CS 473: Algorithms, Fall 2010
HW 5 (due Tuesday, October 12)

This homework contains four problems. Read the instructions for submitting homework on the course webpage. In particular, make sure that you write the solutions for the problems on separate sheets of paper; the sheets for each problem should be stapled together. Write your name and netid on each sheet.

Collaboration Policy: For this homework, Problems 1-2 can be worked in groups of up to 3 students each.

Problem 0 should be answered in Compass as part of the assessment HW5-Online and should be done individually.

0. (10 pts) HW5-Online.

1. • (30 pts) In a word processor the goal of “pretty-printing” is to take text with a ragged right margin, like this,

   Call me Ishmael.
   Some year ago, never mind how long precisely, having little or no money in my purse, and nothing particular to interest me on shore, I though I would sail about a little and see the watery part of the world.

   and turn it into text whose right margin is as “even” as possible, like this

   Call me Ishmael. Some year ago, never mind how long precisely, having little or no money in my purse, and nothing particular to interest me on shore, I though I would sail about a little and see the watery part of the world.

To make this precise enough for us to start thinking about how to write a pretty-printer for text, we need to figure out what it means for the right margins to be “even”. So suppose our text consists of a sequence of words, \( W = \{w_1, w_2, \ldots, w_n\} \), where \( w_i \) consists of \( c_i \) characters. We have a maximum line length of \( L \). We will assume we have a fixed-width font and ignore issues of punctuation or hyphenation.

A formatting of \( W \) consists of a partition of the words in \( W \) into lines. In the words assigned to a single line, there should be a space after each word except the last; and so if \( w_j, w_{j+1}, \ldots, w_k \) are assigned to one line, then we should have

\[
\left[ \sum_{i=j}^{k-1} (c_i + 1) \right] + c_k \leq L.
\]
We will call an assignment of words to a line valid if it satisfies this inequality. The difference between the left-hand side and the right-hand side will be called the slack of the line—that is, the number of spaces left at the right margin.

Given a partition of a set of words \( W \), the penalty of the formatting is the sum of the squares of the slacks of all lines (including the last line). Give an efficient algorithm to find a partition of a set of words \( W \) into valid lines, so that the penalty of the formatting becomes minimized.

1. (20 pts) Implement your iterative algorithm in C or C++ or Java. The input to your program will be \( L \ n \ c_1 \ c_2 \ldots \ c_n \) on a single line: \( L \) is the number of characters per line allowed, \( n \) is the number of words in the text and each \( c_i \) is the length of a word \( w_i \) in our text, \( W = \{w_1, \ldots, w_n\} \), in order. The output should be the penalty of an optimal formatting as well as the lengths of all lines in an optimal formatting of the words. You need to submit a print out of the code along with the output of your code on a set of inputs that we will provide on the website.

2. (40 pts) Let \( G = (V, E) \) be an undirected graph. A subset \( S \subseteq V \) of nodes in \( G \) is called a dominating set if for all \( v \in V \), we have \( v \in S \) or there is some node \( u \in S \) such that \((u,v) \in E\). In other words every node in \( V \setminus S \) is connected by an edge to some node in \( S \). Given non-negative weights \( w(v) \) on the nodes of \( V \) the goal is to find a minimum-weight dominating set in \( G \). This problem is known to be NP-Hard in general graphs. Describe a polynomial time algorithm for this problem when \( G \) is a tree.