Virus Infection

One hundred computers are connected in a 10x10 network grid, as below. At the start exactly nine of them are infected with a virus. The virus spreads like this: if any computer is directly connected to at least 2 infected neighbors, it will also become infected.

Will the virus infect all 100 computers?

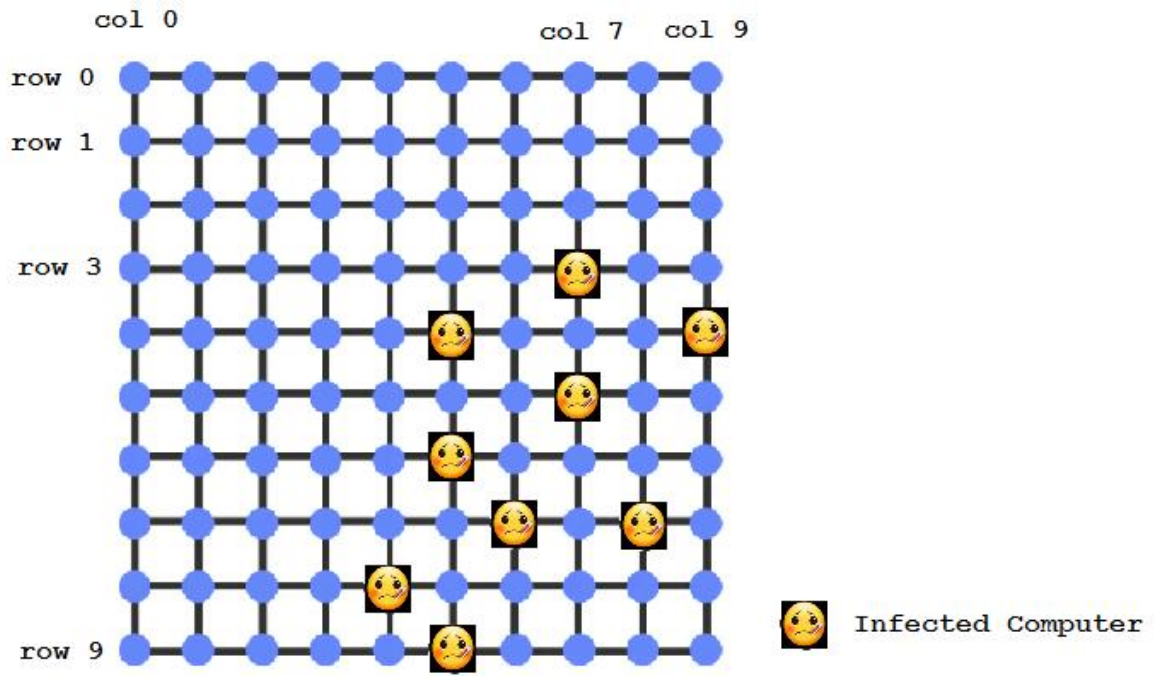

Figure 1

The image shows a possible example of the initial infection. You can try to fill it in to see if ultimately the network will consist of 100 orange dots. But the question is not asking what happens to this example. I want to know what will happen given any initial configuration of infected computers.
It's a lovely question - or rather, the solution is lovely. (And there is a connection to pi, but not an obvious one)

In this problem you will complete three different methods in the `VirusInfection` class. The `VirusInfection` class has `boolean[][] nw` as its instance variable. `nw` represents the network grid of computers in its original configuration. If `nw[row][col] == true` the computer at location `row`, `col` is **NOT** infected. If `nw[row][col] == false` the computer at location `row`, `col` is infected. In the Figure 1 above, the following locations list the nine computers that are infected:

- row =3 and col = 7
- row =4 and col = 5
- row =4 and col = 9
- row =5 and col = 7
- row =6 and col = 5
- row =7 and col = 6
- row =7 and col = 8
- row =8 and col = 4
- row =9 and col = 5

Problem from
https://www.theguardian.com/science/2017/mar/13/can-you-solve-it-pi-day-puzzles-that-will-leave-you-pie-eyed

Note: You may assume every grid will be a rectangle. That is

```
For integers m, n: 0 <= m, n < nw.length, nw[m].length == nw[n].length
```

The first method to implement is the `isSafeLocation(int row, int col)` method. `isSafeLocation` returns `true` if both:

- `nw[row][col] == true` (the current location is **not** infected) and
- less than 2 of its neighbors are infected (false)

Conversely, a location is not safe if either:

- `nw[row][col] == false` (the current location is infected) or
- two or more of its neighbors are infected (true)

In addition:

- You may assume `nw[m].length == nw[n].length, 0 <= m, n < nw.length`
- You may assume `0 <= row < nw.length`
- You may assume `0 <= col < nw[row].length`
- You must NOT modify the instance variable `nw`.

The following code shows the results of the `isSafeLocation` method.

The following code	Returns

`final boolean T = true;`
`final boolean F = false;`
 	
`boolean[][] net = { {T, T, T, F },`
`{T, F, F, T },`
`{F, T, F, T },`
`{T, T, T, F } };`
 	
`VirusInfection v = new VirusInfection(net);`	

`v.isSafeLocation(0, 1);`	`true`
`v.isSafeLocation(1, 1);  //  infected`	`false`
`v.isSafeLocation(2, 1);  //  2 neighbors infected`	`false`

Problem from
https://www.theguardian.com/science/2017/mar/13/can-you-solve-it-pi-day-puzzles-that-will-leave-you-pie-eyed

The second method to implement is the `spreadVirus(int num)` method. `spreadVirus` returns a `boolean[][]` which represents the result of the virus spreading for `num` iterations. An iteration consist of marking each computer infected if two or more of its neighbors are infected in the original configuration.
Each additional iteration consist of marking each computer infected if two or more of its neighbors are infected after the previous iteration has terminated.
Remember:

- You must NOT modify `nw`.
- You may assume `num > 0`
- Please note, the following three examples demonstrate three consecutive iterations with the same network grid of computers.

The following code shows the results of the `spreadVirus` method with parameter num = 1.

The following code	Returns

`final boolean T = true;`
`final boolean F = false;`
 	
`boolean[][] net = { {T, T, T, F },`
`{T, F, F, T },`
`{F, T, F, T },`
`{T, T, T, F } };`
 	
`VirusInfection v = new VirusInfection(net);`
 	
`boolean[][] ans = v.spreadVirus(1);`	

`ans[0][2] || ans[0][3] || ans[1][0] || ans[1][1] ||`
`ans[1][2] || ans[1][3] || ans[2][0] || ans[2][1] ||`
`ans[2][2] || ans[2][3] || ans[3][2] || ans[3][3]`
 	
• This implies that the 12 locations are all infected
	`false`

`ans[0][0] && ans[0][1] && ans[3][0] && ans[3][1]` • This implies that the 4 locations are all **not** infected	`true`

Problem from
https://www.theguardian.com/science/2017/mar/13/can-you-solve-it-pi-day-puzzles-that-will-leave-you-pie-eyed

The following code shows the results of the `spreadVirus` method method with parameter num = 2.

The following code	Returns

`final boolean T = true;`
`final boolean F = false;`
 	
`boolean[][] net = { {T, T, T, F },`
`{T, F, F, T },`
`{F, T, F, T },`
`{T, T, T, F } };`
 	
`VirusInfection v = new VirusInfection(net);`
 	
`boolean[][] ans = v.spreadVirus(2);`	

`ans[0][1] || ans[0][2] || ans[0][3] || ans[1][0] ||`
`ans[1][1] || ans[1][2] || ans[1][3] || ans[2][0] ||`
`ans[2][1] || ans[2][2] || ans[2][3] || ans[3][1] ||`
`ans[3][2] || ans[3][3]`
• This implies that the 14 locations are all infeted
	`false`

`ans[0][0] && ans[3][0]` • This implies that the 2 locations are all **not** infected	`true`

The following code shows the results of the `spreadVirus` method with with parameter num = 3.

The following code	Returns

`final boolean T = true;`
`final boolean F = false;`
 	
`boolean[][] net = { {T, T, T, F },`
`{T, F, F, T },`
`{F, T, F, T },`
`{T, T, T, F } };`
 	
`VirusInfection v = new VirusInfection(net);`
 	
`boolean[][] ans = v.spreadVirus(3);`	

`ans[0][0] || ans[0][1] || ans[0][2] || ans[0][3] ||`
`ans[1][0] || ans[1][1] || ans[1][2] || ans[1][3] ||`
`ans[2][0] || ans[2][1] || ans[2][2] || ans[2][3] ||`
`ans[3][0] || ans[3][1] || ans[3][2] || ans[3][3]`
• This implies that the 16 locations are all infected
	`false`

• No locations are **not** infected.	

Problem from
https://www.theguardian.com/science/2017/mar/13/can-you-solve-it-pi-day-puzzles-that-will-leave-you-pie-eyed

The third method to implement is the `infectAll` method. `infectAll` returns a `boolean` indicating if all the computers in the network grid of computers became infected with the virus.
Remember:

- You must NOT modify `nw`.

The following code shows the results of two different calls to the `infectAll` method.

The following code	Returns

`boolean[][] net = { {T, T, T, F },`
`{T, F, F, T },`
`{F, T, F, T },`
`{T, T, T, F } };`
 	
`VirusInfection v1 = new VirusInfection(net);`	

`v1.infectAll());`	`true`

`final boolean T = true;`
`final boolean F = false;`
 	
`boolean[][] net0A = { {T, F, T, T },`
`{T, T, F, T },`
`{T, T, T, F } };`
 	
`VirusInfection v2 = new VirusInfection(net0A);`	

`v2.infectAll();`	`false`

Problem from
https://www.theguardian.com/science/2017/mar/13/can-you-solve-it-pi-day-puzzles-that-will-leave-you-pie-eyed