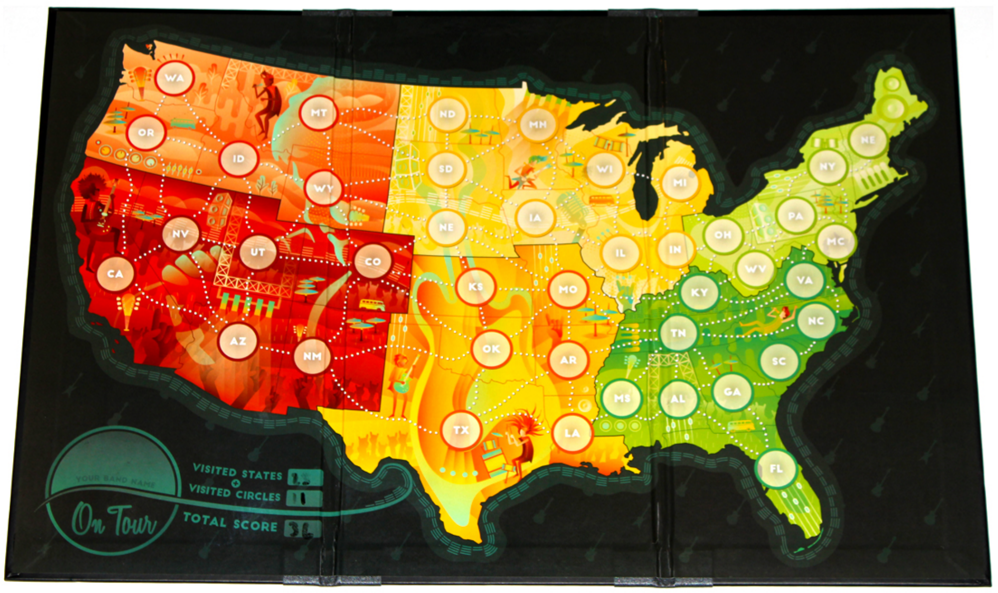
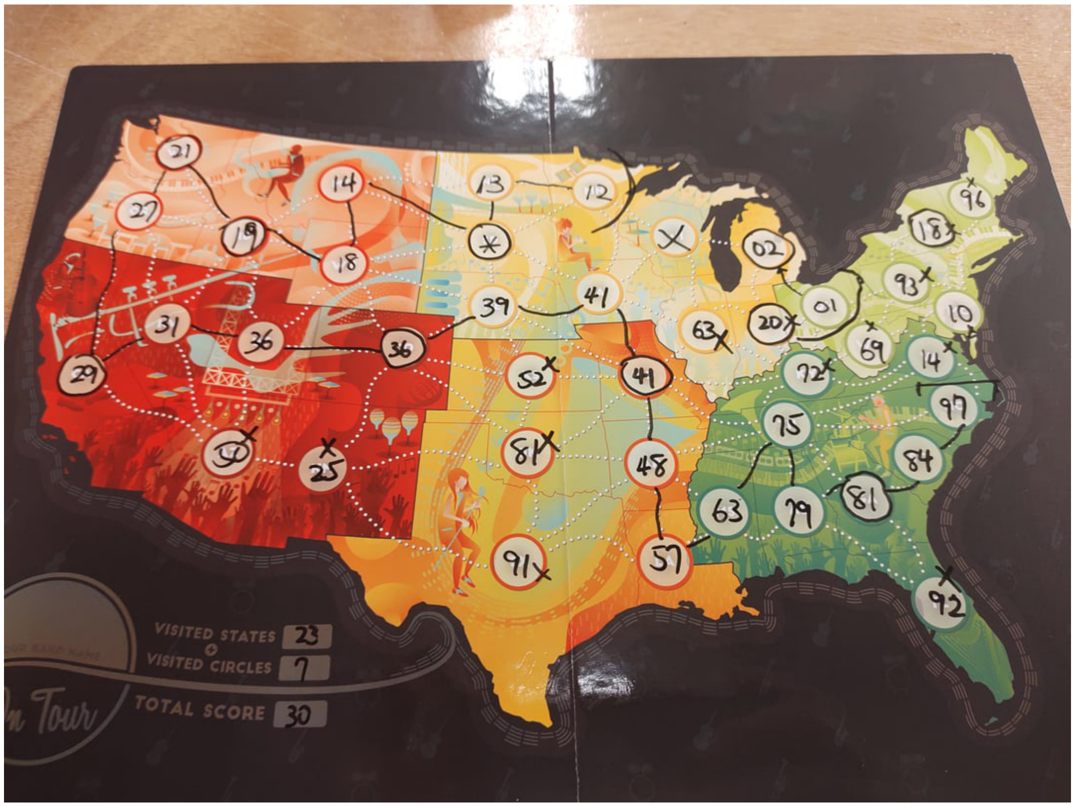
2 Problem Description

In the board game On Tour, players are given a map with various spaces that they will fill with numbers over the course of the game. One of the maps for the game consists of the US states, shown below:



At the end of the game, players will score points based on the longest “tour” they can form in which all of the numbers on the tour increase or stay the same. In addition, players have a chance to circle some of the spaces on the board, which makes these locations worth twice as many points as other locations. The player with the greatest number of points (total number of spaces connected, with an extra point for each circled location) wins the game.

An example of a completed game (with the scoring path highlighted) appears below:



This player created a path of length 23 that passed through 7 circled states, for a total of 30 points. Your algorithm should be able to take in information representing a completed game board and output the maximum possible score for the game as well as the path that achieves this score. In the example above, 30 points is indeed the highest score that this player could make.

In the game, some spaces may be marked as wild cards or unusable (* or X above). It turns out that if we allow wildcard spaces or adjacent spaces with the same number, this problem may potentially require an exponential amount of time to solve. Therefore, you only need to solve the restriction of this problem that does not allow wildcards or equal values for adjacent spaces. We can edit the example above to meet these restrictions by modifying the value of South Dakota (*) to 14, Montana (14) to 15, Colorado (circled 36) to 37, and Missouri (circled 41) to 42. In addition, you may assume that any unusable spaces (Wisconsin, in this example) are omitted.

4.1 Input format

Your program should read its input from the file input.txt, in the following format. The first line of the file will contain a positive integer n representing the total number of spaces on the board (in the case of the US map, 40, as many of the northeastern states plus Alaska and Hawaii don’t have spaces on the board).

The remaining n lines of the file will each contain information about one of the spaces on the board. Each line will begin with the numeric value filled in this space, followed by the number of points this space is worth (1 or 2 in the original game, though we can imagine variants where a space may be worth even more). These two will be followed by the number of neighbors for this space, d, as well as d integers representing the spaces to which it is connected. Each space on the board is identified by a positive integer 1 up to n, where space 1 is the first space described in input.txt (the second line of the file), while space n is the last. All of the values on the same line will be separated by spaces.

For example, in the example above, the player wrote 21 in the space representing Washington (WA) and circled it (making it worth 2 points). If this space is space 1, while Oregon (OR) and Idaho (ID) are spaces 6 and 7 respectively, the line representing Washington in this example would be:

```
21 2 2 6 7
```

Note that all of these numbers will be positive integers. There are no spaces worth 0 or negative points, and there are no isolated spaces on the map. In addition, while multiple spaces may have the same value, none of the spaces with the same value will be adjacent.

4.2 Output format

Your program should write its output to the file output.txt, in two lines. The first line should contain a single integer with the maximum possible score, while the second line should contain the values representing the maximum-scoring tour.

You have been provided with sample files that represent the example game given above, as well as a much smaller example with only 9 spaces on the board.