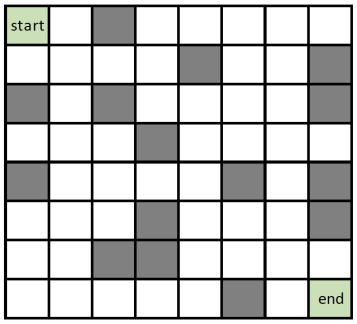
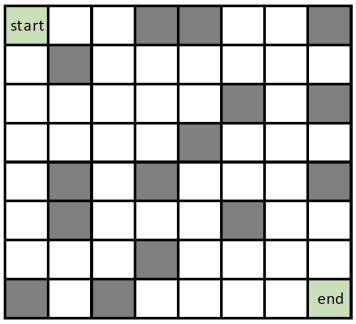
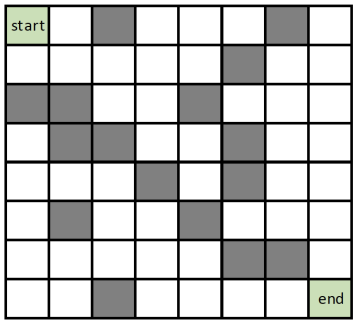
CSC 211
Final Project
Spring 2022

Overview: For your final project, you will write code to create and solve puzzles of the form pictured below. For each of these, we have an $N \times N$ grid of cells, where each cell is either open or blocked. The goal is to find the shortest path from the `start` cell to the `end` cell (which are both always open). A path cannot go through a blocked cell, and it may be possible for no solution to exist. A few examples with $N = 8$ are pictured below.







Part 1 – Create A Random Puzzle: The puzzle is to be an $N \times N$ structured graph, where you should be able to change the value of N at a single place in the code. Each vertex is a cell, and there is a edge between any two adjacent open cells. If a cell is blocked, then it has no edges to any of its neighbors.

To create a new random puzzle, you should start with a new `LinkedDirectedGraph()` object and add N^2 nodes with labels $0, 1, 2, 3, \cdots, N^2 - 1$. This is the same as in a previous assignment.

To choose which vertices are considered blocked should be done randomly with a probability of 0.25 for each cell (other than the start or end cells). For example, the code here chooses a random value in the interval $(0, 1)$, using `random()`, and will print `"blocked"` about 25% of the time and `"open"` about 75% of the time. I'd recommend using something like this as you add edges to the graph.

```
from random import *

if random()<0.25:
    print("blocked")
else:
    print("open")
```

For part 1, your code should produce a randomized puzzle graph object.

Part 2 – Solve the Puzzle: The solution to the puzzle can be found through the application of Dijkstra's Algorithm, starting with the `start` cell. The algorithm should give the shortest path length to the end cell in `D` and the path can be determined by `V`. Note also that if no solution exists, this can be determined by these as well.

For part 2, your code should start with a puzzle graph and print out the solution, which includes both the path length and the path sequence from `start` to `end`. This may all be text-based, but you are free to incorporate a visualization if you want.

Guidelines: To complete the project, you should make an appointment with me to demonstrate your code sometime during finals week (May 2, 3, 4, 5). I will send out instructions via email. For this project you may work alone or with a partner (i.e., a group of two). If you choose to work with a partner I will ask for a statement on how the work was divided between the team. I expect the work for this project to be done by the group with no outside help, unless authorized by me in advance.