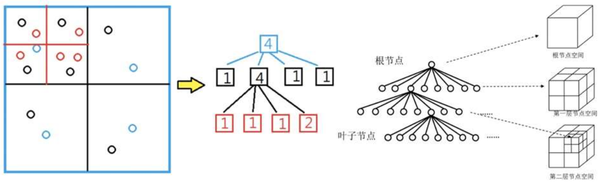
HW6 KD-Tree 小练习

简答（言之有理即可）

1.在构造 KD-Tree 时，如何消除多点共垂直、共水平的退化情况？

我认为其中一种方法可以是在构造 kd-tree 时，给每个点添加微小的随机扰动，但不修改其真实值，这样就能使其在进行划分的时候，避免多点共垂直，共水平的情况出现。

2.KD-Tree 相对于（二维）四分树、（三维）八分树，在什么情况下有什么优势？



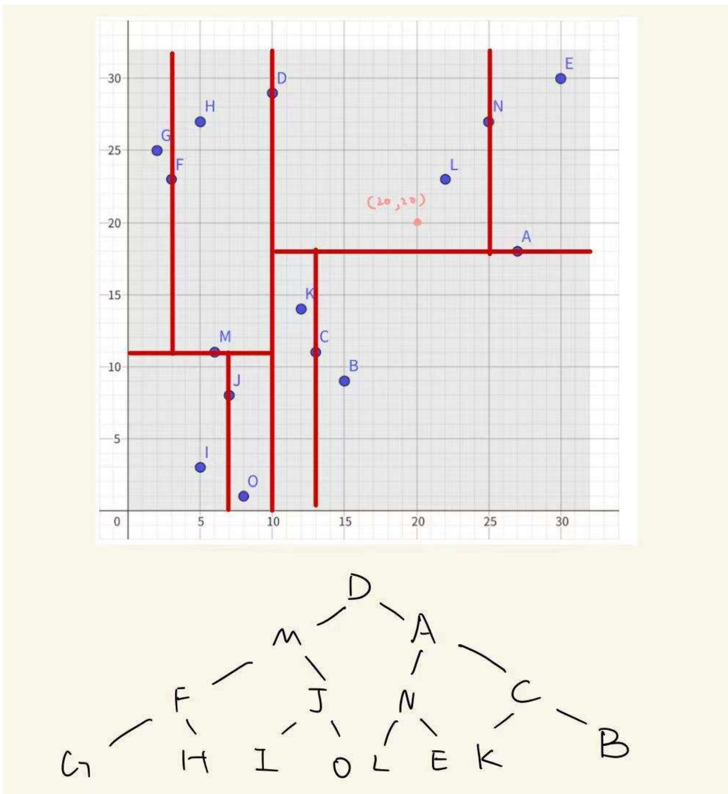
在需要查找最临近节点的情况下，通过最近邻查找算法，在 kd-tree 中，可以剪枝大量无需遍历的子树，其查找平均时间复杂度可以达到$O(n^{1-\frac{1}{k}})$，k 为维度。而（二维）四分树、（三维）八分树难以做到。同时，kd-tree 的存储比（二维）四分树、（三维）八分树更节省空间，在需要有效利用存储空间时，kd-tree 更有优势。

实践

在以下 [0,32]×[0,32] 的二维空间中（灰色区域），有 15 个点。

请回答以下问题:

1.将这 15 个点建立 KD-Tree（要求首先按照横轴划分）。



2.在构造的 KD-Tree 中，要查找离点 (20,20) 距离最近的点，请给出查找过程。

查找 D：发现(20,20)在 D 右侧，因此查找 A 分支

查找 A：发现(20,20)在 A 上侧，因此查找 N 分支

查找 N：发现(20,20)在 N 左侧，因此查找 L 分支

查找 L：发现没有孩子，更新 L 到(20,20)为最短距离，回退至 N

查找 N：发现(20,20)到 N 的 x 坐标的水平距离大于当前最短距离，回退至 A

查找 A：发现(20,20)到 A 的 y 坐标的竖直距离小于当前最短距离，因此查找 C 分支

查找 C：发现(20,20)在 C 右侧，因此查找 B 分支

查找 B：发现没有孩子，但是 B 到(20,20)的距离大于当前最短距离，不更新，回退至 C

查找 C：发现(20,20)到 C 的 x 坐标的水平距离大于当前最短距离，回退至 A

查找 A：发现左右孩子都查找过了，回退至 D

查找 D：发现(20,20)到 D 的 x 坐标的水平距离大于当前最短距离，终止查找

最终最近的点为 L