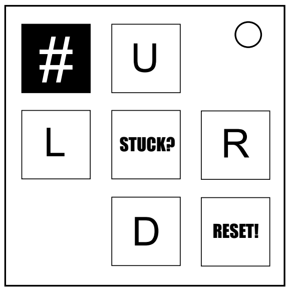
On the Subject of Cruel Boolean Maze



Hey, you guys wanted a wraparound maze right?

This module contains four movement keys **U,L,R,D**, a **STUCK?** key, a **RESET!** key and a display which will display a random integer between 0 and 3. If the buttons on the module are not red then you are looking at a different module.

Tips for Success

- In order to solve this module, travel from the starting point to the ending point.
  - Starting Location: (3rd,4th) positions of the serial
  - Ending Location: (5th,6th) positions of the serial
- Make sure to convert letters to numbers (A = 1, B = 2, ...) and take their value modulo 10.
- The starting and ending locations will be in (row,column) format, with the top left space of the maze being (0,0).
- Convert the number on the display to 2-digit binary.
- Use **U,L,R,D** to move Up, Left, Right, and Down respectively.
- A move is considered legal only if, upon using the two digits of the display in binary as the two inputs for the logic gate in the adjacent space, the result would be 1. (See Appendix A)
- If you attempt to enter a space, and the logic gate would return a 0, you will receive a strike and you will not be moved.
- You **may** leave the edges of the maze. You will wrap around to the other side of the maze.
- If you have no legal moves you can press **STUCK?** to change the display until you can move again, but be careful, using this when you have a legal move will result in a strike and you will be reset back to the start.
- If you think you may be lost you can press **RESET!** to reset back to the starting position.
- Before making your first move, determine your position in the **Not Grid.**
  - Starting Location in Not Grid: (1st,2nd) positions of the serial
- Make sure to convert letters to numbers (A = 1, B = 2, ...) and take their value modulo 5.

Tips for Success continued

- When you make legal moves, your position will be updated in the **Not Grid** (wrapping around as necessary). If you attempt to make a move in the maze, and your current position contains the phrase **NOT**, then you must invert the logic gate you are attempting to enter. (See Appendix B)
- If the **Not Grid** contains anything else in the current cell, then you can ignore it and make your move as normal.
- **NOTE:** If the Ending Location lands on a NOR or an AND, then it will shift 1 cell at a time until it is no longer a NOR/AND. The direction of the shift depends on the original displayed number (0 = UP, 1 = RIGHT, 2 = DOWN, 3 = LEFT). If the shift reaches the edge of the grid it will wrap around to the other side of the grid.

Maze

	0	1	2	3	4	5	6	7	8	9
0	NOR	XOR	OR	AND	OR	AND	XOR	NAND	OR	XOR
1	XOR	AND	OR	NAND	OR	OR	OR	AND	XOR	NAND
2	OR	XNOR	OR	OR	XOR	NOR	OR	AND	OR	XNOR
3	AND	NAND	OR	NOR	OR	XOR	AND	NOR	OR	OR
4	OR	XNOR	AND	OR	NAND	NOR	OR	OR	NOR	XOR
5	XOR	OR	NAND	NOR	OR	OR	AND	NOR	XOR	OR
6	OR	OR	AND	NOR	OR	AND	XOR	OR	OR	XOR
7	XOR	XNOR	OR	XNOR	OR	XOR	XNOR	XNOR	NAND	OR
8	XOR	OR	OR	OR	NAND	XNOR	NOR	NAND	OR	XOR
9	OR	XNOR	XOR	XNOR	AND	OR	XOR	OR	AND	NOR

Not Grid

	0	1	2	3	4
0	NOP	NOT	NOT	NUP	NUT
1	NAT	NOT	NPT	NIT	NOT
2	NOT	NAT	NET	MOT	NUT
3	NUP	NOP	NUT	NOT	NOT
4	NOT	NOT	NUT	MOT	NOT

Appendix A

Decimal	Binary	Logic Gates					
		NOR	XOR	OR	AND	XNOR	NAND
0	00	1	0	0	0	1	1
1	01	0	1	1	0	0	1
2	10	0	1	1	0	0	1
3	11	0	0	1	1	1	0

Appendix B

Logic Gate	Inverted Logic Gate
NOR	OR
XOR	XNOR
AND	NAND
OR	NOR
XNOR	XOR
NAND	AND