Advanced Algorithms – Assignment 2

Lucas Geurtjens (s5132841)

# Q1. Tracking K-Smallest Numbers

**Introduction:**

…

**How it works:**

…

**Input/Output:**

…

**Amortised Analysis:**

…

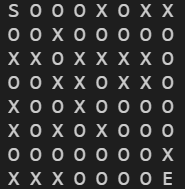
# Q2. Maze Generation

**Introduction:**

A maze generation algorithm was created using the technique of randomly knocking down walls until a path from the start and end is found.

**How it works:**

Here’s an example of an 8x8 maze:



LEGEND:

**S**: Start of maze

**E**: End of maze

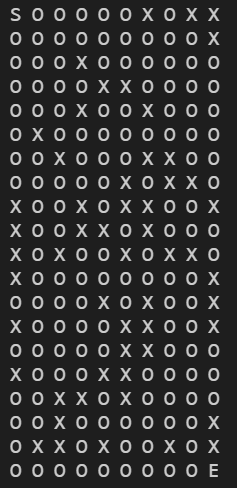
**X**: Wall (cannot pass)

**O**: Path (can pass)

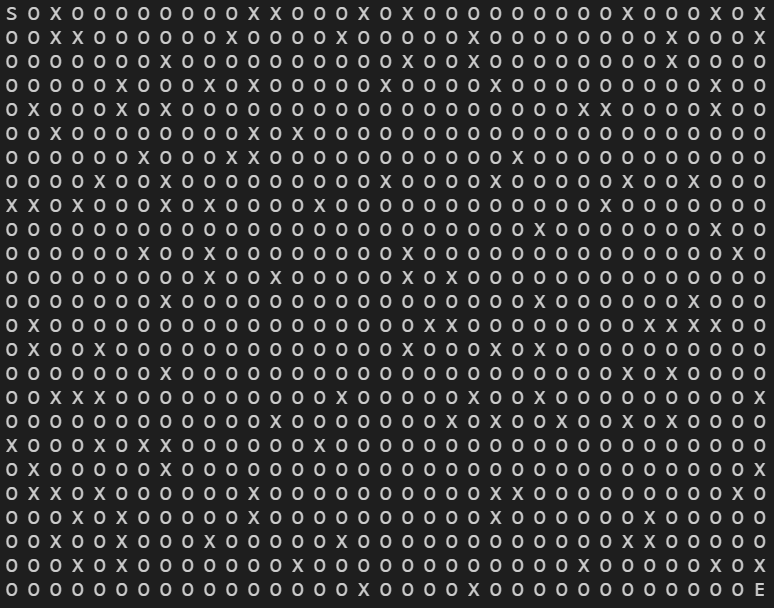
* The maze grid is initialised (filled with walls)
* While there is no complete path from the Start to the End of the maze:
  + Check if the maze is complete (if so, we can end)
    - Run a Breadth First Search from the Start -> End
    - Run a Breadth First Search from the End -> Start
      * During the BFS, calculate the furthest we can get to the goal state (done using Manhattan distance). When the BFS finishes (assuming it didn’t find the goal state), this value will help determine the minimum number of walls we must break down.
    - Calculate the number of walls to break down (the Manhattan distance from the furthest point from the start, to the furthest point from the end).
    - Breakdown the number of walls specified by randomly selecting points. If the point is a wall, break it down, and add this to the total amount of walls broken.

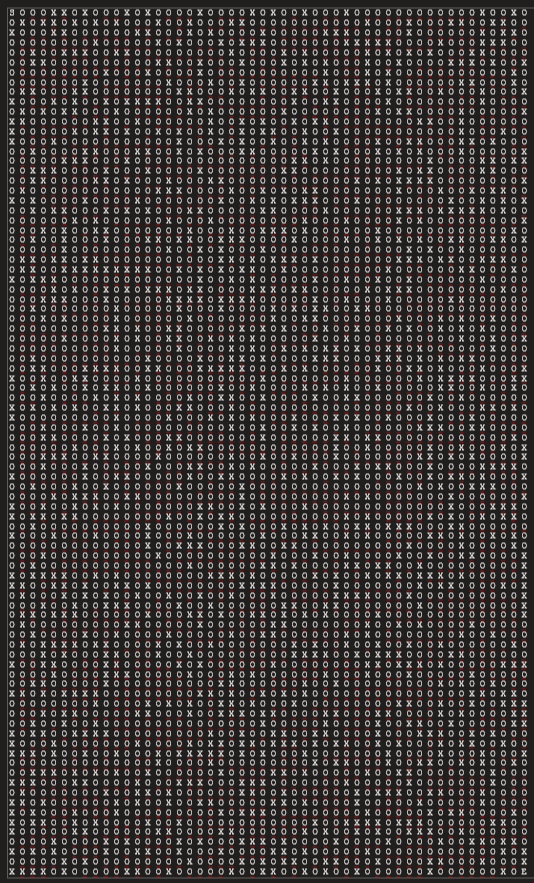
**Input/Output:**

10x20 Maze

****

35x25 Maze

****

50 x 88 Maze: 

# Q3. Red Black Trees vs Van Emde Boas Trees

**Introduction:**

…

**How it works:**

…

**Input/Output:**

…

**Performance Analysis:**

…

# Q4. The Kevin Bacon Game

**Introduction:**

…

**How it works:**

…

**Input/Output:**

…

# Q5. Minimum Vertex Covers for Complement Graphs

**Introduction:**

…

**How it works:**

…

**Input/Output:**

…