CCC '06 J4 - It's tough being a teen!

2006 Canadian Computing Competition, Stage 1, Junior #4

There is always a list of things to be done!

Here is a list for you left just this morning by your parental figure.

1. Do your Math homework.
2. Call your grandma.
3. Call me at work.
4. Call your friend.
5. Feed the dog.
6. Let the dog out.
7. Watch television.

(It is nice that your parental figure makes sure you watch television, and not just use the internet all day long.)

As well, your parental figure has given you constraints on these tasks:

- Do your Math homework BEFORE you watch Television.
- Do your Math homework BEFORE you call your friend.
- Call your grandma BEFORE you do your Math homework.
- Call me at work BEFORE you call your friend.
- Feed the dog AFTER you call me at work.

Your "to do" list (above) can now be abbreviated to:

- 1,7
- 1,4
- 2,1
- 3,4
- 3,5

where `x`, `y` indicates that the task numbered `x` should be done before the task numbered `y`.

Unfortunately, during the day additional instructions are emailed to you by your parental figure. Write a program to use your original "to do" list and the additional instructions to output a list of your jobs in the order you must do them, or alternately, if you cannot complete them, you should output that there is no way to complete these tasks, and you are just going to go to bed.

Input

You will be given pairs of numbers, one number per line, to represent the additional instructions to be included with your original "to do" list given above. The data terminates with the input pair 0 and 0. You can assume that there will be at most 10 additional constraints.

Output

You should output a list of tasks in the order that they should be performed, or an error message saying that the tasks cannot be completed. If there are multiple orders in which the tasks may be completed, the following tie-breaking rule must be used:

> If there is a set of tasks which may be performed at the same time during the process, the smallest numbered task should be performed first.

Sample Input 1

```
6
2
5
4
0
0
```

Sample Output 1

```
3 5 6 2 1 4 7
```

Explanation for Sample Output 1

The input data tells us that task 6 must be performed before task 2, and task 5 before task 4. The only tasks with no prerequisites are 3 and 6, so we choose 3 because it has the lower number. Then 5 and 6 are possible; 5 is chosen, then 6. Next must come 2, followed by 1. Then both 4 and 7 are possible; the lower one is chosen first.

Sample Input 2

```
7
2
4
5
0
0
```

Sample Output 2

```
Cannot complete these tasks. Going to bed.
```

Explanation for Sample Output 2

Notice that task 2 must be done before task 1, which must be done before task 7, which must be done before task 2. This forms a contradiction, and we cannot perform the tasks in the order prescribed.

CCC problem statements in large part from the PEG OJ