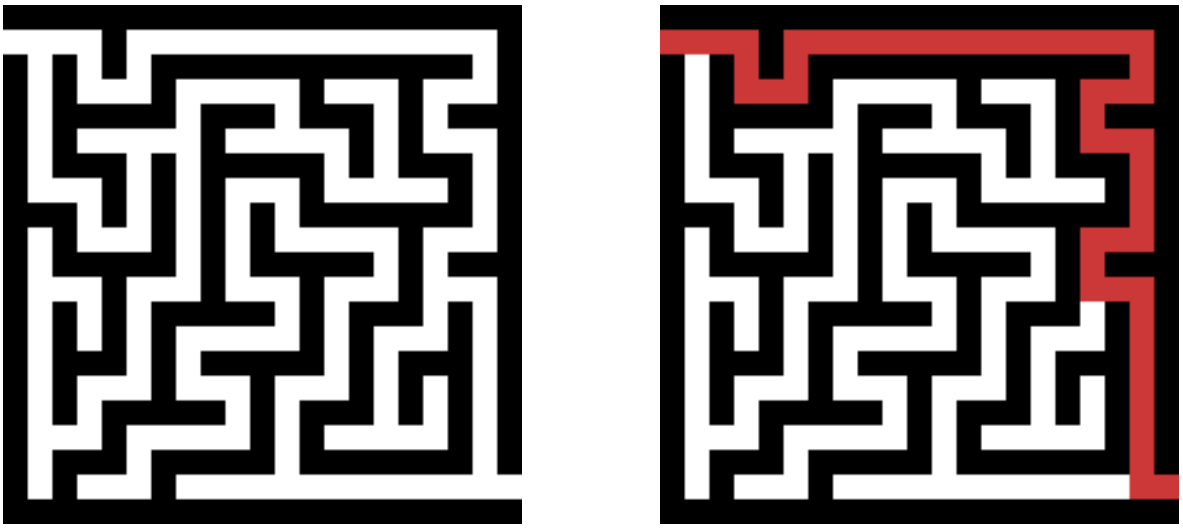
Project 1: Maze Solver using Markov Decision Processes

Due: October 10 - October 17, 2025

Project Details

The goal of the project is to create a maze solver using Markov decision processes and dynamic programming. Your code should take as input an image file named `maze.png`. The image would contain a maze as shown in Figure 1a. Your code must convert the image to a Markov decision process model and use either value or policy iteration to produce as output the solved maze image as shown in Figure 1b. Your code will both display the image on screen and save it as a `solvedmaze.png` in the local directory.



(a) Original maze

(b) Solved maze

Figure 1: Maze images

Some helpful codes and resources are available as below:

- There are several free resources available to generate mazes in order to test your algorithms. For instance, this free generator (`https://keesiemeijer.github.io/maze-generator/`) generates a PNG image for mazes of any custom size, rows and columns.
- The file image2bin.py available in the coursesite folder converts a png file containing a maze (in greyscale) to a binary matrix where walls are 0, and spaces are numbered 1.
- To reconvert an integer matrix back to an image, there are several ways. The easiest might be to use `colors.ListedColormap` from `matplotlib` to define a custom color map for the integers in your solution matrix, and plot your matrix, and save the plot using the `savefig` function as a file.

Submission Details

You will upload a report and your codes along with a README file that provides instructions on how to run your code. Your report should have the following:

1. **Methodology to convert the image into a Markov Decision Process** including the definition of the states, actions, transition, reward and termination criterion.
2. **The algorithm you used to solve for the optimal policy** (value iteration/policy iteration/generalized policy iteration etc), and the implementation of the algorithm which you will explain using snippets of code.
3. **Example outputs with mazes of varying sizes**.
4. **(Optional for 340) A study on execution time** of your code as a function of the maze sizes. You'll need to run multiple mazes of each size and use the average execution time for this study.