```
1: function NEARESTNEIGHBORS(point:point, node:Node, candidates:priority queue, distMin:float, k:int)
2:     if node = null or node.visited = true then                ▷ Stop condition for recursive function
3:         return
4:     end if
5:     dist ← distance(point, node)
6:     if distdistMin then
7:         add(candidates, [dist, point])
8:         sort(candidates)                                                    ▷ according to dist
9:         distMin ← candidates[−1]                          ▷ biggest of the minimum distances
10:    end if
11:    if length(candidates) > k then
12:        remove last element of candidates
13:    end if
14:    if point[node.dimension] < node.value[node.dimension] then
15:        NEARESTNEIGHBORS(point, node.left, candidates, distMin, k)
16:        if |point[node.dimension] − node.value[node.dimension]| ≤ distMin then
17:            NEARESTNEIGHBORS(point, node.right, candidates, distMin, k)
18:        else
19:            node.right.visited ← true                                   ▷ pruning right subtree
20:        end if
21:    else
22:        NEARESTNEIGHBORS(point, node.right, candidates, distMin, k)
23:        if |point[node.dimension] − node.value[node.dimension]| ≤ distMin then
24:            NEARESTNEIGHBORS(point, node.left, candidates, distMin, k)
25:        else
26:            node.left.visited ← true                                    ▷ pruning left subtree
27:        end if
28:    end if
29:    node.visited ← true
30: end function
```