Week 8 Lab

Dynamic programming

Collaboration level 0 (no restrictions). Open notes.

1. **Fibonacci:** The first example this week was computing the n-th Fibonacci number. Java code for this problem was sent on Slack — run through it.

2. **Board game:** The second example this week was the board game problem. Write Java code (similar to Fib.java) to implement both the the DP-recursive and the DP-iterative solutions.

   Write a function, which takes as argument a table of solutions (filled by DP) and computes the full solution, ie the set of jumps corresponding to the optimal cost.

3. **The house robber problem**[1]**:** Imagine you are a professional robber and you set your eye on a block of houses to rob. Each house i has a non-negative $v(i)$ worth of value inside that you can steal (e.g. a stash of money in the house). However, the security cameras are set up and connected so that if two adjacent houses are broken in the same night, the system will alert the police.

   Given a number of houses on the block and a list of non-negative integers representing the amount of money in each house, describe an algorithm to find the maximum amount of money you can rob in one night from the block without alerting the police. Extend your solution to give the the houses associated with this max robbery.

   ```
   Example 1: v = [1, 2, 3, 1]

   Max amount you can rob is 4 (rob house 1 and 3)

   Example 2:  v = [2, 7, 9, 3, 1]

   Max amount you can rob is 12 (rob house 2, 9 and 1)

   Example 3:  v = [6, 7, 1, 3, 8, 2, 4]

   Max amount you can rob is: 19 (rob houses  6, 1, 8 and 4)

   Example 4:  v = [5, 3, 4, 11, 2]

   Max amount you can rob is 16 (rob house 11 and 5)
   ```

[1] The first in our series of applications in the real world; also Leetcode #198