Advanced Topics in Programming - Exercises

Requirements and Guidelines

The exercises in the course would require you to implement a program which can solve jigsaw puzzles.

Exercise 1

In this exercise the program should get an input file which represents the different parts in the jigsaw puzzle - the format of the input file would be explained below.
The program should find the solution for the puzzle and create an output file representing the solution, as would be explained below.

Input File

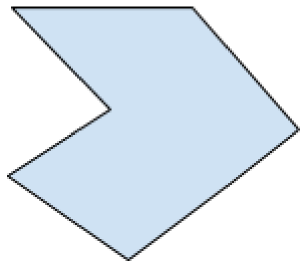
The input file is a text file in the following format:

The first line consists the following information:
NumElements=24
The word NumElements will appear as is, then any number of spaces (zero or more) then the sign '=' , then any number of spaces (zero or more) then a number which is the number of elements in the puzzle.

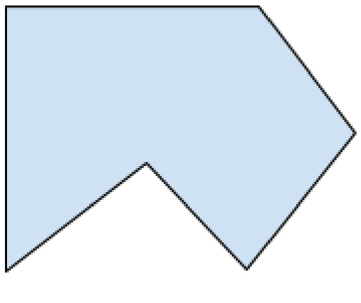
All other lines:
All other lines after the first line would have the following format:
1 -1 0 1 1
2 0 0 1 -1

The first number in each line represents the jigsaw element ID.
Then come 4 numbers representing the 4 sides of the element: left, top, right, bottom
0 means that the edge at this side is straight
1 means that the edge at this side is exterior (a "male")
-1 means that the edge at this side is interior (a "female")

The above lines mean:
Element number 1 which looks something like that:



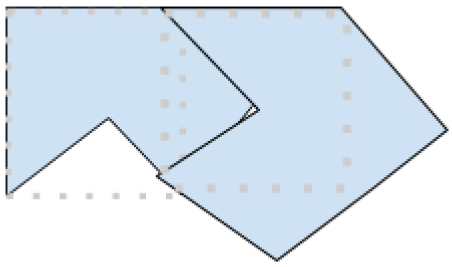
Element number 2 which looks like that:



Note:

1. The IDS of all elements must be in the range [1-NumElements].
2. The elements in file doesn't need to come in order - it might be that the line of element ID 5 will come after element ID 6 and it is OK.
3. In exercise 1 you <u>should NOT</u> rotate the elements in order to find a solution!
4. In exercise 1 elements can sit one near the other only at the same height (i.e. elements <u>should NOT</u> be shifted by 50% up or down).
5. The frame of the jigsaw (all the edges) must be straight!

For example, elements 1 and 2 in the above example can sit one near the other only in one option:



In the above example - element 2 may be the Top-Left corner of the puzzle.
BUT - it can also be at any other place in the puzzle - it is legitimate to put another element to the left of element 2, as long as it has a straight right edge.

(In next exercise, element 2 could also become any other corner of the puzzle by rotation).

<u>Output File</u>
The output file should be the IDs of the elements in the structure that solves the puzzle:
A single space between IDs on the same line, a new line for each row in the puzzle.
You should put the first solution that you find (no need to seek for additional solutions!)

<u>Errors</u>
If there were errors while reading or analyzing the input file, the output file shall have a list of all errors, a line per each error, you must list all relevant errors, <u>in the order below</u>!

Errors may include the following:

- <u>Missing elements in the input file, required error should be in this exact format:</u>
  `Missing puzzle element(s) with the following IDs: <comma separated list of IDs, sorted up, e.g.: 3, 5, 6, 12>`
- <u>Wrong element IDs (e.g. puzzle of size 6 cannot have an ID 7):</u>
  `Puzzle of size <SIZE> cannot have the following IDs: <comma separated list of wrong IDs>`
- <u>Wrong elements format (not having 4 edges, having edges which are not 0, 1, -1):</u>
  `Puzzle ID <id> has wrong data: <complete line from file including ID>`
  <u>Note</u>: (i) a line per each wrong element; (ii) having more than 4 edges is an error!
- <u>Cannot solve puzzle:</u>
```
Cannot solve puzzle: wrong number of straight edges
Cannot solve puzzle: missing corner element: <TL><TR><BL><BR>
Cannot solve puzzle: sum of edges is not zero
Cannot solve puzzle: it seems that there is no proper solution
```

Additional notes:

[1]

```
Cannot solve puzzle: missing corner element: <TL><TR><BL><BR>
```

If you miss more than one corner, list all missing corners, each with its own line in the order `<TL><TR><BL><BR>`

For example suppose that there are missing **TL**, **BL** and **BR**, you should print to file:

```
Cannot solve puzzle: missing corner element: TL
Cannot solve puzzle: missing corner element: BL
Cannot solve puzzle: missing corner element: BR
```

[2]

```
Cannot solve puzzle: it seems that there is no proper solution
```

This error can come as the only error - it will never come with any other error as it means that all other things seem OK, but we just tried and couldn't solve it.

This error should be given only if you exhausted all combinations without finding a solution.

[3]

```
Cannot solve puzzle: missing corner element: <TL><TR><BL><BR>
```

Is there also an error for "too many corner elements"?

- NO. It is legitimate to put a corner element at any point in the puzzle, as long as the edges of the adjacents elements fit.
  For example, there can be a puzzle with all elements being straight on all 4 edges.

[4]

How can we know how many rows are in the puzzle?

- You cannot just "know it". It's part of the puzzle of the puzzle (in the meaning of: part of the challenge, the riddle, i.e. your program should come out with that number somehow)

[5]

Combinations of:

```
Cannot solve puzzle: wrong number of straight edges
Cannot solve puzzle: missing corner element: <TL><TR><BL><BR>
Cannot solve puzzle: sum of edges is not zero
```

The above errors can come together.

In case all the above happen, you should print all the above errors. In the above order.

Note that wrong number of straight edges doesn't necessarily mean that sum of edges is not zero, and vice versa. It can come together. Or not.

[6]

```
Cannot solve puzzle: wrong number of straight edges
```

What is the 'right' number of straight edges?

- Well, this is also part of the puzzle of the puzzle (in the meaning of: part of the challenge, the riddle, i.e. your program should come out with that number somehow, or at least note that a certain number of a certain straight edge is not good).
- Note that it is legitimate to use a straight edge inside the puzzle, as long as there is an adjacent element with a fitting straight edge (i.e. it is not mandatory to use straight edges only for the frame).