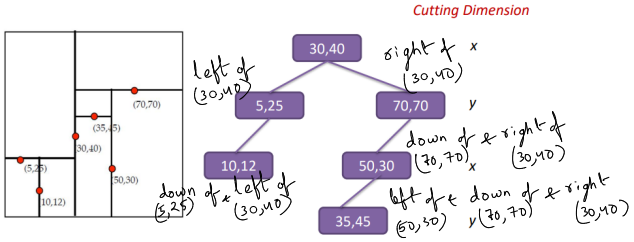
KD Tree:

in a ways extension to BST, while inserting pick median , it will help build balanced tree

(30,40) , (5,25) , (10,12) , (70,70) , (50,30) , (35,45)

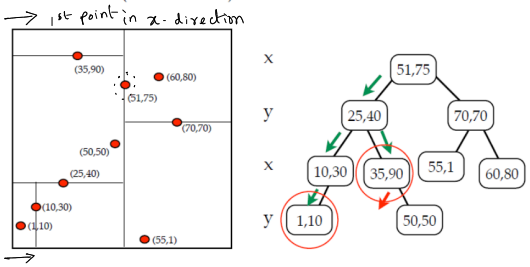



```
insert(Point x, KDNode t, int cd) {
   if t == null
      t = new KDNode(x)
   else if (x == t.data)
      // error! duplicate
   else if (x[cd] < t.data[cd])
      t.left = insert(x, t.left, (cd+1) % DIM)
   else
      t.right = insert(x, t.right, (cd+1) % DIM)
   return t
}
```

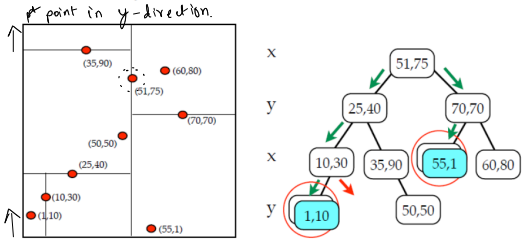
```
Point findmin(Node T, int dim, int cd):
   // empty tree
   if T == NULL: return NULL

   // T splits on the dimension we're searching
   // => only visit left subtree
   if cd == dim:
      if t.left == NULL: return t.data
      else return findmin(T.left, dim, (cd+1)%DIM)
   // T splits on a different dimension
   // => have to search both subtrees
   else:
      return minimum(
         findmin(T.left, dim, (cd+1)%DIM),
         findmin(T.right, dim, (cd+1)%DIM)
         T.data
      )
```

FindMin(x-dimension):

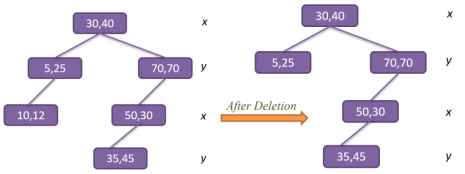


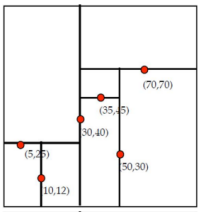
FindMin(y-dimension):

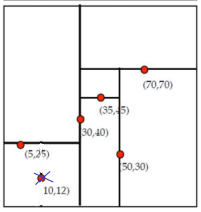


K-d Tree – Deletion

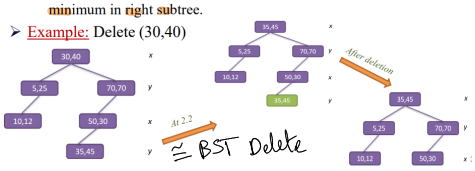
- Case 1: If the current node contains the point to be deleted and the node to be deleted is a leaf node, just delete it.
- Example: Delete (10,12)

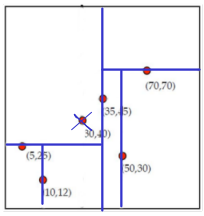






- Case 2: If the current node contains the point to be deleted and the node to be deleted has a right subtree, then:
  - 2.1 Find the minimum of current node's dimension in right subtree i.e. FindMin(currentNode's dim) in right subtree.
  - 2.2 Replace the node with the node found in 2.1 and recursively delete minimum in right subtree.
- Example: Delete (30,40)

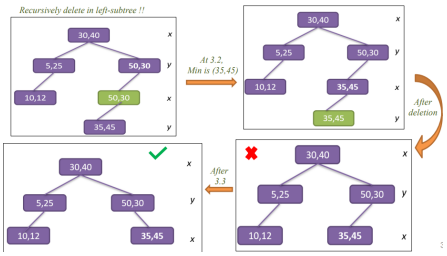


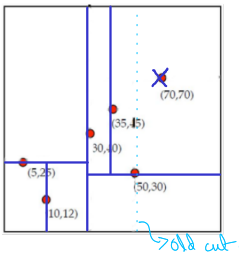


Case 3: If the current node contains the point to be deleted and the node to be deleted has a left subtree (no right subtree), then:

- 3.1 Find the minimum of current node's dimension in left subtree i.e. FindMin(currentNode's dim) in left subtree.
- 3.2 Replace the node with the node found in 3.1 and recursively delete minimum in left subtree.
- 3.3 Make new left subtree as right child of current node [*Caution!*]

Example: Delete (70,70)





old cut-dim of (50,30) previously (35,45) was to left but now it is above ∴ made it right child

Case 4: If the current node does not contain the point to be deleted, then:

- 4.1 If the node to be deleted is smaller than the current node on current dimension, recur for the left subtree. Otherwise, recur for the right subtree.
- 4.2 Eventually after 4.1, we will reach any of the other three cases. Use that case and perform deletion!

```
Point delete(Point x, Node T, int cd):
   if T == NULL: error point not found!
   next_cd = (cd+1)%DIM

   // This is the point to delete:
   if x = T.data:
      // use min(cd) from right subtree:
      if t.right != NULL:
         t.data = findmin(T.right, cd, next_cd)
         t.right = delete(t.data, t.right, next_cd)
      // swap subtrees and use min(cd) from new right:
      else if T.left != NULL:
         t.data = findmin(T.left, cd, next_cd)
         t.right = delete(t.data, t.left, next_cd)
      else
         t = null    // we're a leaf: just remove

   // this is not the point, so search for it:
   else if x[cd] < t.data[cd]:
      t.left = delete(x, t.left, next_cd)
   else
      t.right = delete(x, t.right, next_cd)

   return t
```

links:

https://www.cs.princeton.edu/courses/archive/spring18/cos226/demos/99DemoKdTree.pdf

https://www.youtube.com/watch?v=ivdmGcZo6U8

https://www.youtube.com/watch?v=Z4dNLvno-EY

https://cs.brynmawr.edu/Courses/cs246/spring2013/slides/14KDTrees.pdf