# Unit 3 - Lesson 2. Introduction to Maze Generation Algorithms (Part 2)

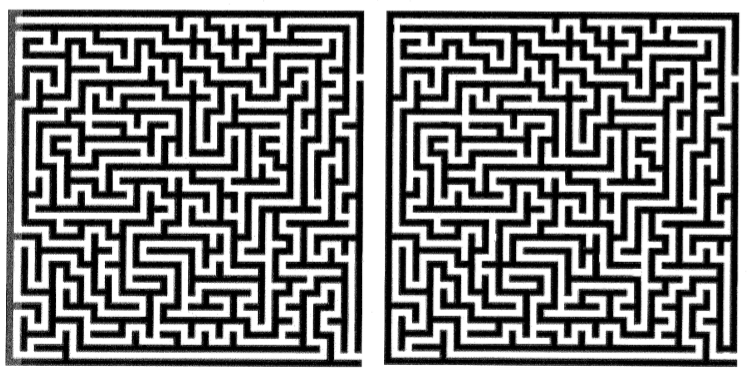
**Aim:** What are some of the simple random maze generation algorithms? How do we use basic graph theory and matrices to describe a maze?

**Objectives:** After the lesson, students should be able to:

* Understand the concept of a perfect maze, and a good maze vs. a bad maze
* Describe some simple random maze generation algorithms
* Interpret maze presented in discrete math and basic graph theory forms

**CLASS PROCEDURE:**

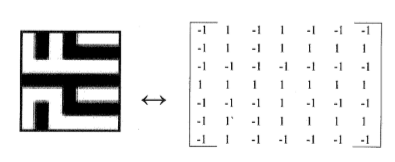
***Do Now:*** Solve the two mazes. Which maze is good? Which maze is bad?



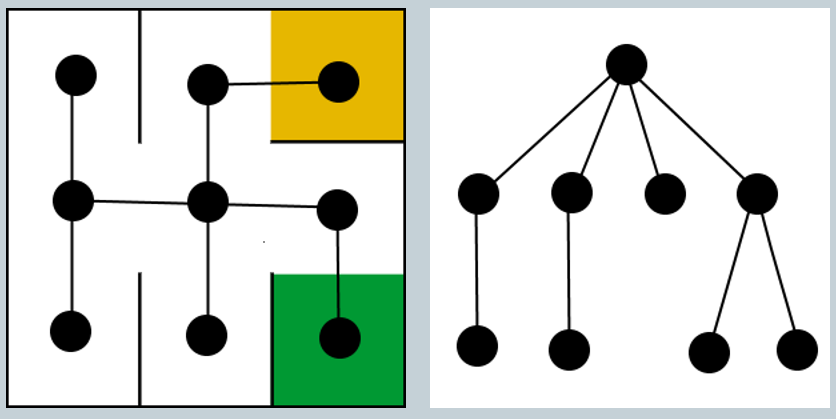
***Class Discussion / Presentation:***

1. What is a good maze and what is a bad maze?
2. What is a perfect maze?
3. If you were given a piece of paper and a pencil, how do you draw a 4 x 4 perfect maze?
4. Maze generation and pathfinding is a popular topic in math. How do we present a graphic maze in math terms? Here are some examples.

* A matrix of 1 and -1



* + We can use -1 to indicate a white cell, and 1 to indicate a black cell
  + Using discrete math and some calculations, we can find some patterns to generate and solve mazes.
* Mazes as trees
  + Mazes can be viewed as a spanning tree of a graph with a grid of vertices each connected to all their neighbors with equally weighted edges.
  + Means that any spanning tree algorithm can create a maze
  + The dimension of the maze does not matter as it can just be represented by a 2n-ary spanning tree for a nD maze (ex: 2D has a 4-ary spanning tree, 3D – 6ary, 4D – 8-ary)



1. When generating a random perfect maze, what are some of the most popular algorithms?
   1. Mazes as trees
   2. Recursive back-tracker
   * Works by randomly selecting a path until there are no more valid moves (there are no unvisited adjacent cells).
   * Next moves back until another move is possible.
   * Continues until the whole maze has been visited.
   1. Recursive division
   * Randomly places a wall dividing the maze in two and picks a random opening
   * Then does that same division with the two new sections
   1. Prim’s algorithm

* Picks a random starting point
* Continuously picks a random cell from the unvisited neighbors of the current visited cell
  1. Growing tree
* Pick a random cell and store it in a list
* Randomly pick more cells until no longer possible
* Once a dead-end is hit, use some condition to pick the next cell to iterate from, ex:
  + Most recent added cell: performs like a back-tracker
  + Random cell: looks similar to Prim’s algorithm

***Pair – sharing Activity #1:*** Choose one of the maze generation algorithms (you can find more on wiki:

<https://en.wikipedia.org/wiki/Maze_generation_algorithm>), work with your partner to draw a maze using pencil and paper.

***Pair – sharing Activity #2:*** The following Unity / C# program generates a maze. Enter this program into Unity and try it out. Discuss with your partner on the algorithm it used.

***Pair – sharing Activity #3:*** In the next MazeRunner game, you need to generate a random maze with walls, paths, cells and floor. You must have a player controlled character (1st or 3rd person controller) to navigate through the maze. You also need to design an agent (game controlled character) to auto-solve the maze. You may start to design the game play with your partners.