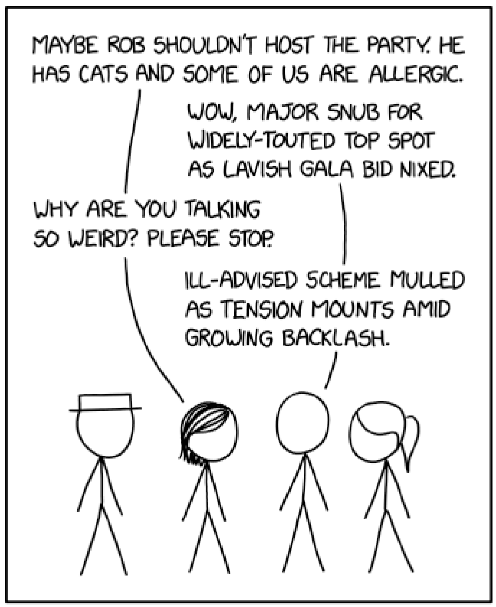
CS140 - Assignment 5

Due: Sunday, 2/25 at ~~10pm~~ 11:59pm



MY PROJECT TO SPEAK ONLY IN WEIRD HEADLINE WORDS DIDN'T LAST LONG.

https://xkcd.com/2584/

For this assignment, you may (and are encouraged to) work with a partner.

For the dynamic programming solutions, in addition to the algorithm, make sure to explicitly state:

- What the table looks like (size and range of values).
- How you initialize the table, i.e., any starting values
- How you fill the table in, i.e., what indices you start at and how you proceed.
- Where the final answer is.

This can either be done with pseudocode or in plain language.

1. CS Snack [**10 points**]

   The CS liaisons have asked for your help with shopping for the CS snack. There are n possible snack options that you can choose from and you have d dollars to spend. Each snack option has been voted on with how excited the students are about the snack: $e_1, e_2, ..., e_n$, with higher being better. Each snack also has a price, $p_1, p_2, ..., p_n$. Your job is to pick which snacks so as to optimize the sum of the excitement of the snacks chosen while spending $\leq d$ dollars. Note that you can choose a given snack option multiple times.

   (a) [**8 points**] Give a dynamic programming solution that determines the maximum excitement sum, given the budget constraints (you don't need to explicitly state what snacks are chosen, just the overall sum of the excitement). Make sure to explicitly state the size of the table, which elements you fill in first, how you fill in the table, and where the solution is found.

   (b) [**2 points**] State the running time of your algorithm in terms of n the number of snack options and d the budget.

2. Words [**12 points**]

   You are given a string of characters $S = s_1, s_2, ..., s_n$ where all non-alphabetic characters have been removed (e.g. "thisisasentencewithoutanyspacesorpunctuation") and a function $\textsc{dict}(w, i, j)$, which takes as input a string w and two indices i and j and returns *true* if the string $w_{i...j}$ is a dictionary word and $false$ otherwise.

   (a) [**10 points**] Give a dynamic programming solution that determines whether the string S consists of a sequence of valid dictionary words. Make sure to explicitly state the size of the table, which elements you fill in first, how you fill in the table, and where the solution is found.

   (b) [**2 points**] State the running time of your algorithm assuming calls to $\textsc{dict}$ are $O(1)$.

3. Party [**15 Points**]

   You've been asked to design an algorithm for deciding who to invite to a company party. The structure of the company can be described by a tree as follows: the CEO is at the root, below the root are VPs, below them are directors, below them are manages, etc., etc., until you get down to the leaves (summer interns). The tree is not necessarily binary; some non-leaf nodes may have one "child", others two, and others even more.

   To make the party fun, we won't invite an employee along with their immediate supervisor (their parent in the tree). In addition, each person has been assigned a positive real number called their *coefficient of fun*. The goal is to invite employees so as to maximize the total sum of the coefficients of fun of all invited guests, while not inviting an employee with their immediate supervisor.

   (a) [**4 points**] Describe a recursive algorithm for this problem (i.e. non-dynamic programming). Assume that the tree is represented as a collection of nodes with links from parents to children and also from children to parents. The tree is passed to you by giving you a reference to the root.

(b) **[8 points]** Describe a DP algorithm for this problem. You may assume that each of the n nodes in the tree has a unique number between 1 and n associated with it. You may also assume that you have a function that will give you a list (array or linked list) of all of the leaves in the tree in time $O(n)$.

(c) **[3 points]** State the running time of your approach with respect to n the number of employees and k the maximum number of children any node has.