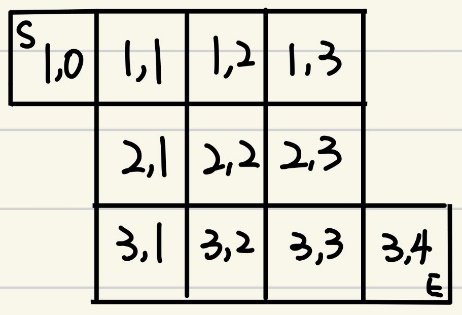
# Project Summary

*This game is about connecting pipes in a 3x3 grid, where the start is in the top left corner and the end is in the bottom right corner. This game aims to connect the starting pipe, the ending pipe, and every pipe in between them (there are 5 types of pipes) by rotating the pipes on the grid. The water can only flow forward and downward. We made a simple diagram to get a better idea of ​​what the entire grid looks like. We will simulate how pipes are placed, and how they are connected in the end.*

# Propositions

* Location (p, loc): There is a pipe with a certain pipe orientation at location loc. There are 11 locations available for a 3x3 grid and 3 pipes with a total of 12 pipe orientations.
* NeighborLR (l1, l2): location l1 and l2 is neighbor and they are beside each other
* NeighborUD (l1, l2): location l1 and l2 is neighbour and one is on top of the other
* Connected (l1, l2): location l1 and l2 is connected
* Have\_to\_east (loc): the pipe on loc has an opening facing east
* Have\_to\_south (loc): the pipe on loc has an opening facing south
* Have\_from\_west (loc): the pipe on loc has an opening facing west
* Have\_from\_north (loc): the pipe on loc has an opening facing north

# Constraints

* In a grid, only one pipe with exactly one config can exist.

¬ location(pipe\_type1, loc) \/ ¬ location(pipe\_type2, loc)

* Two faces are connected if they are each adjacent and have lines facing each other.

Neighbor ()Connect ()

* We need at least one solution to exist, and we can stop when checking for a feasible route.
* Win condition: A transitive connection from the starting pipe to the ending pipe

Connected(start)Connected1Connected2 … Connected(end)

* There is only one possible pipe type for the start(E) and one for the endpoint (W).
* The pipe type of the start and end points can only be on grids “01” and “34” respectively.
* Pipe\_connect: after we randomize the setup, check to see if each grid is connected.
* Grid setup: The pipe type of the setup on each grid can be in a different direction, but the type cannot be changed.
* There is exactly one pair of pipes connected from the neighbor left and right / up and down.
* The start and end pipes can not be connected directly.
* There are constraints on the type of pipe next to the start and endpoints. They can't be in a situation where can't connect.

One location:

* There is only one pipe orientation of one pipe at location 10 and location 34(the start and end piece)

the opening of the start piece at location 10 can only face east

the opening of the end piece at location 34 can only face west

* All other locations (locations that are not 10 and 34) will not have opening only facing east or only facing west
* If there is a certain pipe at a location, then that location can have a different pipe orientation at the same location but not a different pipe

(ex. If there is a straight pipe (orated in EW or NS) at location 11, then location 11 can have exactly one pipe orientation from (EW, NS). But the other 2 pipes in a total of 8 pipe orientations can not be at location 11)

* For all locations except for 22, 10, and 34, there is one pipe orientation from one pipe depending on their location and pipe on them.

(If there is a 3-opening pipe at location 11, it must be the pipe orientation SEW. 3-openning-pipe have 4 orientations where only 1 orientation can both down and right. Since if a solution route is going through grid cell 11, it needs to connect to the grid cell whether on the right or down. This exact pipe orientation contains all the

Two locations:

* If locations differ in exactly 1 or 10, they will be neighbor

(ex. 10 and 11 will be NeighborLR, and 11 and 12 will be NeighborUD)

* Every location does not differ in 1 or 10 will not be neighbor

(ex. 10 and 34 is not NeighborLR or NeighborUD)

* If two locations are not neighbors, they are not connected (ex. 10 and 34 is not connected)
* No connection upward or left.

(Connected(l1,l2) **~** Connected(l2,l1) where l2>l1)

# Model Exploration

For one of our constraint pipe\_type, we write a nested for loop to find every possible figure a pipe can have (like [‘N’,’ W’], but the elements in there should not be the same). Nested for loop for i, for j, for k when running giving back something like [‘N’,’ W’,’ W’] which is not the expectation. Then we realized something was wrong in the j and k loops since they are repeated ones. After correcting the staring value in loops from (0,i+1,i+2) to (0,i+1,j+1). The previous nested loop goes over the second part of the array twice which causes j and k loops to form the same element.

We have this constraint connected which generates with for loop what kind of a pair of pipe\_type can be connected like (pipe with opening to E and opening EW). We found it returned [[‘E’] [‘W’]] which means that the start and end are connected which is not possible. Since that is the only case that is possible in the whole generated array, we just delete that from the array

# Jape Proof Ideas

1. When there are left and right connected grids, it may not be necessary to look at the up and down of the left grid to see whether they are connected or not.
2. No need to consider optimal solutions.First-Order Extension

*Describe how you might extend your model to a predicate logic setting, including how both the propositions and constraints would be updated.* ***There is no need to implement this extension!***

# Useful Notation

*Feel free to copy/paste the symbols here and remove this section before submitting.*