Downloads

- Executable JAR of the visualizer
- Source code of the visualizer

In order to use the offline tester / visualizer tool for testing your solution locally, you'll have to modify your solution by adding the main method that interacts with the tester / visualizer via reading data from standard input and writing data to standard output. Since you do not change the implementation of method *makeConnected*, this doesn't affect the way your solution works when being submitted to our server.

To simulate a single test case, your program should read the following data from standard input in this exact order (each integer/string is located in a separate line):

- Integer *SZ*.
- Strings **board**[0], **board[1]**, ..., **board**[*SZ* - 1], in this order.

Once this data is read, pass it to your implementation of *makeConnected* method and let *ret* be the return value. Write the following data to standard output in this exact order (each integer/string in a separate line):

- Integer *len* denoting the number of elements in *ret*.
- Strings *ret*[0], *ret*[1], ..., *ret*[*len*-1], in this order.

Make sure to flush the standard output after all this data is written.

In other words, you should implement the following pseudocode in the main method of your solution:

```
SZ = parseInt(readLine())

for (i=0; i < SZ; i++)
    board[i] = readLine()

ret = makeConnected(board)

printLine(length(ret))
for (i=0; i < length(ret); i++)
    printLine(ret[i])

flush(stdout)
```

In order to run the tester / visualizer, you should use the following command:

```
java -jar BlackAndWhiteVis.jar -exec "<command>"
```

`<command>` is the command you would use to execute your solution. If your compiled solution is an executable file, the command will just be the full path to it, for example, "C:\TopCoder\solution.exe" or "~/topcoder/solution". In case your compiled solution is to be run with the help of an interpreter, for example, if you program in Java, the command will be something like "java -cp C:\TopCoder Solution".

Additionally you can use the following parameters (all are optional):

- `-seed <seed>`. Sets the seed used for test case generation. Default value is 1.
- `-novis`. Switches the visualization off, leaving only text output.
- `-manual`. Turns on the manual play mode. To make a move, you can click a cell outside the board neighboring to row/column that you'd like to shift in direction opposite to shift direction. For example, in order to shift a row to the right, click the cell directly to the left of this row.

- `-info`. Turns on the debug mode. This way you will see more information in standard output of the visualizer.
- `-delay`. Sets the delay between visualization of consecutive shifts, in milliseconds. The default value is 100.
- `-side`. Sets the size of one cell of the board, in pixels. The default value is 9.

You can print any debug information of your solution to the standard error stream and it will be forwarded to the standard output of the tester.

For more information on using visualizers, please check the following recipe draft from TopCoder Cookbook. Note that this is not a troubleshooting thread, please use the match forum for questions instead.