CS 101 Fall 2020 - Quiz 11-A

12/29/2020 - 25 Minutes **Name:** **ID number:**

---

(1) (5 Points) Consider Knapsack problem without repetition with n items with values v_i and weight w_i. We have following defined sub-problems:

$K[w, i, 1]$ = maximum value of a collection of items with total weight w that contains item i

$K[w, i, 0]$ = maximum value of a collection of items with total weight w that does not contain item i

Is it possible to define a recurrence relation to solve above sub-problems? If possible, give the recurrence formula; otherwise provide the reason why it is not possible.

Not possible. You need more information about the contained items in the sub-problems in order to compute $K[w, i, \star]$, as the knapsack is without repetition. This is why the knapsack problem uses the first i items rather than inclusion/exclusion of a particular item.

(2) (10 Points) In this problem, we want to figure out the number of structurally unique BSTs (binary search trees) that stores the given values.

- (5 Points) Given values 1, 2, 3, draw all the structurally unique BSTs that stores these values. How many structurally unique BSTs can you draw?
- (5 Points) Given values 1, ..., n, design an algorithm with dynamic programming that figures out how many structurally unique BSTs can you draw. Give the explanation of your algorithm and the time complexity.

(a) When $n = 3$, there are a total of 5 unique BSTs.

(b) Let $A[n]$ be the number of such trees (where $A[0] = 1$). Any binary search tree must have a root. Once we have chosen the (zero-indexed) root $0 < k \leq n - 1$, there are $n - 1$ numbers left to place. k of those numbers must be in the left subtree, and $n - k - 1$ numbers must be in the right subtree. Therefore, once we have chosen the root k, there are $A[k]$ ways to arrange the left subtree, and $A[n - 1 - k]$ ways to arrange the right subtree. Summing over all possible roots, we get

$$A[n] = \sum_{k=0}^{n-1} A[k] \cdot A[n - 1 - k]$$

We use this recurrence relation to compute $A[1]$, $A[2]$, ... in sequential order, saving the values as we go along until we reach $A[n]$.