Covid Infection/rotten tomatoes

'''Q. Aterp is the head nurse at city hospital.
City hospital contains RC *number of wards and structure of hospital is in*
the form of a 2-D matrix.
*Given a matrix of dimension R*C where each cell in the matrix can
have values 0, 1 or 2 which has the following meaning:
0 : Empty ward
1 : Cells have uninfected patients
2 : Cells have infected patients
An infected patient at ward [i,j] can infect other uninfected patient at indexes:
[i-1,j]
[i+1,j]
[i,j-1]
[i,j+1]
(up, down, left and right) in unit time.
Help Aterp determine the minimum units of time after which there won't
remain any uninfected patient i.e all patients would be infected.
If all patients are not infected after infinite units of time then simply return -1.'''

```
def helpaterp(hospital):
    neg = set()
    pos = set()

    m = len(hospital)
    n = len(hospital[0])

    for i in range(m):
        for j in range(n):
            if hospital[i][j] == 2:
                pos.add((i, j))
            elif hospital[i][j] == 1:
                neg.add((i, j))

    mins = 0
    dirs = [(1, 0), (-1, 0), (0, 1), (0, -1)]

    while len(neg) > 0:
        infected = set()

        for r in pos:
```

~~~~python
            row = r[0]
            col = r[1]

            for dir in dirs:
                new_row = dir[0] + row
                new_col = dir[1] + col

                if (new_row, new_col) in neg:
                    neg.remove((new_row, new_col))
                    infected.add((new_row, new_col))

        if len(infected) == 0:
            return -1

        pos = infected
        mins += 1

    return mins


hospital =  [[2, 1, 0, 2, 1],
            [1, 0, 1, 2, 1],
            [1, 0, 0, 2, 1]]

print(helpaterp(hospital))
``````Python
def orangesRotting(self, hospital: List[List[int]]) -> int:
        if len(hospital)==1:
            if len(hospital[0])==1:
                return -1 if hospital[0][0]==1 else 0
        m = len(hospital)
        n = len(hospital[0])
        queue = []
        ones = 0
        for i in range(m):
            for j in range(n):
                if hospital[i][j] == 2:
                    queue.append((i, j))
                if hospital[i][j]==1:
                    ones=ones+1
        mins = 0
        dirs = [(1, 0), (-1, 0), (0, 1), (0, -1)]
~~~~

```
        while len(queue) and ones>0:
            length = len(queue)
            mins = mins+1
            while length:
                row, col = queue.pop(0)
                for dir in dirs:
                    new_row = row+dir[0]
                    new_col = col+dir[1]
                    if self.isValid(new_row, new_col, m, n) is True and
hospital[new_row][new_col] == 1:
                        ones=ones-1
                        hospital[new_row][new_col] = 2
                        queue.append((new_row, new_col))
                length = length - 1

        return mins if ones==0 else -1

    def isValid(self,row,col,R,C):
        if row>=0 and col>=0 and row<R and col<C:
            return True
        return False
```