Austin Dollar
4/22/21
CSCI 344
Lab 11/12

Python Tkinter/GUI

Code (copied/pasted for readability):

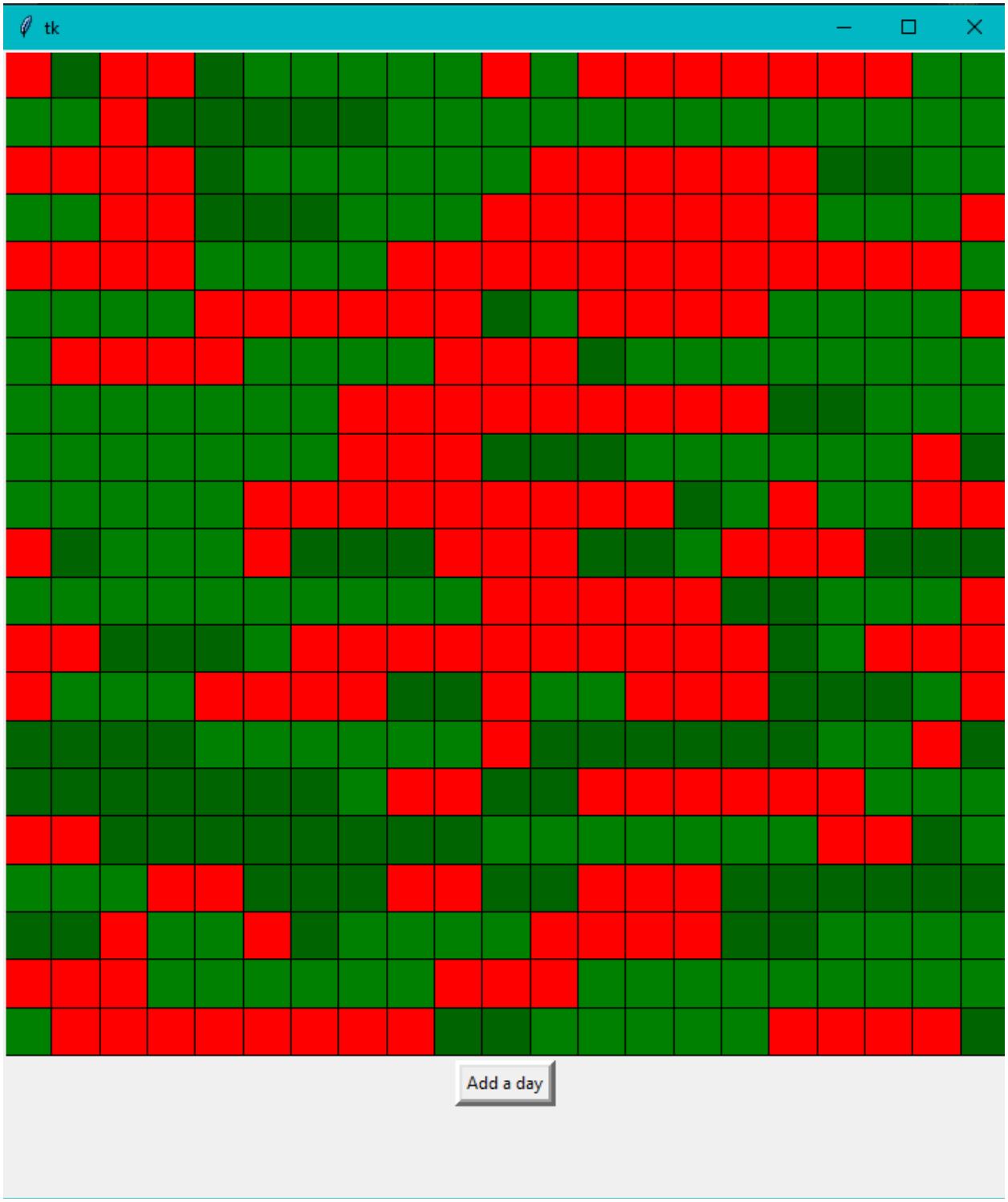
```
#!/usr/bin/python3.5
print("Starting Simulation...")

from tkinter import *
import tkinter as tk
import random

winH = 700
winW = 700
ncols = 21
nrows = 21
cellW = winW / ncols
cellH = winH / nrows
status = 2

class Node:
    def __init__(self, row, col):
        self.row = row
        self.col = col
        return

def generatGrid(nrows, ncols):
    grid = []
    for r in range(nrows):
        row = [ Node(r, c) for c in range(ncols) ]
        grid.append(row)
    return grid

def drawNode(canvas, node):
```

```
    x1 = cellW * node.col
    y1 = cellH * node.row
    x2 = x1 + cellW
    y2 = y1 + cellH
    my_rectangle = canvas.create_rectangle(x1, y1, x2, y2, fill = "green")
    return

def event(canvas, node):


    btn = Button(window, text = 'Add a day', bd = '5',command=lambda:
handler(canvas, node))
    btn.pack(side = 'top')

def handler(canvas, node):
    global status
    for row in grid:
        for node in row:
            x1 = cellW * node.col
            y1 = cellH * node.row
            x2 = x1 + cellW
            y2 = y1 + cellH


            if status == 0:
                num = random.randint(0,100)
                if num <= 35:
                    status = 1


            if status == 1:
                num = random.randint(0,100)
                if num <= 5:
                    status = 0
                if num <= 40:
                    status = 2
```

```
            if status == 2:
                num = random.randint(0,100)
                if num <= 5:
                    status = 3
                if num <= 15:
                    status = 1
                if num <= 20:
                    status = 0

            if status == 3:
                num = random.randint(0,100)
                if num <= 10:
                    status = 0



            if status == 0:
                canvas.create_rectangle(x1, y1, x2, y2, fill = "red")
            elif status == 1:
                canvas.create_rectangle(x1, y1, x2, y2, fill = "dark green")
            elif status == 2:
                canvas.create_rectangle(x1, y1, x2, y2, fill = "green")
            elif status == 3:
                canvas.create_rectangle(x1, y1, x2, y2, fill = "light green")




def drawGrid(canvas, grid):
    for row in grid:
        for node in row:
            drawNode(canvas, node)

    return




window = tk.Tk()
```

```
window.geometry("700x800")
canvas = tk.Canvas(window, width=winW, height=winH)
canvas.pack()

grid = generatGrid(nrows, ncols)
drawGrid(canvas, grid)
event(canvas, grid)



window.mainloop()

print("Exiting Simulation...")
```

Screenshot After 100 Days:


tk
Add a day

Comparison Analysis:

1. Ran for 100 days (shown above)
2. My hypothesis is that the transition with the least impact would be wear and tear, as it only doubles from 5% to 10%. The one with the greatest impact would be install, which, when doubled, becomes 80%.
3. Doubling of values is calculated in the table below.
4. My hypothesis was correct.
5. If this were to be increased to a 33x33 matrix, little to nothing would change in the percentages, because they are just that, percentages of the total, not amount of values.

Condition	**% new**	**% normal**	**% aged**	**%defective**
Base cond.	19%	45%	0%	36%
Double install	10%	54%	0%	36%
Double w&t	19%	41%	4%	36%
Double decay	19%	45%	0%	36%
Double repair	39%	45%	0%	16%
Dbl prevention	39%	25%	0%	36%
Double lemon	10%	45%	0%	45%
Dbl neighbor	19%	40%	0%	41%

Most difficult task/is complete:

For me, the most difficult task in this project was navigating the grid. It was simple to display the initial grid, however, going node by node to edit/check colors was extremely difficult. As such, the only thing I was unable to complete in this lab was the statistics display in X window, as well as checking neighboring nodes. Due to not being able to check neighbors, the possibilities that include checking neighbors were changed to an arbitrary percentage, close to what it would have been if neighbors were checked.