Week 8 Lab

Module 4: Techniques

Collaboration level 0 (no restrictions). Open notes.

1. **Rod Cutting:** One of the examples this week was the rod cutting problem. Download the notebook *RodCutting.ipynb*, and test your understanding of this problem by running through it step by step and reflecting on the answers. Fill in the missing function.

2. **Longest `true` interval**: Suppose we are given an array $A[1..n]$ of booleans. We want to find the longest interval $A[i..j]$ such that every element in the interval is true – in other words, $A[i], A[i+1], .., A[j]$ are all true.

   (a) Try solving this problem without DP. What running time do you get?

   (b) Now consider the following approach: for any x such that $1 \leq x \leq n$, define $G(x)$ to be the length of the longest suffix[1] of $A[1..x]$ that is all true. In other words, $G(x)$ is the largest integer l such that $A[x-l+1], A[x-l+2], .., A[x]$ are all true, or 0 if $A[x]$ is false.

   Show a recursive formula for $G(x)$. Your recursive formula for $G(x)$ should be defined in terms of the solution(s) to smaller/easier subproblem(s), and do not forget the base cases.

   (c) Describe a dynamic programming algorithm to compute $G(n)$.

   (d) How do you use $G(x)$ to find the longest true interval in A? How long does it take?

3. **The house robber problem[2]:** 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]
   ```

[1] An array $B[1..m]$ is a suffix of an array $A[1..n]$ if $A[n-k] = B[m-k]$ for $0 \leq k < m$

[2] Leetcode #198

```
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)
```

Additional problems —Optional

1. Let's play a game where the input is a sequence $A[1], .., A[n]$ of numbers. At each turn, you can either (1) take the first number in the sequence, add it to your score, and then delete that element from the sequence; or (2) you can delete the first number in the sequence, take the second number in the sequence and add it to your score, and then finally delete the second number from the sequence. You keep playing until all of elements in the sequence have been deleted.

   Find a choice of moves that maximize your score.

   For instance, if the sequence is $5, -2, 4, -4, -1, 5$, your best choice of moves is "take first, take second, take second, take first", for a total score of $5 + 4 + -1 + 5 = 13$.

   Suppose we want to solve this problem using dynamic programming. Let $f(i)$ denote the best score attainable starting from the sequence $A[i], .., A[n]$. Write a recursive formula for $f(i)$.