Problem 1. Circular Railway

(Time Limit: 2 seconds)

Problem Description

There is a circular railway system for sightseeing. Along the railway, there are n stations labeled from 0 to n-1 clockwise. The tourists need to pay a fee for entering or exiting a station. Also there is a fee for traveling between two consecutive stations and a tour from a station to another takes a cost of the total fee of entering station, exiting station, and the sum of the traveling fees the tour passes.

As you know, there are two routes between two stations because the railway is circular. However, tourists may not always choose the cheaper one since it is for sightseeing. Now you are asked to find the most expensive tour from one station to another. Note that the tour cannot pass a station twice or start and finish at the same station, i.e., a tour cannot travel the whole railway. For each test case:

- The station number $n \leq 60000$.
- The fees for entering and exiting a station are the same and are positive 4-byte integers.
- The railways are bidirectional and the costs are the same for both direction.
- The traveling cost from stations i to i+1 (mod n) is a positive 16-bits integer.

Input Format

There are several test cases. Each case starts with an integer n which is the number of stations. There are n lines followed. Each line has two integers $w(i)$ and $l(i,i+1 \bmod n)$ for i from 0 to n-1, and the two numbers are separated by a space, in which $w(i)$ is the cost for entering or exiting station i and $l(i,i+1)$ is the cost for traveling between i and i+1. The case n = 0 ends the input and you don't need to process it.

Output Format

For each test case, output the most expensive fee in one line.

Example

Sample Input:	Sample Output:
3 100 20 200 10 200 30 4 100 10 50 20 100 10 30 30 0	450 240

Problem 2. The Construction of an Amusement Park

(Time Limit: 3 seconds)

Problem Description

The Adventures of Computation and Magic (ACM) Inc. is initiating the construction plan of an amusement park. The park will be divided into several regions and each region will accommodate one attraction. The designer offers different design options for each region. As the project leader of the construction plan, you have to choose exactly one option for each region. As the budget is limited, you cannot spend more money than it, but you indeed want to spend the maximum possible. The construction plan may fail in case that you cannot choose one option for each region due to the short amount of money.

Technical Specification

1. The budget would be an integer M such that $1 \leq M \leq 200$.
2. The number of regions (attractions) C would satisfy $1 \leq C \leq 20$.
3. The number of options K for each region would satisfy $1 \leq K \leq 20$.

Input Format

The first line of the input contains an integer N, indicating the number of test cases. For each test case, some lines appear, the first one contains two integers, M and C, separated by blanks ($1 \leq M \leq 200$ and $1 \leq C \leq 20$), where M is the available amount of money and C is the number of regions (attractions) in the construction plan. Following this line, there are C lines, each one with some integers separated by blanks; in each of these lines the first integer, K ($1 \leq K \leq 20$), indicates the number of different design options for each attraction and it is followed by K integers indicating the construction cost of each option of that attraction.

Output Format

For each test case, the output should consist of one integer indicating the maximum amount of money necessary for the construction plan without exceeding the initial budget. If there is no solution, you must print `plan failure'.

Example

Sample Input:	Sample Output:
3 100 4 3 8 6 4 2 5 10 4 1 3 3 7 4 50 14 23 8 20 3 3 4 6 8 2 5 10 4 1 3 5 5 5 3 3 6 4 8 2 10 6 4 7 3 1 7	75 19 plan failure

Problem 3. The Gladiators

(Time Limit: 3 seconds)

Problem Description

In ancient Rome, an arena was a place used for Gladiatorial contests. Gladiators were armed combatant who entertained audiences. Each gladiator was trained for fighting styles and able to use many kinds of weapons. A Gladiator would fight with the others who could use an identical weapon as he could. For example, one gladiator can use sword, bow and axe and the other can use axe and spear, then they would fight with each other. In general, the former would win the fight, because he could use more kinds of weapons. However, if two gladiators could use the same number of kinds of weapons, no one will win. Please write a program to select the top k gladiators in terms of the number of wins.

Technical Specification

- All the given numbers are integers and there is no sign.
- The number of gladiators n is from 1 to 30,000.
- The number of kinds of weapons m is from 1 to 64.
- k is from 1 to n.

Input Format

The first line is an integer which indicates the number of test cases. Each case begins with three integers n, m and k. The following n lines represents the specialties that a gladiator had. Each line is composed of m digits, either 1 or 0. One means the gladiator could use this weapon, otherwise not.

Output Format

For each test case, output the number of wins of the top k gladiators in one line and separates them by a space.

Example

Sample Input:	Sample Output:
2 4 4 3 1000 1100 1110 1111 3 5 2 01010 10101 11110	3 2 1 2 0

Problem 4. Very Even Number

(Time Limit: 1 second)

Problem Description

HH is good at math. He just learnt the concept of even and odd numbers. But he doesn't like odd numbers since odd numbers are too odd. However, he likes even numbers very much since even numbers are very even!

Some even numbers called "very even number" are his favorite numbers. We say a number is a "very even number" if all its digits in decimal representation are even numbers.

For example, both 666 and 20480 are very even numbers; but 314, 777, 28825252 are not. HH is also learning the concept of modulo and remainder.

To practice, he wants to know what's the smallest very even number x, such that $x \bmod m = r$?

Technical Specification

- The number of test case $T \leq 250$
- $1 \leq m \leq 100000$
- $0 \leq r < m$

Input Format

The first line contains an integer T indicating the number of the test cases. For each test case, there are two integers m, r in one line.

Output Format

For each test case, output the smallest very even number x, such that $x \bmod m = r$. If that number doesn't exist, output -1.

Example

Sample Input:	Sample Output:
3 77 7 100 77 123 45	84 -1 660

Problem 5. Save the Martian

(Time Limit: 3 seconds)

Problem Description

The crew of *Ares V*, a manned mission to Mars, has established an artificial habitat on Mars. They wish to stay for years in order to investigate the biodiversity of bacteria on Mars. However, a massive Martian storm hits the base and they are forced to abandon the base. These crews evacuate immediately and independently because of the fear that their escape vehicle will collapse and leave them stranded. After the chaos of evacuation, astronauts are separated in different isolated regions on the Mars and live on his/her own with limited food, water, etc.

Unfortunately, the bacteria on Mars are all lethal to human, and different strains of bacteria on Mars consume different material as food resources. For example, the red strain of bacteria like to drink water and the blue strains are addicted to sweet cookies. Six months later, these astronauts are running out of their own supply (e.g., battery, cookie, water). Some of them are even infected by lethal bacterial strains. With the lives of these crew at stake, mission commander Melissa Lewis has to deliver proper supply items to save their lives. Each crew member independently asks Melissa for specific supply items, and/or declines specific items for preventing further infection from lethal bacteria.

However, due to the poor communication between the crews and Melissa and limited source on the spaceship, she can only take two requests from each crew member, where each request may ask for and/or decline a specific item. For example, one crew may send two requests asking for both water and food. Another may only need at least one of the two items for survival (e.g., water or food). For the crews infected by the red strain of bacteria, they may ask for food and decline water in case of further infection. In addition, some crews are infected by a very greedy yet kind strain of bacteria which always attack the humans possessing two items (e.g., both water and food), but they have no interest in the poor humans having little or no items. Therefore, the crew may ask for exactly one of the items but not both.

On the spaceship, Melissa has prepared *N* groups of robots, where *N* is equal to the number of supply items. Each group contains *M* robots, where *M* is the same as the total number of crews. Each robot group is responsible for transporting a specific item, and each robot in the group is programmed to fly to a specific crew member. Moreover, the robots of the same group can only be either launched all together or all of them should stand by on the spaceship. Different groups of robots can be launched or stay independently. Note that owing to the lethal bacteria surrounding some crews, a subset of robot groups may be better not launched. In addition, the launch of any robot group will run out of the energy of the spaceship, and Melissa will never be able to return to earth. Therefore, if there exists a subset of robot groups such that all the crews' requirements can be met by launching them, Melissa will deliver these robots. Otherwise, Melissa will just fly back to earth on her own, since there is no way to save all of them.

Melissa is now sitting in the control room and wondering whether she should launch a subset of robots or not. Given *N* groups of robots, and the two requests from *M* crew members, write a program that helps Melissa determine whether she can launch a subset of the robot groups for saving all crews, or she better just flies back to earth.

Technical Specification

- The number of crews *M* ranges from 1 to 10000.
- The number of supply items *N* ranges from 1 to 500.
- Each request for a specific supply items is represented as an integer ranging from 1 to *N* or from -1 to -*N*. The positive and negative integers (*i*) indicates a crew need and decline the *i*-th item, respectively.
- Each crew can only issue two requests (i.e., ask/decline for two specific items), which are represented as two positive/negative integers.
- Each crew will also indicate at least one (+), both (*), or exactly one (/) of the two requests should be met.

Input Format

The first line contains the number of test cases. Each of the following two lines represents a test case. For each test case, the starting line stores the numbers of items (*N*) and of crews (*M*) separated by a white space. The next line stores the (two) requests from each crew separated by a white space. For each crew, there are two integers indicating he/she asks for (positive) or declines (negative) specific items. Between the two integers, there is an operator representing at least one request should be satisfied (+), both requests should be satisfied (*), or either one (but not both) request should satisfied (/). Note that the two integers and operator are also separated by a white space.

For the first test case shown below, "1 + 2" means that the first crew asks for the supply of 1st or the 2nd items, and he can survive if at least one item is sent. "1 * -2" means that the second crew can only survive if 1st item is delivered and 2nd item is not.

Output Format

The output should consist of one line for each test case. Each line contains “Yes” if all crew members can be saved by any robot launching procedure.

Otherwise, output “No” if there is no way to save all crews. For the first test case shown below (i.e., “1 + 2 1 * -2”), Melissa can launch the robot groups carrying 1st items and stop the other carrying the 2nd items to save both crews. But there are no launching options for her to save all crews in the 2nd test case.

Example

Sample Input:	Sample Output:
2 2 2 1 + 2 1 * -2 3 4 1 + 2 2 * 3 3 / -1 1 * -2	Yes No