Zodiac Fever

Since you found the life of programmer boring, you start treasure hunting on your own. Recently you discovered a map, revealing the vast treasure left over by an ancient tribute and you want to hunt down the treasure. However, being the worshipper of the god of wisdom --- ЌзЙрóóб, the tribute has high civilization level and the maze is packed with clever traps. Particularly, the door of the treasure room is locked with the Zodiac Ring as shown in Fig 1:

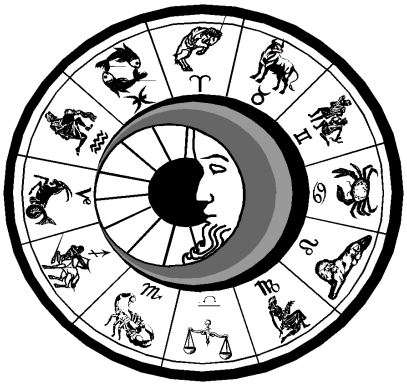

Figure 1

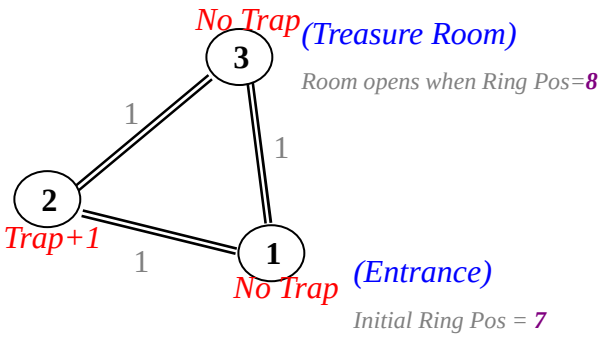

Figure 2

The (*turn-able*) ring is subdivided into equal sectors, representing the zodiac constellations with a numbering system from 1 to 12. The door is designed in a way that it opens if and only if the ring is rotated to the correct position. However, the problem is that the ring could not be turned at will, but instead, it is controlled by triggers(traps) scatter in different positions of the maze.

There are four types of traps in the maze, each will be triggered once you step on it: Type one rotates the ring clockwise by some number of sector(s) (i.e. **+ *n***, *for some non-negative integer* ***n***). Type two rotates in counter-clockwise direction (i.e. **– *n***). Type three rotate the ring to some predetermined position (i.e. **= *n***). Finally the last type rotates the wheel to some multiple of the current position (i.e. *x* ***n***). Since the wheel is round, *all* numbers wrap around, for instance, adding **2** to **11** gives **1**.

Given the map of the maze and the positions of the traps, your task is to determine the minimal travel distance required *(from the starting point)* to get the treasures. Consider the map in Figure 2, consisting of three rooms *(numbered 1..3)* and corridors connecting the rooms. Your starting point is room1, and the treasure room is room3. Among the three rooms, only room2 has a trap, which turns the ring clockwise by one sector (i.e. +1). The treasure room is initially locked, since the ring position is now at 7 *(instead of 8, which opens the door)*. To get the treasure, the minimal travel route required is travelling from room1 to room2 (stepping on the trap there), and then from room2 to the room3. Therefore the minimal travel distance here is 2.

Note the followings:

- Every room (including the entrance and treasure room) has at most 1 trap
- When you visit a room, the trap there will ALWAYS be triggered
- You may walk on corridors from either direction, but you cannot turn in the midway
- Traps could be triggered repeatedly, but you must visit another room before coming back
- Your route always end at the treasure room (since you have no assistant helping you)
- By the time you visit the treasure room, the ring position *(after considering the trap there, if any)*, should be the same as the "open position". Even if the treasure room was opened midway in your trip, the door is locked again by the time the ring moves again

INPUT

Input is consisted of multiple test cases, continue until the end of file. The first line of test case contains 7 integers, representing the ***total_number_of rooms*** (no greater than 1000), ***number_of_corridors***, ***number_of_traps***, ***room#_of_entrance***, ***initial_ring_position***, ***room#_of_treasure_room*** and the ***open_ring_position*** respectively. Note that the ***initial_ring_position*** has already considered the effect of the trap in entrance, if any.

Following the first line is the list of corridors. Each line contains the (different) ***room numbers of both ends of the corridor***, followed by a non-negative integer showing the length.

Finally comes the list of traps, each line contains the ***room number***, ***trap type*** (+, -, ***** or =) and a non-negative integer ***n***, specifying the ***trap movement behavior***.

OUTPUT

Print, on a line by itself, the minimal travel distance required to get the treasure. If there is no solution, print "**`Pray!`**" (without quotes).

SAMPLE INPUT *(corresponding to figure 2)*

```
3 3 1 1 7 3 8
1 2 1
2 3 1
1 3 1
2 + 1
```

SAMPLE OUTPUT

```
2
```