CS2302 Data Structures
Spring 2020
Backtracking and Dynamic Programming

Individual work, you may ask for assistance from your assigned staff member, the instructor and your teammate form last week.

1. (Backtracking) Trace the execution of the backtracking algorithm to determine if there is a subset of the integers [9,5,3,1] that adds up to 6. Draw a recursion tree displaying the recursive calls that are made by the function.
2. (Dynamic programming) Trace the execution of the minimum coins algorithm to determine the minimum number of coins needed to make 9 cents with denominations [6,4,1]. Write the contents of the coins array after every iteration.
3. (Dynamic programming) Trace the execution of the edit distance algorithm to determine the edit distance from 'whales' to 'wash'.
4. (Backtracking) The partition problem consists of, given a set of positive integers S, determining if there is a way to partition S into two sets S1 and S2 such that sum(S1) == sum(S2). Recall that in a partition of S into S1 and S2, each element of S most belong to either S1 or S2, but not to both. Hint: use subsetsum.
5. (Dynamic programming) Modify the edit distance function to allow insertions and deletions, but not replacements. Hint: this does not require any major modification to the function.
6. (Dynamic programming) Intuitively, since a replacement is equivalent to an insertion and a deletion in sequence, its cost should be higher than the cost of each of these operations. Modify the edit distance function to assign a cost of 2 to insertions and deletions, and a cost of 3 to replacements. Hint: this does not require any major modification to the function.
7. (Dynamic programming – Extra credit) Modify the minimum coins function to also return a list containing the number of coins of each denomination that must be given.