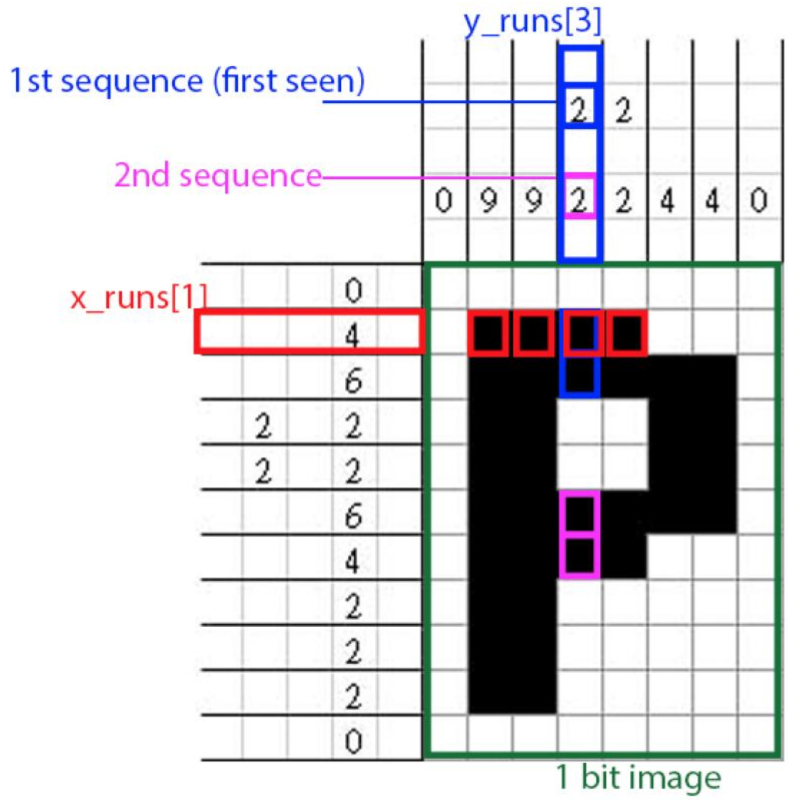
```
6 4
0 1 1 1 1 0
1 0 0 0 0 1
1 0 0 0 0 1
0 1 1 1 1 0
```

An example of a binary nonogram image:



(https://en.wikipedia.org/wiki/Nonogram#/media/File:Nonogram2.jpg)

This may help with understanding how nonograms work:



Example 1:

Input

```
6 4
0 1 2 2 1 0
1 0 0 0 0 1
1 0 0 0 0 1
0 1 1 1 1 0
```

Output:

```
X:
1 2 1
1 1
1 1
4

Y:
2
1 1
```

```
1 1
1 1
1 1
2
```

Example 2:

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

Output:

```
X:
0
5
2 1
2 2 1
2
2

Y:
0
1 2 2
1 2 2
1 1
1 1
2 1
0
```

Example 3:

Input:

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

Output:

X:
2 2
1
3
2 1

Y:
1
1 1 1
1 1 1
1 2
1

If no width or height (or width and/or height are <= 0) are specified then the program should output that it cannot decode the image

Output: Cannot decode

When a height or width is specified but the image itself doesn't match those specifications your program should output

Output: Cannot decode