COS30008: Data Structures and Patterns

Problem Set 3

Weighting: This assessment contributes 5% to the overall unit marks.

Learning Outcome(s) assessed: 1, 2, 3, 4

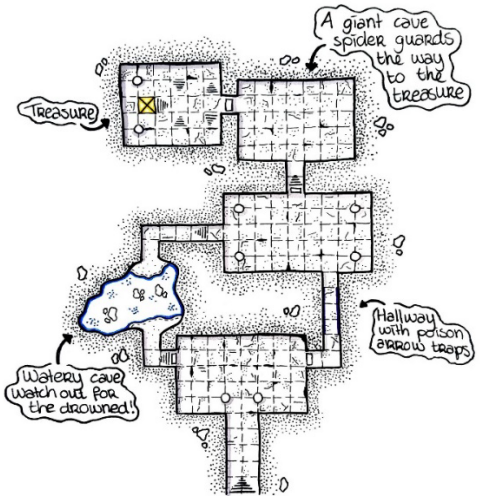
The focus is on being aware of commonly reoccurring programming tasks, how apply the best choice of data structures to quickly deliver them, and fully utilize the structures' accessibility/functionality.

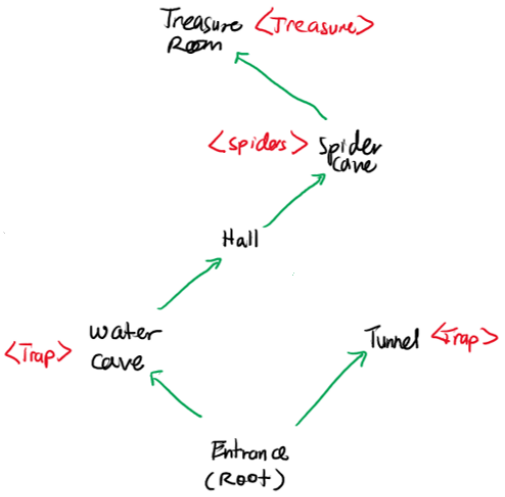
Introduction: Trees & Visitor

A Tree is an Abstract Data Type structure used to map interlinked data nodes that originate from a single starting point (known as the root node of the tree). This helps organize the collection of nodes by relativity (one node connected to another by an edge) so a variety of search algorithms can be used to navigate through the edges and nodes to find a target destination.

The path selection at each subtree is determined by a decision-making process. In a game AI behavior tree, this is simply a check to see if a specific game condition was met. Other examples of trees applied to organize room or event sequences may sometimes violate the classic Tree rule of no cyclic edges (or having multiple parents, thus turning this structure into a Graph), and this is okay (if by design, you don't plan to implement conclusive search like DFS or BFS).

The simplest application of a Tree is in organizing the layout of a Dungeon in a fantasy adventure game. Here, exploring one room into next (nodes in the tree) is done manually by the player. Each room can contain entities like monsters, treasure, traps, and other narrative items. The player can interact with these entities inside the rooms by applying actions (implemented as a concrete visitor). Refer to Week 9 Lab Tutorial Examples as reference.





Task 1 (1%): Develop you example simple text game environment: game loop + Player entity + other entities such as Items, Monsters, or Clues.

Report: Short description of the simple game + Class Diagram showing the data structure design involved.

Task 2 (1%): Plan your rooms using a diagram (indicate what entities each room contains and what actions you intend the player can do in them). Examples shown above (make sure yours is different!)

Report: Draw a diagram to represent the room layout in the form of a Tree.

Task 3 (1%): Build your in-game dungeon:

a) A node class to describe the room and keep track of its entities.
b) A dungeon class to manage this tree of rooms for the game.
c) Test exploration of this dungeon to show that you (the Player) can move through the Rooms (nodes) by selecting which door to enter (the edges at each sub-tree Root).

Report: Use Raw Copy/Paste of Code (and comments) + Screencap Output console windows to show it working.

Task 4 (2%): Implement player actions using the Visitor design pattern.

Report:

a) Use Raw Copy/Paste of Code (and comments) + Screencap Output console windows to show it working.
b) Reflection Summary (in point form):
    a. Did you run into any problems while completing this PS? How did you solve it? What references did you use?
    b. Comment on your experience using the Tree ADT and Visitor pattern (how challenging was it to learn, was it practical for your program, what else can you use it for, etc.).

Name your report as **"COS30008_2025S2_PS3_<YourStudentID>"** and submit via Canvas as **PDF**.