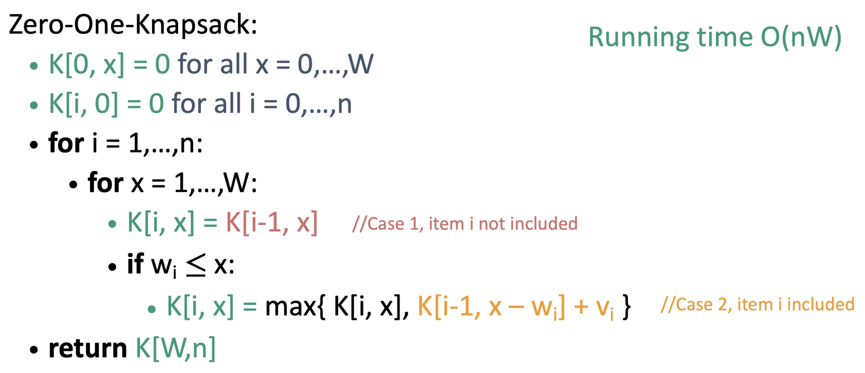
Week 9: Lab

Module 4: Techniques

Collaboration level 0 (no restrictions). Open notes.

1. **Knapsack botton-up:** We have a backpack of capacity W and a set of n items 1...n, each item with weight w_i and value v_i. Let $K[i][x]$ represent the maximal value for packing a backpack of capacity x with a subset of items 1 through i. Consider the iterative DP algorithm which fills in the table $K[0..W][0..n]$:



```
Zero-One-Knapsack:                              Running time O(nW)
• K[0, x] = 0 for all x = 0,…,W
• K[i, 0] = 0 for all i = 0,…,n
• for i = 1,…,n:
    • for x = 1,…,W:
        • K[i, x] = K[i-1, x]      //Case 1, item i not included
        • if $w_i \leq x$:
            • K[i, x] = max{ K[i, x], K[i-1, x – $w_i$] + $v_i$ }      //Case 2, item i included
• return K[W,n]
```

Now assume we have a backpack of capacity 3, and three items ($n = 3$): a hat of weight 1 and value 1, a ball of weight 2 and value 4, and a bottle of water of weight 3 and value 6.

(a) Show how the table is filled in by the iterative algorithm above.

	x=0	x=1	x=2	x=3
i=0	0	0	0	0
i=1	0			
i=2	0			
i=3	0			

	x=0	x=1	x=2	x=3
i=0	0	0	0	0
i=1	0			
i=2	0			
i=3	0			

(b) For each entry $[i][x]$ in the table, label it as "with" if $K[i][x]$ includes item i, and "without" otherwise.

(c) Use the table to compute the set of items corresponding to $K[3][3]$.

(d) Give pseudocode to compute the set of items corresponding to $K[n][W]$.

2. **Pharmacist problem:** A pharmacist has W pills and n empty bottles. Bottle i can hold p_i pills and has an associated cost c_i. Given W, $\{p_1, p_2, ..., p_n\}$ and $\{c_1, c_2, ..., c_n\}$, you want to store all pills using a set of bottles in such a way that the total cost of the bottles is minimized. So the problem is to find the minimum cost for storing the W pills and what bottles to use. Note: If you use a bottle you have to pay for its cost no matter if you fill it to capacity or not.

   (a) Explain how the problem has optimal substructure.

   Answer: Consider an optimal solution O, and consider one of the bottles in it. Let's say this is bottle k, and it holds p_k pills. Then we know that the remaining bottles in O must be the optimal way to store ....

   (b) Consider the following subproblem: Let $MinPill(i, j)$ be the minimum cost obtainable when storing j pills using bottles among 1 through i.

   Give a recursive formulation of this problem. You can describe it either using mathematical notation or as pseudocode.

   (c) Give pseudocode for a top-down recursive dynamic programming algorithm with memoization and analyze its running time.

3. **Greedy pharmacist?** Someone proposes the following greedy strategy to solve the pharmacist problem (above): Pick the bottle with the smallest cost-per-pill, and recurse on the remaining pills with the remaining bottles. Show that this greedy strategy is not correct by giving a counterexample.

4. **A different pharmacist problem**: Same problem as above, but all bottles cost the same. Given a number of pills W and n bottles which can hold $\{p_1, p_2, ..., p_n\}$ pills, respectively, describe a greedy algorithm to determine the fewest number of bottles needed to store the pills.

   (a) The algorithm:

   (b) Analysis:

   (c) Correctness: Remember that in order to prove that a greedy algorithm is correct, it is sufficient to prove that **there exists an optimal solution that contains the first greedy choice**. For this problem, you want to show that:

Claim: There exists an optimal solution which contains the first bottle chosen by the greedy algorithm above.

Proof: For simplicity we assume the bottles are numbered in order of their capacities, with bottle 1 being the largest. The first choice of the greedy algorithm is ....

Now consider an optimal solution O. We want to prove the following that there exists an optimal solution that contains bottle 1.

We'll prove this by showing that O can be transformed into another optimal solution O' that contains bottle 1. There are two cases: If O already contains bottle 1 then we are all done. Otherwise, If O does not contain bottle 1, then:

O' =..

....

....

....

....

As mentioned in the lecture, once we prove the claim about the first greedy choice, the rest of it follows by induction. We can apply the claim to replace the second bottle in O with the second greedy choice, and so on, and we'll end up with a solution made entirely of greedy choices. At that point, it follows that the greedy solution cannot have more bottles than O —–basically the optimal solution transformed by replacing every bottle with the greedy choice, would have contained all pills by then. Formally, the proof follows by induction on the number of bottles in O.

Note that in this case there may be more than one optimal solution, so we do not get a contradiction. There can be optimal solutions that do not contain the greedy choices; but the claim says that the greedy solution is optimal as well.

Additional problems–OPTIONAL

1. Pharmacist problem: Give pseudocode for a non-recursive dynamic programming algorithm for computing the optimal cost. Analyze its running time.

2. Pharmacist problem: Describe how to augment the algorithm in order to find the bottles used in the optimal solution.

3. Pharmacist problem: Assume $n = 3, W = 7$, and $c[1] = 3; c[2] = 5; c[3] = 7$, and $p[1] = p[2] = p[3] = 1$. Call MinPill(3, 2) and show what entries in the table are filled, and their values.

4. Consider the activity selection problem: Given a set of n activities each with its start and finish times, $\{s_i, f_i\}$, find the maximum number of non-overlapping activities.

   Consider the following greedy algorithm: for each activity compute how many activities it overlaps; pick the activity with smallest number of overlaps; repeat.

   Prove that this algorithm is wrong by showing a counter-example where it fails to produce an optimal solution.