Problem 1. (6 points) Consider a binary tree $T$ whose nodes are labeled with alphabetic characters. Suppose you are told that $e/i/a/j/d/b/f/g/h/c$ is an inorder traversal of this tree, and that $a/j/i/e/f/h/g/b/c/d$ is a postorder traversal of this tree. Does this information allow you to uniquely reconstruct a binary tree? If yes, draw such a tree. If not, explain why either no such tree exists, or why more than one such tree exists.

Problem 2. (8 points) Let $T_1$ be a binary tree, and let $T_2$ be a version of $T_1$ which is threaded in preorder. Assume the existence of fields LTAG and RTAG (with ‘t’ and ‘l’ as possible values denoting thread/link) respectively which are used to check if the left/right child is a link or a thread. Suppose you are given a pointer $P$ to a node in tree $T_2$. Is it possible to find the preorder successor of the node $P$ is pointing to? If this is possible, give an algorithm in pseudo-code. Otherwise explain why it is not possible.

Problem 3. (16 points) You are familiar with quadtrees and 2-d trees to represent 2-dimensional data. In this problem, we define a new data structure to represent 2-d data called a ternary-tree. As in the case of quadtrees and 2-d trees, the root of a ternary tree represents a rectangular region $R$. Every node $N$ has fields $N.x, N.y$ representing a point in 2-space, fields $N.left, N.right, N.top, N.bot$ denoting the extremities of the rectangle (implicitly) represented by node $N$ (and in which the point $N.x, N.y$ lies), and three link fields $N.NW, N.NE, N.SO$ denoting the regions obtained by splitting the rectangle specified by $N.left, N.right, N.top, N.bot$ into three parts — the northwest, northeast, and southern regions obtained by first drawing a horizontal line through $N.x, N.y$ to obtain two parts — the southern part (pointed to by $N.SO$ and the northern part. The northern part is then split in two by drawing a vertical line through $N.x, N.y$ to obtain the regions associated with the $N.NW, N.NE$ children of $N$. Assume all regions are closed on the left and bottom, and open on the top and right.

1. (2 points) Suppose you know all the fields of a node $N$. How would you compute $M.left, M.top$ from the fields of $N$ when $M$ is the $SO$ child on $N$.

2. (4 points) Construct the tree obtained when you insert the following points in the order shown: $(50,50), (25,30), (45,75),(20,35)$. Show all fields of all nodes.

3. (10 points) Write, in pseudo code, an algorithm that takes as input, the root $T$ of a ternary tree, and a point $(a, b)$, and returns the “furthest neighbor” of $(a, b)$ in $T$, i.e. it returns that point in $T$ which is furthest away from $(a, b)$. When writing this algorithm, you may assume the existence of functions called $mindist$, $maxdist$ which take three arguments - an $x$ coordinate, a $y$-coordinate, and a pointer to a node $N$ in the ternary tree. $mindist(a, b, N)$ returns $\min \{d((a, b), (x, y)) | (x, y) \text{is in the rectangular region associated with } N \}$ while $maxdist(a, b, N)$ returns $\max \{d((a, b), (x, y)) | (x, y) \text{is in the rectangular region associated with } N \}$. $d(a, b), (x, y))$ here is the distance between $(x, y)$ and $(a, b)$. 
