Bonus Homework

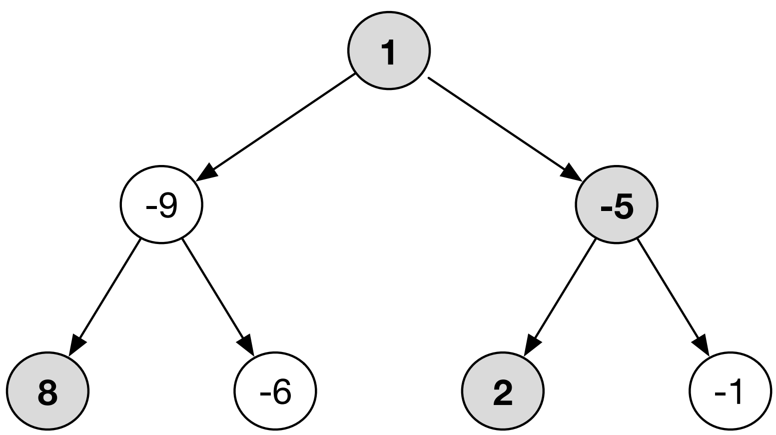
CS/ECE 374 B

Due 8 p.m., **Wednesday**, ~~December 11~~ December 18

- This is a bonus homework; solving it is not required
- This homework will not be graded until after the final
- The grades in this homework, as with all bonus grades, will not be used in initial letter grade computation
- This homework will not be graded unless it has the potential to improve your letter grade. E.g., if your worst three homework scores are 5, 7, and 8, this homework can at most improve your final score by 1%. If an extra 1% will not change your letter grade, we might not grade your homework.
- As always, you need to provide justification that your algorithm is correct. If you use a greedy strategy, prove that your greedy strategy produces an optimal solution.

Question 1: Scoring Trees .... 10 points

In this problem, you will be considering a binary tree with a score assigned to each node. Given a subset of the tree nodes, the score of that subset is the sum of the score of the nodes in the subset that do not also have a parent inside the subset. For example, in the tree below, the highlighted nodes have a score of $8 + 1 = 9$ because the nodes 8 and 1 do not have parents in the subset, whereas -5 and 2 do not count because their parents are selected.



Design and analyze an efficient algorithm that, given a tree T, the scores at each node, and a number k, finds maximum score of any subset of size k.

Question 2: Searching for an Exit .... 10 points

You are given a directed graph $G = (V, E)$ with non-negative scores on both vertices and edges: $\ell : V \cup E \to \mathbb{R}+$. The cost of a path $v_1 \to v_2 \to \cdots \to v_n$ is the sum of the scores of both edges and vertices:

$$\sum_{i=1}^{n} \ell(v_i) + \sum_{i=1}^{n-1} \ell(v_i \to v_{i+1})$$

You are given a source node s and a set of target nodes T. Design and analyze an efficient algorithm to find the cost of the least-cost path from s to some node in T.

Question 3: Building a Matrix .... 10 points

You are given two arrays $C[1..n]$ and $R[1..n]$ of non-negative integers. Design and analyze an efficient algorithm to construct a binary matrix (i.e., so that each entry is 0 or 1) where the ith row adds up to $R[i]$ and the jth column adds up to $C[j]$, or output that this is impossible. For example, given $R = [1, 3, 1, 3, 3]$ and $C = [2, 2, 1, 2, 4]$, you might produce the matrix:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Note that there may be multiple possible matrices that work; your algorithm can return any of them. If the solution is impossible you should return an error.