Generally, in order to produce a run time of O(log n) we need to perform a constant amount of work, throw away half of the input, and continue recursively on what's left (Klienberg, 243).
This strategy applies nicely to this problem. Given a binary tree T, we first probe the values of the root and its two children. If the left child has a smaller value than the root, then we recursively call this method on the left subtree. If the right subtree is smaller than the root, then we recursively call this method on the right subtree. Otherwise, the root is the local minimum and we return that.
This takes O(log n) because there can be at most log n recursive calls, and each probe takes constant time.