LECTURE 12

RECURSION WITH BACKTRACKING

MCS 275 Spring 2023

Emily Dumas

LECTURE 12: RECURSION WITH BACKTRACKING

Reminders and announcements:

- Project 1 due 6pm today.
- Project 2 description will be posted by Monday.
- Project 2 due 6pm on **Fri Feb 24**.

PLAN

- Recall backtracking algorithm to solve a maze
- Implement the maze solver
- Experiment with it

Algorithm `depth_first_maze_solution`:

Input: a maze and a path under consideration (partial progress toward solution).

1. If the path is a solution, just return it.
2. Otherwise, enumerate possible next steps that don't go backwards.
3. For each of the possible next steps:
    - Make a new path by adding this next step to the current one.
    - Make a recursive call to attempt to complete this path to a solution.
    - If recursive call returns a solution, we're **done**. Return it immediately.
    - (If recursive call returns `None`, continue the loop.)
4. If we get to this point, every continuation of the path is a dead end. Return `None`.

LET'S WRITE THIS IN PYTHON

`depth_first_maze_solution(M,path=None):`

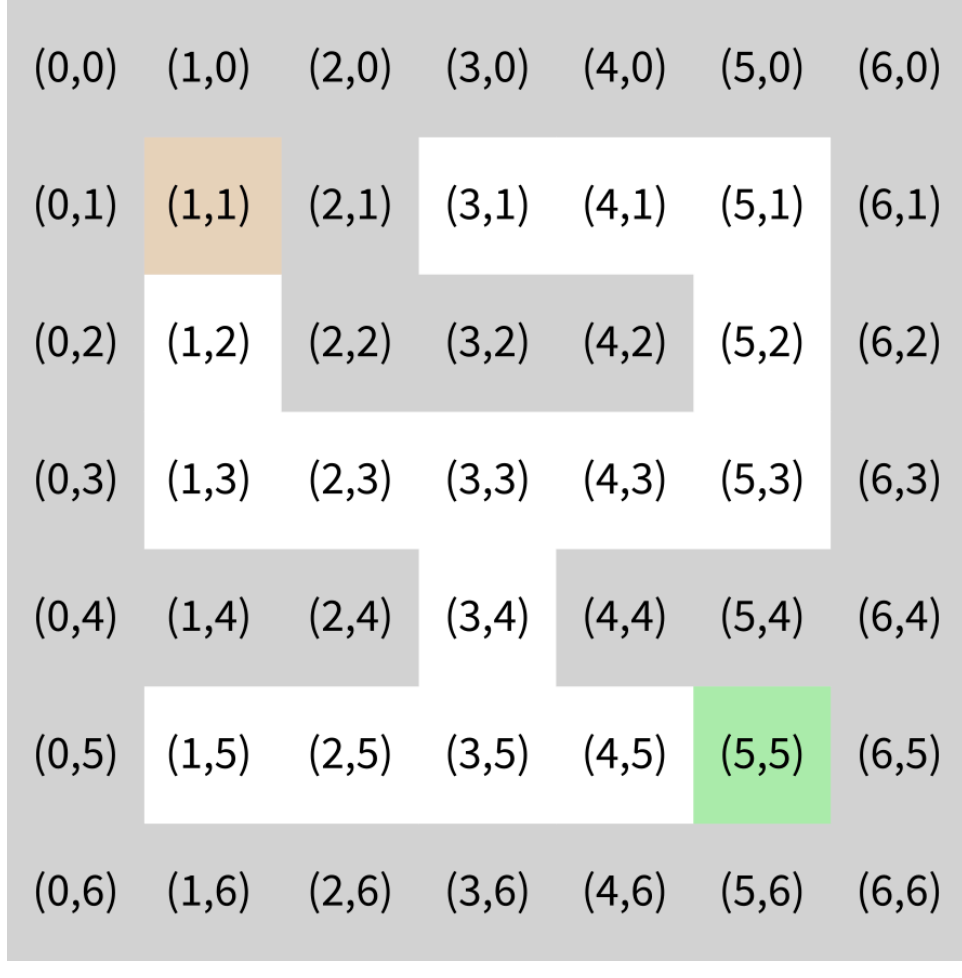
Arguments:

- `M` - a `Maze` object to be solved (read only)
- `path` - a list of `Point2` objects

Returns: Either

- List of `Point2` objects (solution extending `path`), or
- `None` (if no solution exists that extends `path`)

MAZE COORDINATES



MAZE COORDINATES

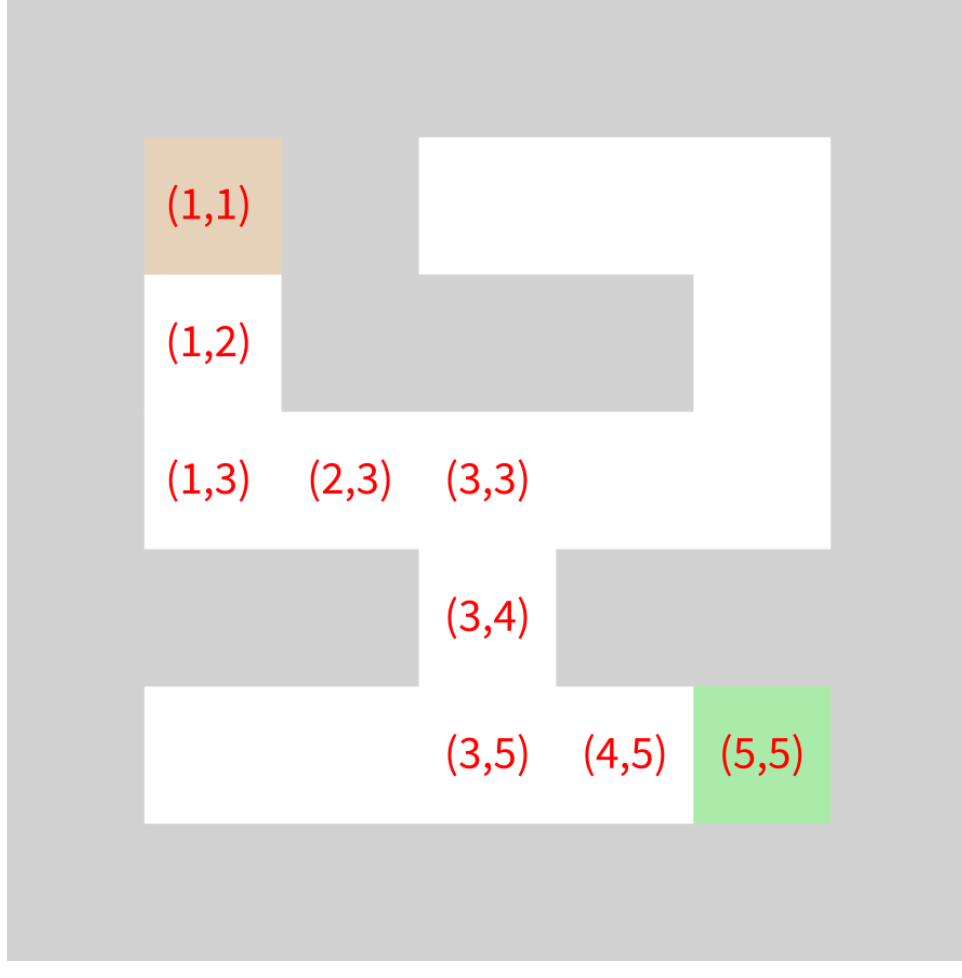


IMAGE SUPPORT

Class `Maze` can save an instance as SVG (`.save_svg(fn)`) or PNG (`.save_png(fn)`).

The latter requires a module called Pillow we'll discuss later. Can install with:

```
python3 -m pip install pillow
```

REFERENCES

Same suggested references as Lecture 10.

REVISION HISTORY

- 2022-02-14 Last year's lecture on this topic finalized
- 2023-02-10 Updated for 2023