cs4102 - algorithms - s'16 — abhi shelat
due fri mar 4, 2016 at 5p via submission site

H4.as4bd

You may collaborate with other students on the homework but you must submit your own individually written solution, identify your collaborators, and acknowledge any external sources that you consult. Do no submit a solution that you cannot explain to me.

problem 1 *Day Trader's Decisions*

A day trader wants to trade $n \cdot 100$ shares of the stock AAPL in blocks of 100. Her risk manager forbids her from short-selling; in other words, she must own a block of shares before selling them. At the end of the day, she wants to own zero shares of AAPL. How many ways can she do her trading? For example, when $n = 2$, she can do "BUY, SELL, BUY, SELL" or "BUY, BUY, SELL, SELL" and so there are 2 ways. Present your answer as a recurrence, explain why it is correct.

problem 2 *Price Run*

Given a sequence of closing stock ticker prices $p_1, p_2, \ldots, p_n$, devise an $O(n^2)$ algorithm that finds the longest (not necessarily consecutive) streak of prices that increase or stays the same. For example, given the prices $2, 5, 2, 6, 3, 3, 6, 7, 4, 5$, there is the streak $2, 5, 6, 6, 7$ of prices that increase or stay the same, but an even longer sequence (streak) is $2, 2, 3, 3, 4, 5$.

(Challenge: by using both dynamic programming and binary search, you can solve this problem in $O(n \log n)$ time.)

problem 3 *Chompo bar*

You are given an $n \times m$ chompo bar. Devise an algorithm A that takes as input (n, m) and returns the minimal number of cuts needed to divide the bar into perfect squares of either 1x1, 2x2, 3x3, …, jxj. With each cut, you can split the bar either horizontally or vertically. For example, $A(2, 3) = 2$ because 2x3 → (2x2, 1x2) → (2x2, 1x1, 1x1).

problem 4 *Tug of War*

We want to play *roughly fair* tug of war in cs4102. You are given an array that holds the weights of n people in the class $W = (w_1, w_2, \ldots, w_n)$. Your goal is to divide the n people into two teams such that the total weight of the two teams is equal or as close as possible to equal. Describe such an algorithm and give its running time. The total number of people on each team should differ by at most 1. Assume that M is the maximum weight of a person, i.e., $\forall i, w_i \leq M$. The running time should be $O(n^3 M)$. The output should be the list of people on each team and the difference in weight between the teams.