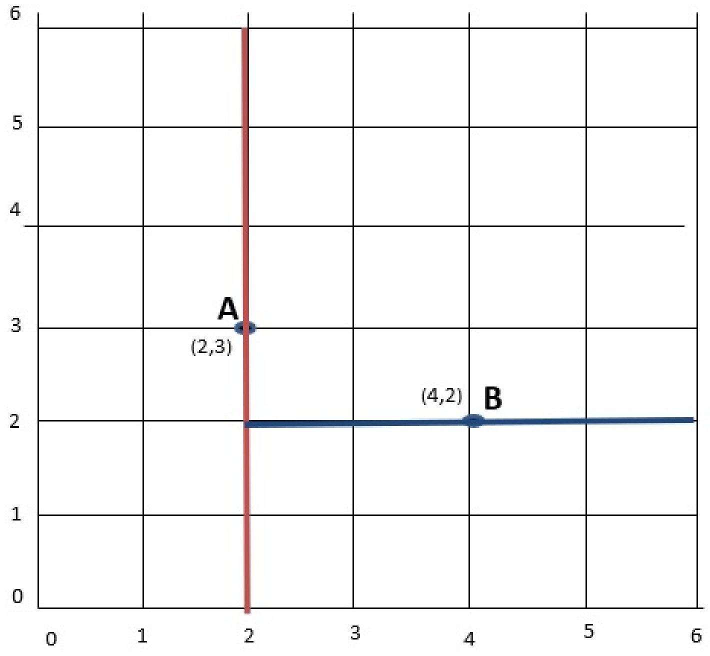
PRINCETON UNIVERSITY

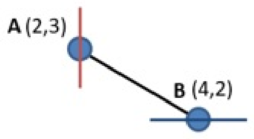
COS 226–Algorithms and Data Structures
Week 6: *Kd-Trees*

Version: March 28, 2018

Exercise 1 – Kd Trees

A. Suppose we are inserting the 2D points $[A(2,3), B(4,2), C(4,5), D(3,3), E(1,5), F(4,4)]$ into a Kd-Tree. We have shown the Kd-Tree after inserting the first two points including how the 2D plane is bisected as points are inserted. To the right we show the 2D binary tree that represents the Kd-Tree. Insert the other points into the Kd-Tree and show how the plane is bisected. Be careful when inserting $(4,4)$ as ties are treated the same as greater than. On the axes, draw each point as well as the horizontal or vertical lines that bisect the plane through each point.





B. List the bounding rectangles (RectHV) for each point in the table below. The bounding rectangle RectHV of A(2, 3) is given as

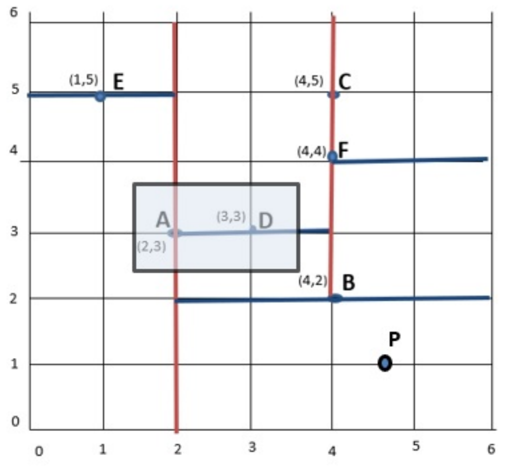
$[(-\infty, -\infty), (\infty, \infty)]$

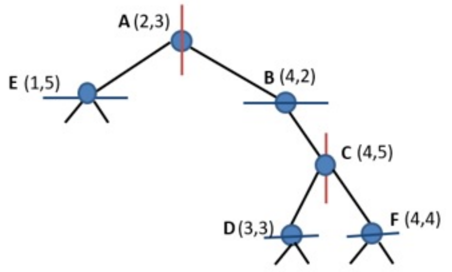
Write down RectHV for all points, with the ordering being min x, min y, max x, max y.

Point	RectHV [(min x, min y),(max x, max y)]
A	$[(-\infty, -\infty), (\infty, \infty)]$.
B	
C	
D	
E	
F	

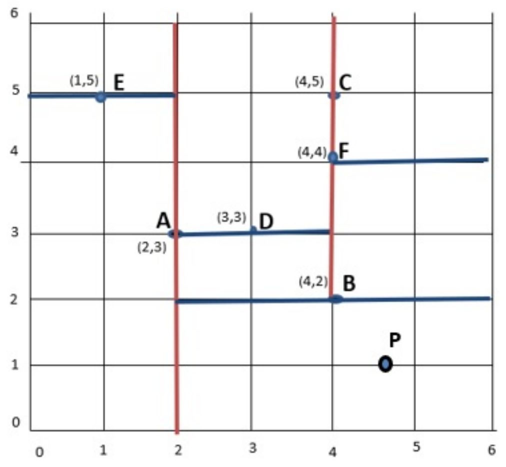
```java
public class RectHV {
  public    RectHV(double xmin, double ymin, double xmax, double ymax)
  public  double xmin()
  public  double ymin()
  public  double xmax()
  public  double ymax()
  public boolean contains(Point2D p)
  public boolean intersects(RectHV that)
  public  double distanceSquaredTo(Point2D p)
  public boolean equals(Object that)
  public  String toString()
}
```

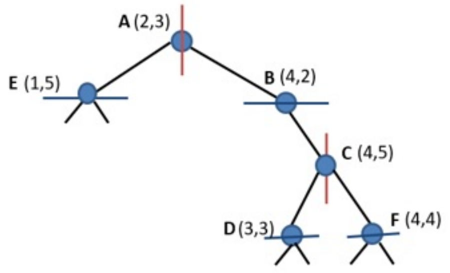
C. Consider a range query on the shaded rectangle. Write R next to each node in your KdTree (drawn in Part A) that is considered during the KdTree traversal by the range search algorithm. Circle the nodes that were found to be inside the shaded rectangle. Do not count null nodes or nodes whose corresponding rectangles do not intersect the query rectangle.





D. Consider a nearest neighbour query on point p. For your convenience, the correct answer from part A is provided below. Number each node (starting with 0) by the order in which it is visited by the nearest neighbor algorithm UNLESS that node's corresponding rectangle rules out that node or its children. For nodes that are pruned based on rectangle distance, write an X over the node. number null nodes.


(1,5) E
(4,5) C
(4,4) F
A
(3,3) D
(2,3)
(4,2) B
P
0 1 2 3 4 5 6
0 1 2 3 4 5 6


A (2,3)
E (1,5)
B (4,2)
C (4,5)
D (3,3)
F (4,4)