

Tanish Mittal
IT-C
2000290130174

Problem 6: You will be given an $m \times n$ grid, where each cell has the following values :

1. 2 – represents a rotten orange
2. 1 – represents a Fresh orange
3. 0 – represents an Empty Cell

Every minute, if a Fresh Orange is adjacent to a Rotten Orange in 4-direction (upward, downward, right, and left) it becomes Rotten.

Return the minimum number of minutes required such that none of the cells has a Fresh Orange. If it's not possible, return -1.

```
from collections import deque
```

```
def orangesRotting(grid):
```

```
    directions = [(0, 1), (0, -1), (1, 0), (-1, 0)]
```

```
    queue = deque() fresh_oranges =
```

```
    0
```

```
    minutes = 0
```

```
    for i in range(len(grid)):
```

```
        for j in range(len(grid[0])):if
```

```
            grid[i][j] == 2:
```

```
                queue.append((i, j))elif
```

```
            grid[i][j] == 1:
```

```
                fresh_oranges += 1
```

```
    while queue:
```

```

size = len(queue)
rotten_found = False
for _
in range(size):
    x, y = queue.popleft()

    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < len(grid) and 0 <= ny < len(grid[0]) and grid[nx][ny] == 1:
            grid[nx][ny] = 2
            fresh_oranges -= 1
            queue.append((nx, ny))
            rotten_found = True

    if rotten_found:
        minutes += 1

if fresh_oranges > 0:
    return -1
else:
    return minutes
grid = [[2, 1, 1], [0, 1, 1], [1, 0, 1]]


print(orangesRotting(grid))

```

Jobs new

Sign Up

Login



GOT AN OPINION

About • FAQ • Blog •
Terms of Use • Contact
Us • GDB Tutorial •
Credits • Privacy
© 2016 - 2023 GDB

```
41 print(orangeRotting(grid))
42
```

input

```
-1
...Program finished with exit code 0
Press ENTER to exit console.
```