# Tarzan of The Apes

Problem category: Graph theory
Expected difficulty: 2000

**Solution**

As you might have guessed, the most straightforward solution to this problem is Dijkstra's algorithm. Indeed, we can construct a digraph in which the edges indicate that a target tree can be reached from a source tree if the Euclidean distance between them is less than or equal to the height of the source tree. Then, we apply the usual Dijkstra's algorithm, which runs in $O(n^2)$ for dense graphs.

This would be fine if the constraints of the problem allowed us to allocate enough memory to maintain the graph, which amounts to $O(n^2)$ to hold all the edges. Alas, we have to do without it. We just have to be mindful of a few quirks.

First, we need to sort the trees by their $x$-coordinate. This will allow us to iterate over neighbouring trees in the $x$-axis, and stop short of evaluating pairs of trees when their horizontal distance becomes greater than the height of the current tree.

Second, we must use the actual Euclidean distance (which implies floating-point square root computation). Otherwise, the algorithm would yield an incorrect result.

Finally, we should skip a target tree that has been reached with a traveled distance less than that of the current tree (since it won't get a better score from the latter), to save the cost of a square root computation.

**Complexity**

Since we need to evaluate pairs of trees, the time complexity is $O(n^2)$. On the other hand, since we are not building a graph to perform the Dijkstra algorithm, the overall memory complexity is $O(n)$.