# CS 161 (Stanford, Winter 2026) Section 8

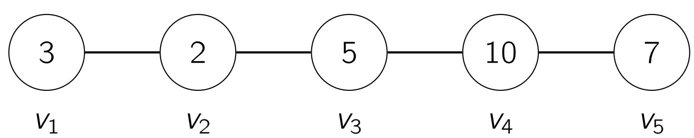
## 1 LCS Warm-up

1. Given sequences $X = \texttt{ABCB}$ and $Y = \texttt{BDCAB}$, fill in the LCS dynamic programming table $C[i, j]$ below, where $C[i, j]$ is the length of the LCS of $X[1 : i]$ and $Y[1 : j]$. Then use the completed table to recover an actual longest common subsequence.

| | $\emptyset$ | B | D | C | A | B |
|---|---|---|---|---|---|---|
| $\emptyset$ | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | | | | | |
| B | 0 | | | | | |
| C | 0 | | | | | |
| B | 0 | | | | | |

## 2 DP Exercises

1. Consider a path graph with 5 vertices, where each vertex $v_i$ has weight $w_i$ as shown. An **independent set** is a subset of vertices such that no two are adjacent. Find a maximum weight independent set.




   (a) Define $A(i)$ to be the weight of the maximum weight independent set considering only vertices $v_1, \ldots, v_i$. Write a recurrence for $A(i)$.

   (b) Compute $A(i)$ for $i = 0, 1, \ldots, 5$. Which vertices are in the optimal set?

   (c) What is the runtime of the resulting algorithm?

2. In how many ways can you tile a $2 \times n$ board using $1 \times 2$ dominoes (placed either horizontally or vertically)?

   (a) Let $T(n)$ be the number of tilings of a $2 \times n$ board. Write a recurrence for $T(n)$ and justify it.

   [**Hint:** *Consider how the rightmost column can be filled.* ]

   (b) Compute $T(n)$ for $n = 1, 2, \ldots, 7$.

   (c) Does this recurrence look familiar?

## 3 Activity Selection: Greedy Strategies

The activity selection problem asks: given $n$ activities with start and finish times, find a maximum-size set of non-conflicting activities. In lecture, we saw that always selecting the activity with the earliest **finish time** (that doesn't conflict) gives an optimal solution.

For each of the following alternative greedy strategies, give a counterexample showing it does not always produce an optimal solution.

1. Always select the activity with the earliest **start time** that doesn't conflict with already-selected activities.
2. Always select the activity with the **shortest duration** that doesn't conflict with already-selected activities.

## 4 Minimum Refueling Stops

You need to drive from city $A$ to city $B$, a total distance of $L$ miles. Your car can travel at most $D$ miles on a full tank. Gas stations are located at distances $g_1 < g_2 < \cdots < g_n$ from city $A$. You start with a full tank. You are guaranteed that the distance between any two consecutive stops is at most $D$ (where city $A$ and city $B$ are also considered stops), so it is always possible to reach city $B$.

Design an algorithm that finds the minimum number of refueling stops needed to reach city $B$.

(a) Describe a greedy strategy for this problem.

(b) Apply your strategy with $L = 20$, $D = 7$, and stations at positions $3, 5, 8, 12, 14, 17$. List the positions where you stop to refuel.

(c) Prove your greedy strategy is optimal.

[**Hint:** *Show by induction that after k stops, the greedy car is always at least as far along as in any other solution.* ]