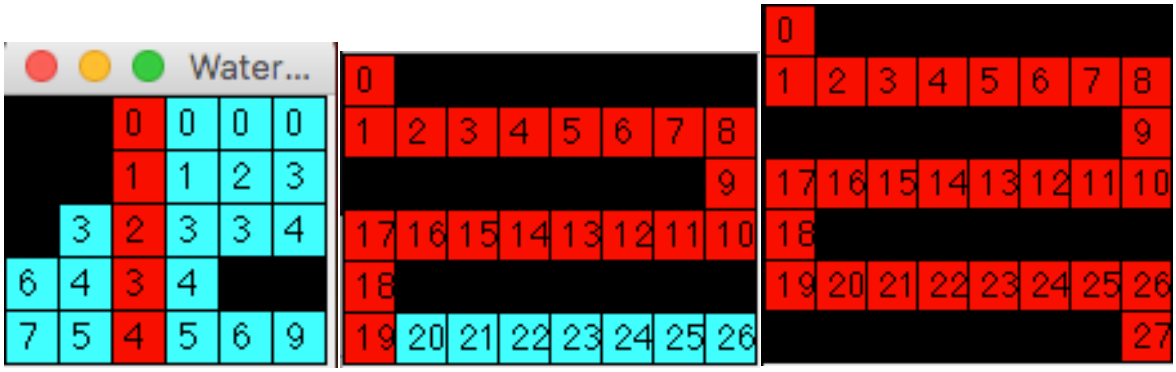
Analysis Questions

1. Consider a flowGrid of size n by m grid cells. What is the *minimum* number of cells that an earliest flow path can have? Show and explain what such a path would look like. What is the *maximum* number of cells an earliest flow path can have? Show (include a visual) and explain what such paths would look like. Express all bounds in terms of n and m.

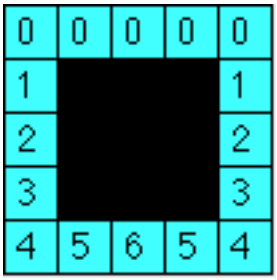
The minimum number of cells would be n. considering that it is rows x columns.
The shortest path is always the straight path



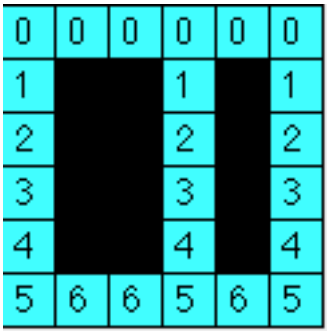
Longest path would be:

m(n/2 – 1) + n for even number
or m(n/2) + n for odd number

2. Choose a cell in row 0 and one in row n-1. What is the maximum number of cell-disjoint flow paths that can exist between the two cells? Cell-disjoint means that no two cell on different paths touch in a way that allows a flow from one path to the other. Explain and illustrate your answer.
   If the cell is in the corner, there can be a maximum of 2 paths.



   If the cell is NOT in the corner, there can be a maximum of 3 paths.



3. Explain and justify the choice of the data structure used to determine the flow.

   For this Project, I used and Array of Linked Lists. Linked lists are easy to implement and the elements can be added to the front or the back of linked lists.
   Their complexity is extremely low and can store data linearly.

4. In terms of n and m and in the asymptotic sense, what is the running time of your algorithm (worst case)?

   First, the code scans the first row. The complexity of scanning is m.
   Then, the worst case scenario for the code would be *m(n/2 – 1) + n or m(n/2) + n.*
   Which simplifies to O(mn/2)
   Which simplifies to
   O(mn)

5. Assume the flow values are 0 and 1 only and n=m (for simplicity) . During method DetermineFlow, what is the maximum number of cells that can be in your chosen data structure at a time t? Explain your answer in terms of n and show a grid that achieves this maximum. State at what time it happens.
   The maximum number of cells at a given time T would be n.
   Since my data structure stores the cells according to the time that they're supposed to be updated, n is the maximum number of cells.

0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4

EXPERIMENTAL

1. **200 x 200**

Probabilty	Flowed	Didn't Flow
0.2	100	0
0.3	100	0
0.4	9	91
0.5	0	100
0.6	0	100
0.7	0	100

2. **200 x 50**

Probabilty	Flowed	Didn't Flow
0.2	100	0
0.3	100	0
0.4	0	100
0.5	0	100
0.6	0	100
0.7	0	100

50 x 200

Probabilty	Flowed	Didn't Flow
0.2	100	0
0.3	100	0
0.4	96	4
0.5	0	100
0.6	0	100
0.7	0	100

At increasing probability of a cell being blocked, the number of flows would decrease because that would leave less space for the cell let the value into the next cell.

At 50x200 The number of flows was higher than the 200x200 or 200x50 because the rows are less. The lesser the rows, the lesser would be the chances of it getting blocked.