sources: wikipedia K-d_tree
You are asked to complete the pseudocode draft below to realize the KD tree construction over a set of points P = {p1,p2,...,p_n} and the coordinates of point p_i is (p_i.x,p_i.y). The algorithm should satisfy the following requirements:

1. Start with X-axis when choosing the first division.
2. Divide the space until there is at most one point in each sub-panel
3. Do not divide further if there is only one point left in a panel
4. Assume there exists a function mid(Points s, boolean byX_axis) that returns the point that is the middle point by either X_axis or Y_axis
5. Each KD tree node has a value field pointing to the dividing point, one left child and one right child.

Algorithm 1 Construct the KD tree from a set of points

---

1. ***procedure*** *KD(P, byX_axis) . ‣ If byX_axis is true, then divide the points by X_axis. Otherwise divide by Y_axis. P is the set of points.: //This algorithm constructs a KD tree based on a points set P and the first division is*
2. *by X_axis if the parameter byX_axis is true. It returns the KD root tree node.*
3.
4. if !p {return NULL} *//If there are no points in list, return NULL*
5. if P.len==1 {return root = P} //If there is only one point, return the one point as KD Tree
6. P.sort(X_axis); *//Sorts points*
7. p = mid(Points s, boolean byX_axis) *//Sets the correct median point by X axis*
8. KDnode root.point = p *//Sets KDnode to mid point*
9. Node *//Create node and construct subtrees*
10. (
11. left_child=KD(P[1...med]), !byX_axis), *//Creates left child from left half of P*
12. right_child=KD(P[med+(1…n), !byX_axis) *//Creates right child from right half of P*
13.);
14.
15. root = Node; *//Sets root to Node*
16. **return** root
17. **end procedure**