CS 473, Fall 2017
Homework 2 (due September 20 Wednesday at 8pm)

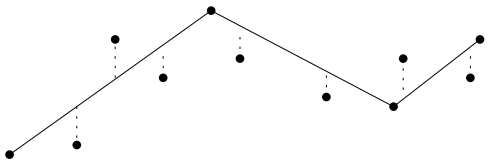
You may work in a group of at most 3 students. Carefully read `http://engr.course.illinois.edu/cs473/policies.html` and `http://engr.course.illinois.edu/cs473/integrity.html`. One member of each group should submit via Gradescope.

1. [*20 pts*] We are given a sequence of points $p_1 = (x_1, y_1), \ldots, p_n = (x_n, y_n)$ sorted from left to right (i.e., $x_1 < x_2 < \cdots < x_n$) and a number k between 1 and n. We want to find a minimum-error polygonal path from p_1 to p_n with k edges that goes from left to right, where the *error* of a path is the sum of the vertical distances of the points $p_1, \ldots, p_n$ to the polygonal path. (The motivation is in finding a piecewise linear function that best "fits" the data points.)

   More precisely, we want a subsequence $p_{i_0}, p_{i_1}, p_{i_2}, \ldots, p_{i_k}$ where $1 = i_0 < i_1 < i_2 < \cdots < i_{k-1} < i_k = n$, minimizing the error function $f(i_0, i_1) + f(i_1, i_2) + \cdots + f(i_{k-1}, i_k)$, where

$$f(a,b) = \sum_{i=a+1}^{b-1} \left| (y_i - y_a) - \left(\frac{y_b - y_a}{x_b - x_a} \right) (x_i - x_a) \right|$$

   represents the sum of the vertical distances of the points $p_{a+1}, \ldots, p_{b-1}$ to the line through $p_a p_b$.

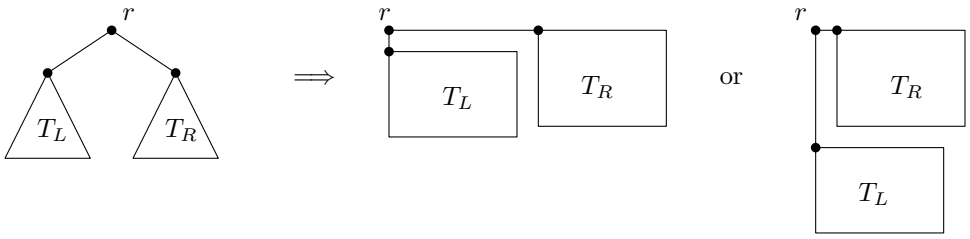


($k = 3$)

   Present an efficient (polynomial time) algorithm for this problem using dynamic programming. Include the following steps: (a) first define your subproblems precisely, (b) give base cases, (c) derive the recursive formula, (d) write pseudocode, (e) describe how to output the optimal subsequence, and (f) analyze the running time (as a function of n and k).

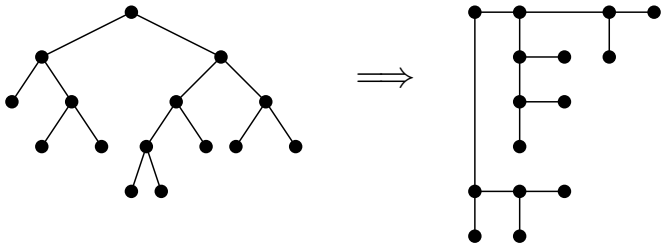
2. [*20 pts*] This problem concerns how to draw a binary tree, where vertices are to be placed at integer points in the plane and edges are to be drawn horizontally or vertically without crossings. We define a class of *neat drawings* (and their *widths* and *heights*) recursively as follows. Let T be a binary tree with root r and left and right subtrees T_L and T_R. We can combine a neat drawing of T_L with width W_L and height H_L and a neat drawing of T_R with width W_R and height H_R, to obtain a neat drawing of T, in 2 possible ways:

- *horizontal rule*: place the root of T_R at $W_L + 1$ units to the right of r, and place the root of T_L at 1 unit below r; the resulting drawing of T has width $W_L + 1 + W_R$ and height $\max\{H_L + 1, H_R\}$; or
- *vertical rule*: place the root of T_L at $H_R + 1$ units below r, and place the root of T_R at 1 unit to the right of r; the resulting drawing of T has width $\max\{W_L, W_R + 1\}$ and height $H_L + 1 + H_R$.



For a tree with one node, there is a neat drawing of width 0 and height 0. The *area* of a drawing is defined as its width times height.

The example below shows one neat drawing of a binary tree, with width 4 and height 5 (and thus area 20):



Given a binary tree with n nodes, we want to find a neat drawing that minimizes the area. Present an efficient (polynomial time) algorithm for this problem using dynamic programming. Include the following steps: (a) first define your subproblems precisely, (b) give base cases, (c) derive the recursive formula, (d) write pseudocode, and (e) analyze the running time. For this problem, your algorithm only needs to output the minimum area, and not the drawing itself.

[Suggestion: solve a more general problem, of finding the minimum-height drawing for any given width.]