Written Assignment 3

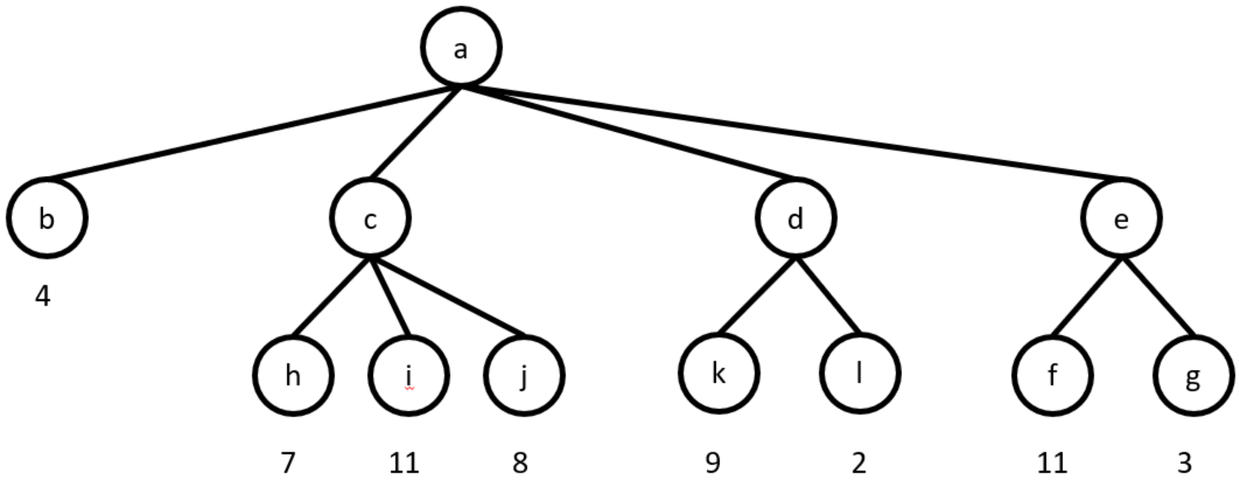
Due: Friday 02/21/2025 @ 11:59pm EST

Disclaimer

I encourage you to work together, I am a firm believer that we are at our best (and learn better) when we communicate with our peers. Perspective is incredibly important when it comes to solving problems, and sometimes it takes talking to other humans (or rubber ducks in the case of programmers) to gain a perspective we normally would not be able to achieve on our own. The only thing I ask is that you report who you work with: this is **not** to punish anyone, but instead will help me figure out what topics I need to spend extra time on/who to help. When you turn in your solution (please use some form of typesetting: do **NOT** turn in handwritten solutions), please note who you worked with.

Question 1: Minimax Execution (25 points)

Consider the tree expansion:



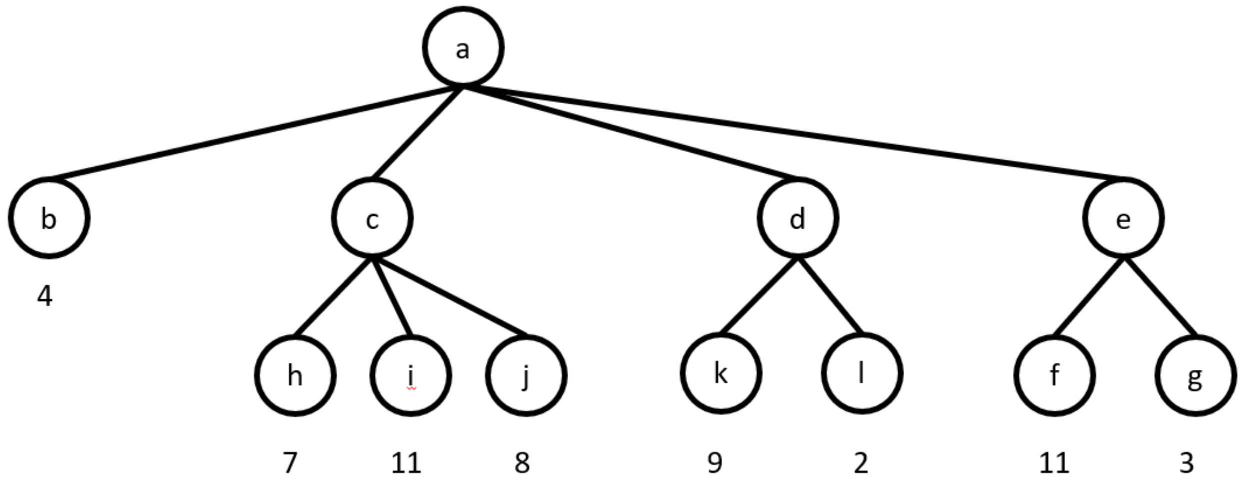
Run the vanilla minimax algorithm on this tree (the root node is the `MAX` player and the two players `MAX` and `MIN` alternate turns) to determine what move the `MAX` player should make at the root. If you need a heuristic value, that the heuristic function is defined as follows:

$$h(n) := n.level + n.idx$$

where n is a node in the tree, $n.level$ is the 1-indexed level of that node in the tree (i.e. the root has level 1, the next layer has level 2, etc.), and $n.idx$ is the 1-indexed index of that node within its level (i.e. the left most node in a level has index 1, the node immediately to its right has index 2, etc.). Show all of your steps.

Question 1: Iterative Deepening Execution (25 points)

Consider the tree expansion:



Run the Iterative Deepening algorithm on this tree (the root node is the `MAX` player and the two players `MAX` and `MIN` alternate turns) to determine what move the `MAX` player should make at the root. If you need a heuristic value, that the heuristic function is defined as follows:

$$h(n) := n.level + n.idx$$

where n is a node in the tree, $n.level$ is the 1-indexed level of that node in the tree (i.e. the root has level 1, the next layer has level 2, etc.), and $n.idx$ is the 1-indexed index of that node within its level (i.e. the left most node in a level has index 1, the node immediately to its right has index 2, etc.). Show all of your steps.

Extra Credit: Correctness of Alpha-Beta Pruning (25 points)

Let s be the state of the game, and assume that the game tree has a finite number of vertices. Let v be the value produced by the minimax algorithm:

$$v = \texttt{Minimax}(s)$$

Let v' be the result of running Alpha-Beta Pruning on s with some initial values of α and β (where $-\infty \leq \alpha \leq \beta \leq +\infty$):

$$v' = \texttt{Alpha-Beta-Pruning}(s, \alpha, \beta)$$

Prove that the following statements are true:

- If $\alpha \leq v \leq \beta$ then $v' = v$
- If $v \leq \alpha$ then $v' \leq \alpha$
- If $v \geq \beta$ then $v' \geq \beta$

This means that if the true minimax value is between α and β, then Alpha-Beta pruning returns the correct value. However, if the tru minimax value if outside of this range, then Alpha-Beta pruning may return a different value. However, the incorrect value that Alpha-Beta pruning returns is bounded in the same manner that the true minimax value is (i.e. if the true minimax value is $\leq \alpha$ then the value produced by Alpha-Beta pruning is also $\leq \alpha$ and vice versa). Note that this implies that Alpha-Beta pruning will be correct with initial values of $(-\infty, +\infty)$ for (α, β).

Hint: use induction. If s is not a terminal state, then you can correctly assume that the claim above holds for all children of s. Use this assumption to prove that it also holds for s (the base case is trivial: minimax and Alpha-Beta pruning produce the same value for terminal states)