Building a Balanced kd-Tree

1. **kdtree_create(points)**
   - `sort_x` <- points sorted by x (lon)
   - `sort_y` <- points sorted by y (lat)
   - `root` <- `kdtree_create_helper(sort_x, sort_y)`

2. **kdtree_create_helper(sort_cut, sort_other)**
   - if `sort_cut` has 0 points, return NULL
   - `node` <- new node containing middle point from `sort_cut`
   - `cut_left` <- 1st half of `sort_cut`
   - `cur_right` <- 2nd half of `sort_cut`
   - `other_left` <- pts from `sort_other` to left of middle
   - `other_right` <- pts from `sort_other` to right of middle
   - `node->left` <- `kdtree_create_helper(other_left, cut_left)`
   - `node->right` <- `kdtree_create_helper(other_right, cut_right)`
   - return `node`

3. `T(n) = 2T(\frac{n}{2}) + \Theta(n)`
4. `T(n) = \Theta(n \log n)`

This is the same as for mergesort.

**Diagram:**
- Points are sorted and split recursively to create a balanced tree.
- The cut dimension alternates between x and y.
- The tree structure is shown with points and recursive calls.