CS218 HW 3 Programming

due Thursday, May 2, 23:59 PM

Problem A // ID: 257228109

Dynamic programming: rows with the subset of items used, and the columns being the weight limit. The limitation of k items is introduced through a loop that runs the lesser of: 1) the current weight limit divided by the current item, or 2) k times, the max any item can be used. Same recursive formula as the original knapsack problem, except introducing the number of times to repeat an item for weights and values.

Runtime: $O(n * W * k)$ due to the triple-nested loop populating the DP array; Space Complexity: $O(n * W)$, n rows, W columns in the DP array.

Problem B // ID: 257232800

Dynamic programming with memoization. Loop through all rows and columns as starting point, to determine longest path out of all starting points. Then, the helper function chooses the max length path from making a valid move in the 4 adjacent cells. The overall result would be the longest length path throughout the grid.

Runtime: $O((r * c)^2)$ due looping through r rows and c columns, and each call to the helper function takes $O(r * c)$ due to recursive calls to find best path; Space Complexity: $O(r * c)$, storing memoization for longest path at each row column.