Heat map cellular simulation - bug fix

The problem with the heat map code is that it moves people in a particular order, going over the grid left to right, top to bottom. The fix for this is described below. This fix is implemented in the current live version of the 'Heat map cellular simulation' page.

Imagine Person A left of Person B, with Person B closer to the exit. Person A is considered first, and they are to move into Person B's place. Since there is no one in that space in `nextgrid`, Person A is placed there. Now if Person B can move on, all is fine. In the event that Person B discovers their way blocked, the code keeps them where they are. The problem is that Person A is already there, and so the code writes Person B over Person A, effectively deleting one person from the simulation.

One way to fix this is to consider the squares nearest the exit first, moving those people if possible, then move away from the exit until the whole grid is dealt with.

To do this, first we need to know the maximum heat map number (apart from any walls). We can record this information where we are building the `heatmap`. We fill in the heatmap in the `while heatmapIncomplete:` loop. This uses a variable `count` to keep track of how far away from any exits we are. Immediately after this loop, record the end value of `count` like this:

```
heatmap_max = count
```

Now inside the `while run:` loop there is a nested `for` loop where we move the people. We put this into a `for` loop which counts the distance from the exit from `0` up to `heatmap_max` using a loop variable `h`. Each square is only considered for movement if the heatmap value matches the current value of `h` being consider, i.e. we have added `if heatmap[r][c] == h:`. That block of code now looks like this:

```
    for h in range(0,heatmap_max):
        for r in range(len(grid)):
            for c in range(len(grid[0])):
                if heatmap[r][c] == h:
                    if grid[r][c] == 1 and heatmap[r][c] > 1: # person not by
                        min_r = r
                        min_c = c
                        if r-1>0 and heatmap[r-1][c] < heatmap[r][c]:
                            min_r = r-1
                            min_c = c
                        if c-1>0 and heatmap[r][c-1] < heatmap[r][c]:
                            min_r = r
                            min_c = c-1
                        if r+1<50 and heatmap[r+1][c] < heatmap[r][c]:
                            min_r = r+1
                            min_c = c
                        if c+1<70 and heatmap[r][c+1] < heatmap[r][c]:
                            min_r = r
                            min_c = c+1
                        if nextgrid[min_r][min_c] == 0:
                            nextgrid[min_r][min_c] = 1
```

```
                else:
                    nextgrid[r][c] = 1
```