Playing partitions

Partitions again?? Man I'm getting sick of these things. I'm sure you are too. But this time we're going to do something a little different. We're going to play a game with them. Or at least, we're going to study a game that could be played with them.

A position in this game is simply a Ferrers board, and a move is exactly one of the moves described in *Pathfinding partitions*. The game is played by two players, who take turns making moves. The game is defined by a starting position, and a list of target positions. The winner is the first player to achieve one of the target positions. The game is drawn if each player can prevent the other from achieving a target position.

Task

Write a program that takes input from `stdin` a data file formatted according to the rules of the *Parsing Partitions* étude where each scenario in the file will consist of a starting position, and a list of target positions. The starting position will never be one of the target positions. The output (to `stdout`) should be in standard form. Each scenario should be represented by the starting position, a blank line, the list of target positions, and then a single comment line which is one of the following:

```
# WIN
# LOSE
# DRAW
```

indicating the outcome of the game from the starting position for the first player assuming that both players are trying to win if possible.

Example

Input:

```
2 1
3
---
3 2 1
2 2 1 1
```

Output:

```
2 1

3
# DRAW
---
3 2 1

2 2 1 1
# WIN
```

In the first scenario, the legal moves are to $(2, 1)$ (using the first column) or $(1, 1, 1)$ (using the second column). But, the second move is bad since it allows an immediate win for the second player. So, the first player will "do nothing" and the game will be drawn (since the second player will do the same, and so on until they get fed up and quit).

In the second scenario, the first player can win immediately by removing the second column.

Standards

For an achieved standard, the program must work correctly on valid input representing partitions of size 20 or less.

Merit criteria include the ability to handle much larger partitions efficiently, handling poorly-formatted input gracefully, and clearly written code.

Excellence criteria include some significant extension to the functionality of the program, or an investigation of general properties of the problem. Pair or group submissions for excellence standard will be considered (but individual submissions of the main part of the étude are still required).

Objectives

2.2, 2.5, 2.7, 2.9, 3.3-3.5, 3.7, 4.1

(Individual)