Department of Computer Engineering
Faculty of Engineering
University of Sri Jayewardenepura

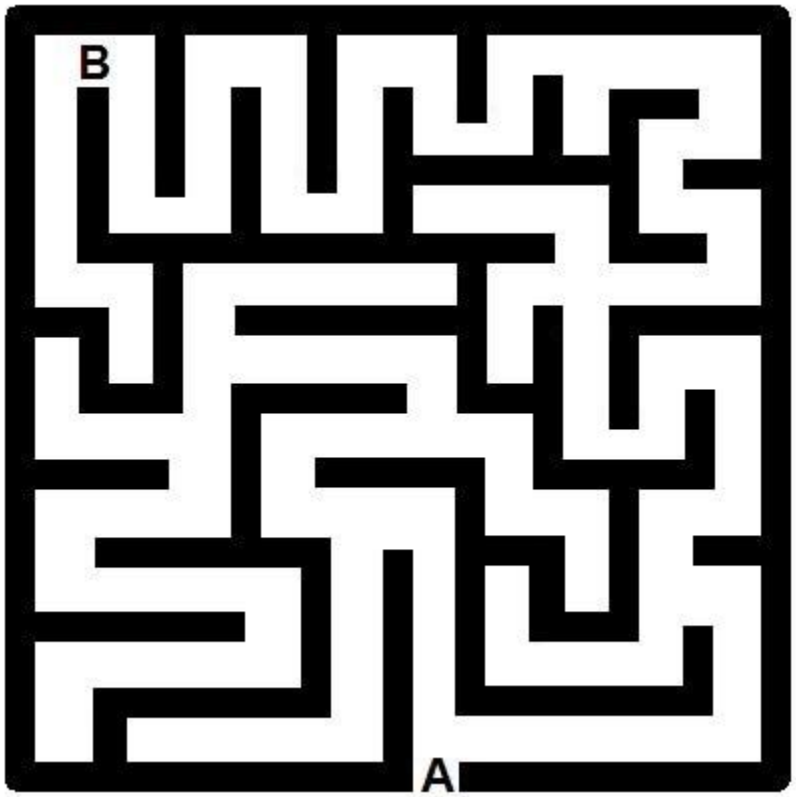
Course	Data Structures & Algorithms
Course Code	CO2201
Title	Maze Generator and Maze Solver
Practical Number	2
Outcomes	Implementing recursive BFS and DFS algorithms
Deadline	**22nd June 2021**

General Instructions:

- No food, drinks, backpacks, and bags are allowed to take inside the laboratory.
- Please save your work frequently during the practical session to avoid data loss due to unavoidable circumstances.
- Your files will be erased after the practical session. Therefore, please keep a backup for yourself.
- Please archive all files to a zip file, upload the zip file to LMS, and send it as an attachment to coassignments@gmail.com.
- Use the following format when you are naming the zip file: yy_ENG_xxx_L.zip, yy_ENG_xxx is your registration number, and L stands for the practical number (e.g. 16_ENG_135_1.zip)

1. The Maze Puzzle

The Maze puzzle is a popular puzzle game in which the player is required to find a route from the designated starting point to the endpoint of a maze-like puzzle. The Maze is typically rectangular and consists of paths usually denoted by white in which the player can freely travel through and walls denoted by black which a player cannot move across. The image below shows a typical Maze Puzzle.



As can be observed from the image above, a maze puzzle consists of many paths that lead to a dead-end, and one path leading to the goal. There are some extended variants with more than one path leading to the goal as well.

2. Generating a puzzle maze

Generating a challenging puzzle maze with a valid solution is a task that is as challenging as solving one. One of the easiest ways to create a puzzle maze is by generating it using a computer with the help of a randomized algorithm.

For this practical, we will break down the puzzle into “cells” where a single cell represents a single block on the puzzle. A ‘0’ would represent a white block which would be part of a path and 1 would represent a black block which would be a part of a wall.

The starting point and the goal must always be on white blocks and must be denoted by the letters “A” and “B” respectively. All horizontal blocks must be space separated. For example, the puzzle in the above image would be represented by a 21x21 grid as follows:

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 B 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 1 1 0 1
1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 0 0 1
1 0 1 0 1 0 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1
1 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1
1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1
1 1 1 0 1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1
1 0 1 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1
1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 0 1 0 1
1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 1
1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1
1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 1
1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 0 1 0 1 1 1
1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1
1 1 1 1 1 1 1 0 1 0 1 0 1 1 1 1 1 0 1 0 1
1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 1
1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 0 1
1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1 1 A 1 1 1 1 1 1 1 1 1
```

Implement an algorithm to create a **random square-shaped puzzle** of a given size. The algorithm must always create a valid puzzle with **one** solution. The generated puzzle must be saved to a text file. The implementation will be graded according to the complexity of the puzzle as well as the time complexity of the algorithm.

3. Implementing a Maze Puzzle Solver

Implement a separate algorithm to solve the mazes generated and saved as text files in section 2. The algorithm must be completely separate from the generator implemented in part 2 and must not be able to access any variables of the generator. The Solver **must** incorporate **recursion** and **backtracking**. The solution path found by the algorithm must be denoted by replacing the cells with “#” symbols and saved to a separate text file. For example, the solution to the puzzle shown above must be denoted as follows:

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 B # 1 # # # 1 # # # 1 # # # # # # # 1
1 0 1 # 1 # 1 # 1 # 1 # 1 # 1 0 1 1 1 # 1
1 0 1 # 1 # 1 # 1 # 1 # # # 1 0 1 # # # 1
1 0 1 # 1 # 1 # 1 # 1 1 1 1 1 1 1 # 1 1 1
1 0 1 # # # 1 # # # 1 0 0 0 0 0 1 # # # 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 # 1
1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 # # # # # 1
1 1 1 0 1 0 1 1 1 1 1 1 1 0 1 # 1 1 1 1 1
1 0 1 0 1 0 0 0 0 0 0 0 1 0 1 # 1 # # # 1
1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 # 1 # 1 # 1
1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 # # # 1 # 1
1 1 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 1 1 # 1
1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 # # # 1
1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 0 1 # 1 1 1
1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 # # # 1
1 1 1 1 1 1 1 0 1 0 1 0 1 1 1 1 1 0 1 # 1
1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 0 1 # 1
1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 # 1
1 0 1 0 0 0 0 0 0 0 1 # # # # # # # # # 1
1 1 1 1 1 1 1 1 1 1 1 A 1 1 1 1 1 1 1 1 1
```

4. Submission Guidelines

Put your codes and answer files into a zip file and name it with your index number (ex. 16_ENG_001_P1.zip) and upload it to the LMS on or before the deadline. When the file is submitted to the LMS, make sure

- The file is properly uploaded.
- Check the student statement, and
- Press the submit button.

Warning: The assignment will not be properly submitted if the above steps are not correctly followed.

Deadline: 22nd June 2021

Prepared by Mr Shalila Balasooriya, Instructor, Department of Computer Engineering