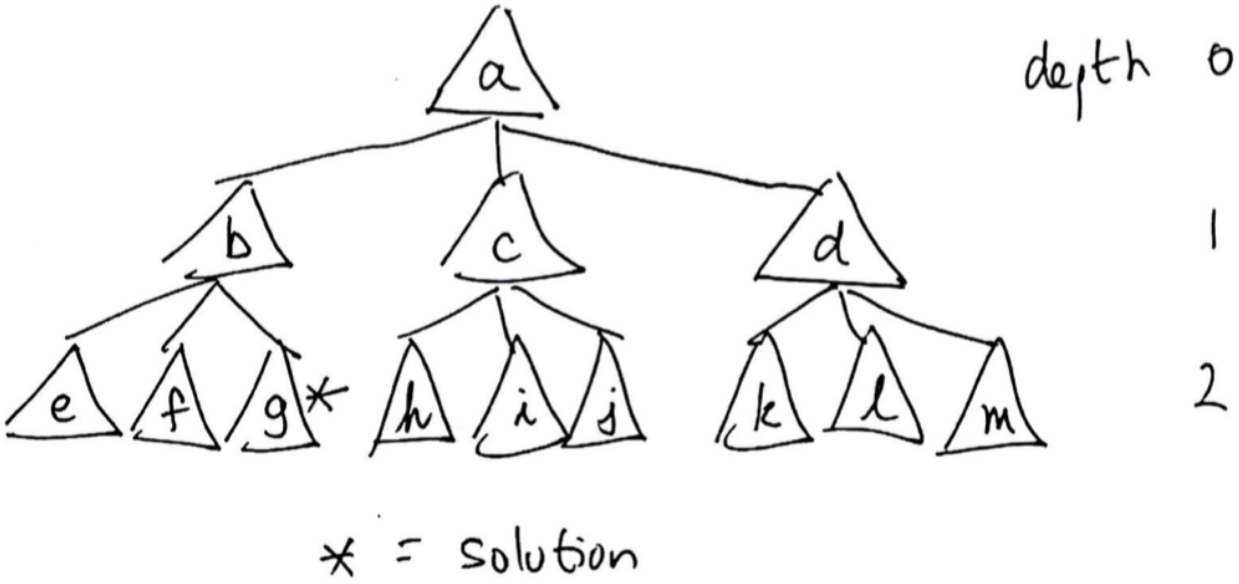
SEARCH ALGORITHMS AND ANALYSIS.

Let's understand two search algorithms: depth-first search (DFS) and breadth-first search (BFS).

In the diagram below, the triangles represent states in a search problem. For example, you can think of triangle "a" as the initial state of the cars in the traffic game.

Each state has successor states, indicated by the lines. State "a" has three successors: b, c, and d. Building on the traffic game example: State b, c, and d are each reached by a different move from state a. A solution or goal state is marked with an asterisk (*).



At each stage of the search:

1. The goal test is applied. If the state under consideration is a goal, the search terminates (and returns the path to the goal).
2. Otherwise, the state is "expanded" to produce its successor states. Then those states are searched.

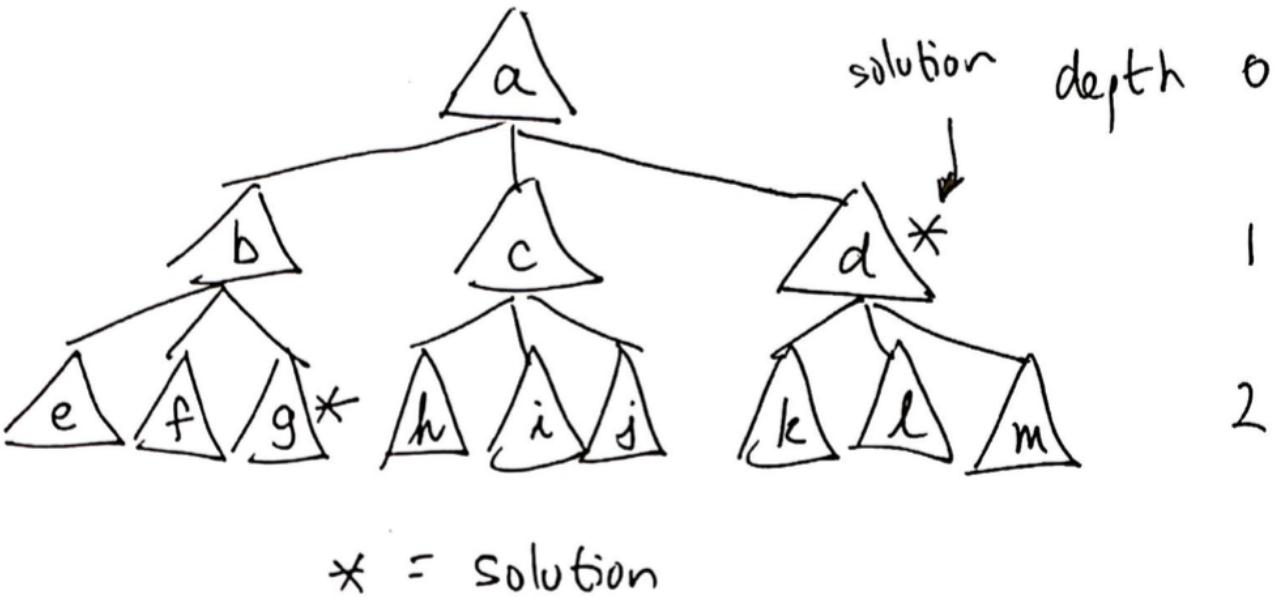
Depth-first search (DFS) searches down, then across.
Breadth-first search (BFS) searches across, then down.

Perform DFS and BFS, and write the solution path discovered ***and*** the sequence of states visited along the way to finding the solution (if found). Assume each state is expanded from left to right.

DFS (solution) (states visited)

BFS (solution) (states visited)

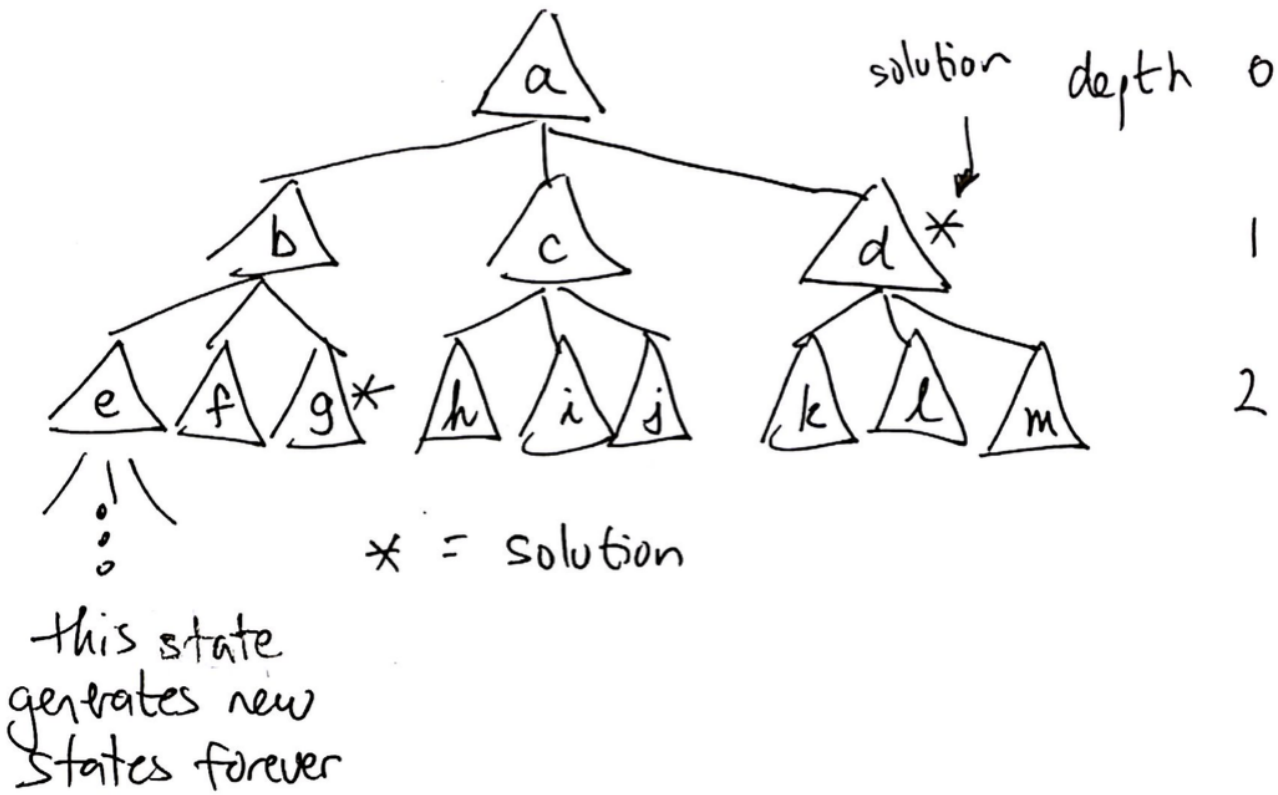
Now there is a second goal state. Determine the solution discovered and states-visited sequence for DFS and BFS.



DFS (solution) (states visited)

BFS (solution) (states visited)

Now, state e generates a tree of successor states that never terminates. Do the same exercise.



DFS (solution) (states visited)

BFS (solution) (states visited)

Next we will characterize properties of the two search algorithms. There are four properties of interest:

Complete: Whether a solution will be found, if a goal state exists. Yes or no.

Optimal. Whether the best solution will be found, if there is >1 goal state. Yes or no.

Time complexity. How long does it take to find a solution? Counting the states-visited along the way is a good way to determine this.

Space complexity. How much memory is used during the search process? Think about which states need to be kept track of (maintained in memory) in order to perform the search.

For space and time complexity, express your answers in terms of:

- The branching factor, b. In our examples, $b = 3$.
- The depth of the shallowest solution, d. In the second example, d=1.
- The maximum depth of the search tree, m. In the first and second examples, m=2. The last example, m= infinity.

	BFS	**DFS**
Complete?	yes no	yes no
Optimal?	yes no	yes no
Time complexity		
Space complexity		