1. You are given a maze in the form of a matrix of size n * m. Each cell is either clear or blocked denoted by 1 and 0 respectively.
A rat sits at the top-left cell and there exists a block of cheese at the bottom-right cell. Both these cells are guaranteed to be clear.
You need to find if the rat can get the cheese if it can move only in one of the two directions - down and right. It can't move to blocked cells.

Input Format

The first line contains an integer T, denoting the number of test cases.

For each test case the input has the following lines:

The first line contains two space-separated integers n and m denoting the number of rows and columns of the matrix respectively.
The next n lines contain m space-separated integers which are either 0 or 1.

Output Format
For each test case, a line containing 1 or 0 based on whether the rat can get the cheese or not respectively.

Sample Input
4
4 4
1 1 0 0
1 1 1 1
0 1 0 1
0 1 0 1
4 4
1 1 0 0
1 1 1 1
0 1 1 0
0 1 0 1
4 5
1 0 1 1 1
1 1 1 0 1
0 1 0 0 1
0 1 1 0 1
3 4
1 0 0 0
0 0 0 0
0 0 1 1

Expected output
1
0
0
0

Constraint
1 <= T <= 100
1 <= n, m <= 100

0 <= mazeij <= 1
==========
2. Have you ever been a part of the exciting game Passing the Parcel ? Sid is on a school picnic with his classmates. The teacher decides to make the whole class play the game of Passing the Parcel. Since the winner of the game gets lots of chocolates and ice cream as his/her prize, all the students are over-excited about the game, including Sid. Here are the rules of the game:

For the purpose of the game, our Parcel here is a football.
There are a total of N students in the class. Their roll numbers being .
All N students are made to sit uniformly in a circle in roll number order (ie. from 1 to N in clockwise direction).
The Parcel is first handed over to the student with roll number 1.
The teacher starts playing a song using a loud stereo system. The lyrics of the song are denoted by a string which consists of only letters 'a' and 'b'. Assume that each lyric of the song is a single letter.
If the lyric 'a' occurs in the song, the student who is currently holding the Parcel passes it on to the next student. This passing takes place in clockwise direction.
If the lyric 'b' occurs in the song, the student who is currently holding the Parcel loses his/her chances of winning the game. He/she hands over the parcel to the next student (in clockwise direction) and moves out.
The game continues until a single student survives in the end. He/she will be the winner of the game.
Note that the song repeats continuously ie. while the game is going on, if at all the song ends, the stereo system will automatically start playing the song from the start without any delay.
Given N the number of students in the class and the lyrics of the song, you have to find out the roll number of the student who wins the game.

Input: The input consists of 2 lines. The first line consists of N, the number of students in the class. The next line consists of a string denoting the lyrics of the song the teacher plays.

Output: Print a single integer denoting the roll number of the student who wins the game.

Constraints :
2<=N<=1000
1<=|S|<=1000, where |S| denotes the length of the input string.
It is guaranteed that at least 1 lyric in the song will be a 'b'.

Explanation
a : 1->2 b : 2 goes out, handing over the parcel to 3 b : 3 goes out, handing over the parcel to 4 a : 4->5 a : 5->6 b : 6 goes out, handing over the parcel to 1 b : 1 goes out, handing over the parcel to 4 a : 4->5 a : 5->4 b : 4 goes out.

Hence the winner is 5.
==========
3. You are given a 0-indexed array of integers nums of length n. You are initially positioned at nums[0].

Each element nums[i] represents the maximum length of a forward jump from index i. In other words, if you are at nums[i], you can jump to any nums[i + j] where:
--> 0 <= j <= nums[i] and
--> i + j < n

Return the minimum number of jumps to reach nums[n - 1]. The test cases are generated such that

you can reach nums[n - 1].

Example 1:
Input: nums = [2,3,1,1,4]
Output: 2
Explanation: The minimum number of jumps to reach the last index is 2. Jump 1 step from index 0 to 1, then 3 steps to the last index.

Example 2:
Input: nums = [2,3,0,1,4]
Output: 2

Constraints:
1 <= nums.length <= 10^4
0 <= nums[i] <= 1000
It's guaranteed that you can reach nums[n - 1].
==========
4. You have n boxes. You are given a binary string boxes of length n, where boxes[i] is '0' if the ith box is empty, and '1' if it contains one ball. In one operation, you can move one ball from a box to an adjacent box. Box i is adjacent to box j if abs(i - j) == 1. Note that after doing so, there may be more than one ball in some boxes. Return an array answer of size n, where answer[i] is the minimum number of operations needed to move all the balls to the ith box. Each answer[i] is calculated considering the initial state of the boxes.

Example 1:
Input: boxes = "110"
Output: [1,1,3]

Explanation: The answer for each box is as follows:
1) First box: you will have to move one ball from the second box to the first box in one operation.
2) Second box: you will have to move one ball from the first box to the second box in one operation.
3) Third box: you will have to move one ball from the first box to the third box in two operations, and move one ball from the second box to the third box in one operation.

Example 2: Input: boxes = "001011" Output: [11,8,5,4,3,4]
==========
5. A word chain is a sequence of words [word1, word2, ..., wordk] with k >= 1, where word1 is a predecessor of word2, word2 is a predecessor of word3, and so on. A single word is trivially a word chain with k == 1.
Return the length of the longest possible word chain with words chosen from the given list of words.

Example 1:
Input: words = ["a","b","ba","bca","bda","bdca"]
Output: 4
Explanation: One of the longest word chains is ["a","ba","bda","bdca"].

Example 2:
Input: words = ["xbc","pcxbcf","xb","cxbc","pcxbc"]
Output: 5
Explanation: All the words can be put in a word chain ["xb", "xbc", "cxbc", "pcxbc", "pcxbcf"].

Example 3:
Input: words = ["abcd","dbqca"]
Output: 1
Explanation: The trivial word chain ["abcd"] is one of the longest word chains.
["abcd","dbqca"] is not a valid word chain because the ordering of the letters is changed.

Constraints:
a. 1 <= words.length <= 1000
b. 1 <= words[i].length <= 16
c. words[i] only consists of lowercase English letters.
d. word1 is a predecessor of word2 if and only if we can insert exactly one letter anywhere in word1 without changing the order of the other characters to make it equal to word2.
==========