# CSE 421 Winter 2021
# Homework 5

**Due: Friday, February 19, 2021, 6:00 pm**

**Problem 1:**
Given a 1-dimensional array of $n$ real numbers, on Homework 4 you were asked to use divide and conquer to solve the problem of finding the following three items:

- the contiguous sub-array of the input with maximum sum.
- the prefix of the array with maximum sum, and
- the suffix of the array with maximum sum.

Give an $O(n)$ *dynamic programming* algorithm to find all three of these segments in an array of $n$ numbers and compute their associated sums.

**Problem 2:**
In typesetting systems, like the LaTeX system in which this problem set is formatted, the input for paragraphs is given in blocks of plain text consisting of words and punctuation separated by blanks and new lines. Formatted paragraphs are produced by laying out this text using fixed maximum width $L$ per line by figuring out where to break each line of text and continue the paragraph on the next line. An innovation of the TeX system on which LaTeX is based was to phrase paragraph formatting as an optimization problem. which produced superior paragraph formatting and has been adopted by many other word processing systems.

We consider a simplified version of this optimization problem in which words are considered indivisible units of fixed width (no hyphenation) and punctuation is considered part of the word is next to.

In our version, the input consists of a sequence of $n$ positive real numbers $w_1, \ldots, w_n$ representing the lengths of that the words would occupy in a given font in order, together with a positive real number $b$ representing the minimum width of a blank in that font, and another real number representing the allowed width $L$ of each line. (You can assume that each $w_i \leq L$.)

A legal formatting for this text consists of a sequence of integers $0 = e_0 < e_1 < \ldots < e_k = n$ for some $k$, where the $i$-th line of the paragraph consists of the words numbered $e_{i-1} + 1, \ldots, e_i$ separated by blanks. (That is, $e_i$ is the number of the word that ends the $i$-th line in the paragraph and the number of blanks in a line is 1 less than the number of words in it.) It is required that no line be longer than $L$, so if a line consists of words $t$ through $u$ then

- $w_t + b + w_{t+1} + b + \ldots + b + w_u \leq L$.
- The *slack* of this line is the difference between the left and right sides of this inequality.

An optimum formatting for this input is one that minimizes the sum of the squares of the slacks of its lines.

Give an efficient algorithm that finds an optimal formatting from the inputs $w_1, \ldots, w_n, b, L$.

**Problem 3:**
In this problem, we have $n$ items with positive integer values $v_1, \ldots, v_n$ that are all different. There is also a positive integer target value $V$. The goal in this question is to figure out the *number of different subsets $S$* such that $\sum_{i \in S} v_i = V$.

1. Give an algorithm with running time $O(nV)$ to produce this answer.
2. How would you modify your solution to keep the same runtime and use space $O(V + n)$?

**Problem 4 (Extra Credit):**
In this problem you are given as input a rooted binary tree $T = (V, E)$ with each leaf $w$ labelled by a symbol $s_w$ from some fixed alphabet $A$, as well as a two-dimensional non-negative array of costs indexed by pairs of elements of $A$ such that $c[s, t]$ is the cost of changing symbol $s$ to symbol $t$ and this satisfies $c[s, s] = 0$ and $c[s, t] = c[t, s]$ for all $s, t$ in $A$.

Design an efficient algorithm to label each internal node $v$ of the binary tree $T$ by a symbol $s_v$ from $A$ so as to minimize the sum over all $(u, v)$ in $E$ of $c[s_u, s_v]$.

(This kind of problem arises in computational biology when one wishes to assess a potential evolutionary tree and reconstruct properties of potential ancestral species given this tree. In this case each "symbol" might be a very short sequence - or even just a letter - of DNA or protein that might at a given position within a longer sequence, and $c[s, t]$ is the cost of the evolutionary change in going from $s$ to $t$.)

**Problem 5 (Extra Credit):**
In Problem 2, the quantity we wanted to optimize was the sum of the squares of the slacks of all the lines. In normal text formatting of paragraphs we want to allow the last line to be shorter without penalty. Describe how to find the optimum solution efficiently when the goal is to optimize the sum of the squares of slacks of all but the last line.