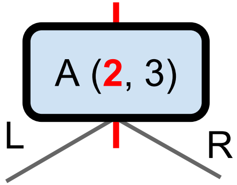
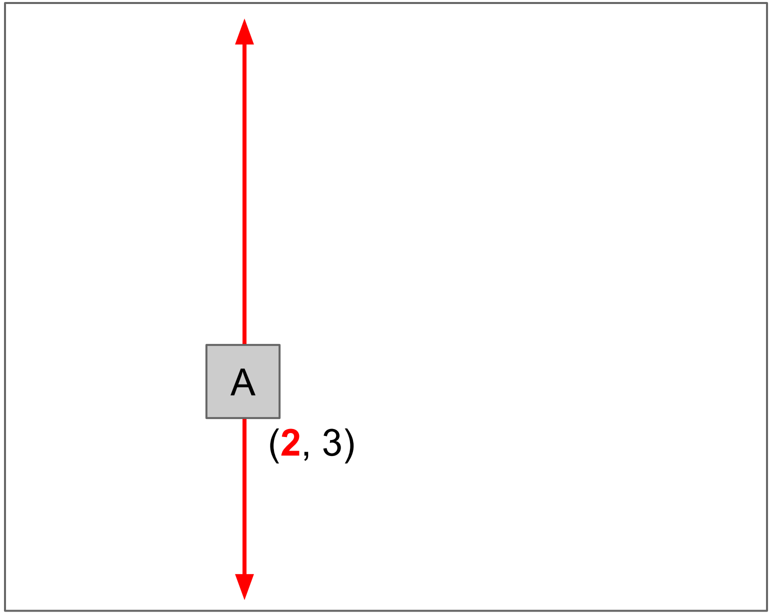
K-d Tree Insertion Demo

Insert A (2, 3) ?

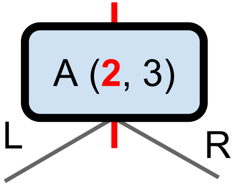
K-d Tree Insertion Demo



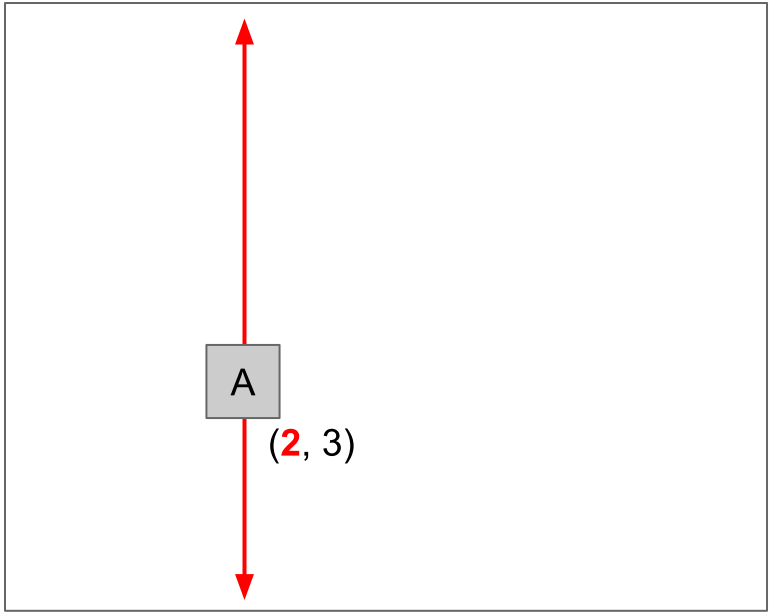
Insert A (2, 3)



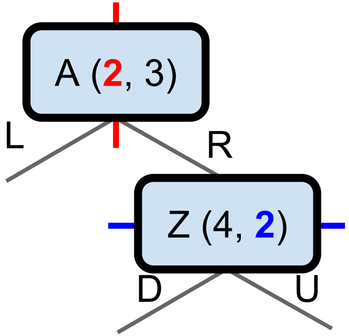
K-d Tree Insertion Demo

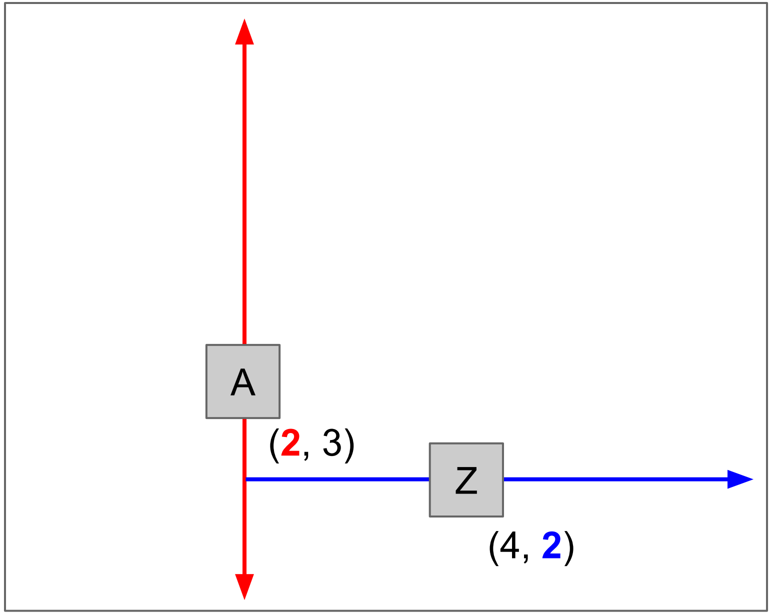


Insert Z (4, 2) ?



K-d Tree Insertion Demo

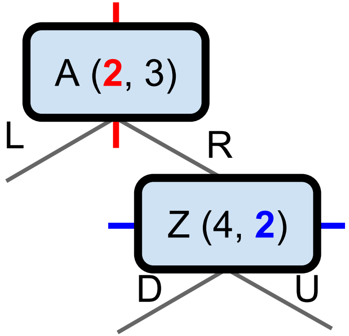




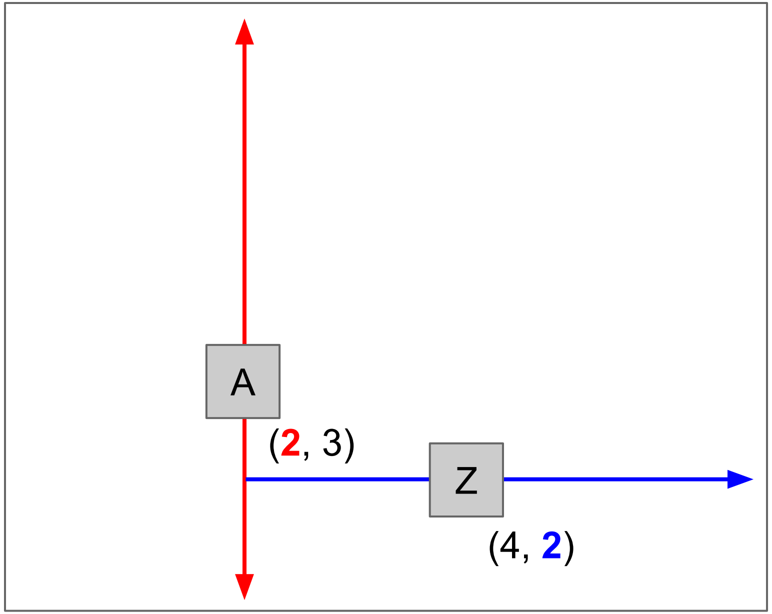
Insert Z (4, 2)

- Z is to the right of A, because 4 > 2.

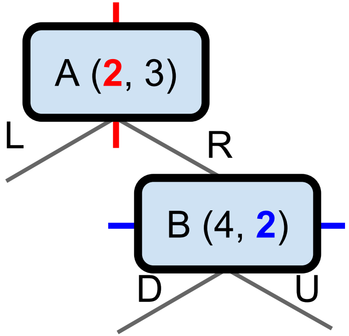
K-d Tree Insertion Demo

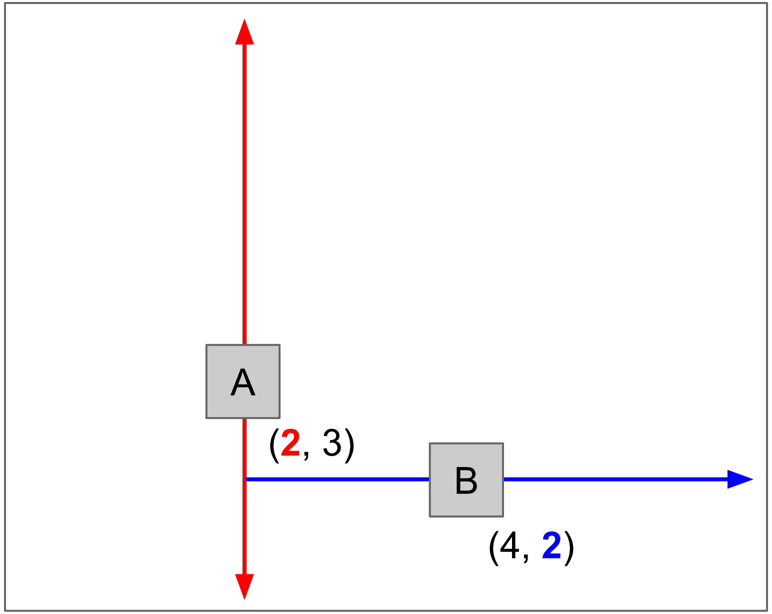


Insert B (4, 2) ?



K-d Tree Insertion Demo

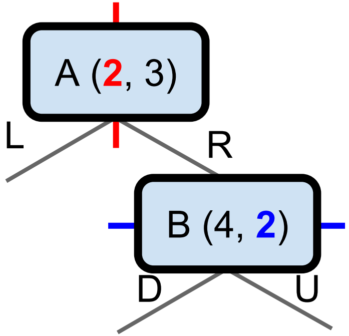




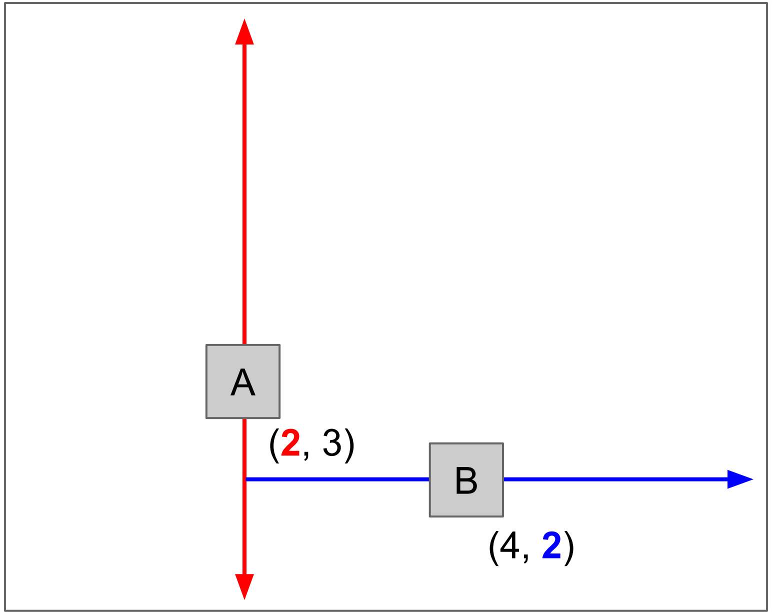
Insert B (4, 2)

- B is to the right of A, because 4 > **2**.
- (4, 2) is already a key in our table.
  - Replace with new value B.

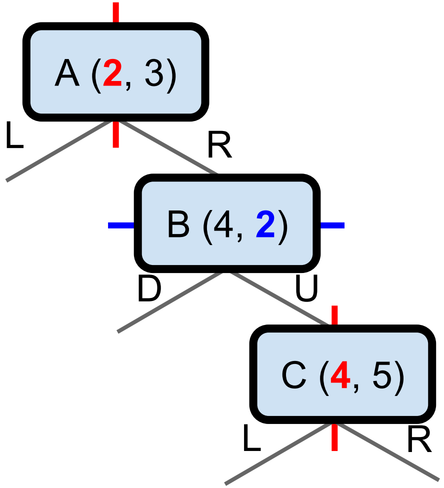
K-d Tree Insertion Demo

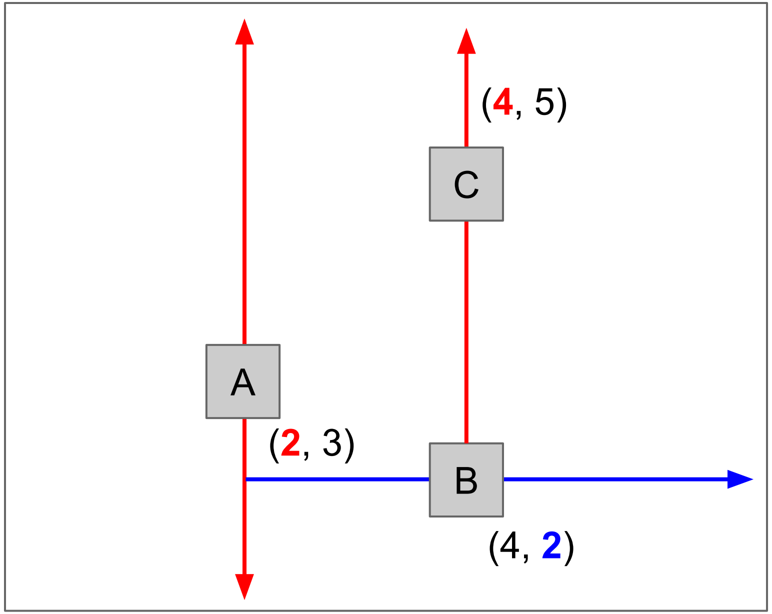


Insert C (4, 5) ?



K-d Tree Insertion Demo

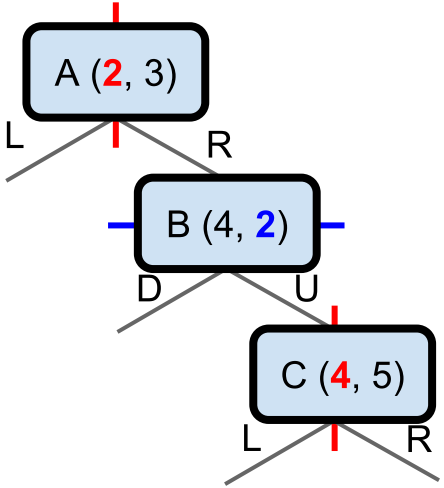




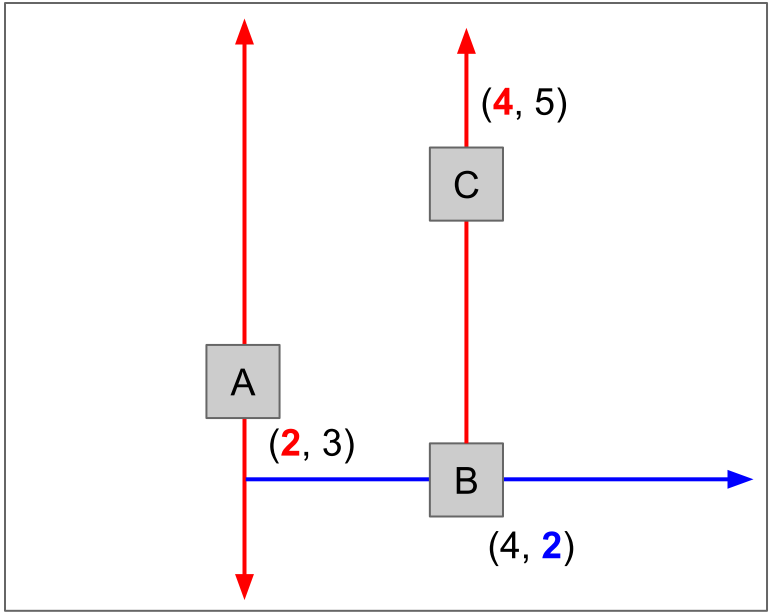
Insert C (4, 5)

- C is to the right of A, because 4 > **2**.
- C is to the up of B, because 5 > **2**.

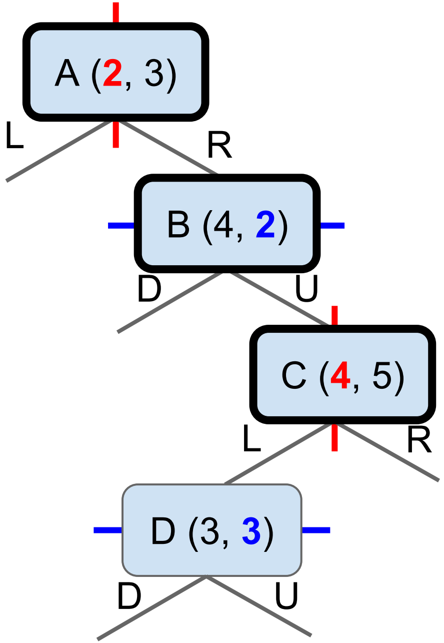
K-d Tree Insertion Demo



Insert D (3, 3) ?

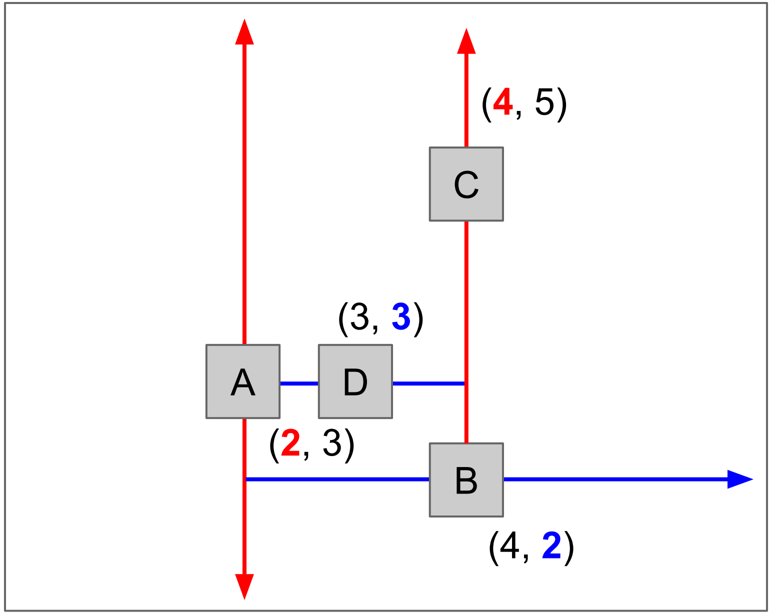


K-d Tree Insertion Demo

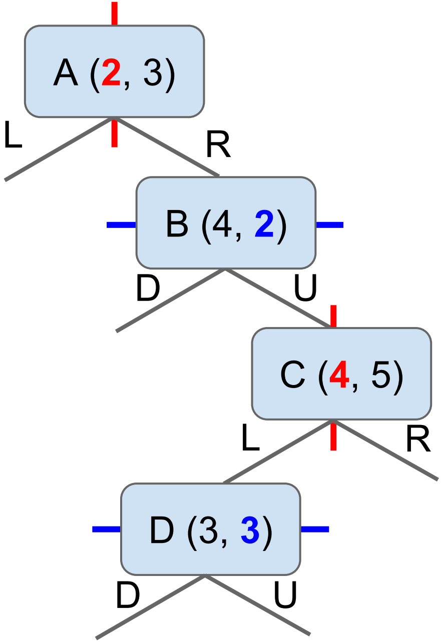


Insert D (3, 3)

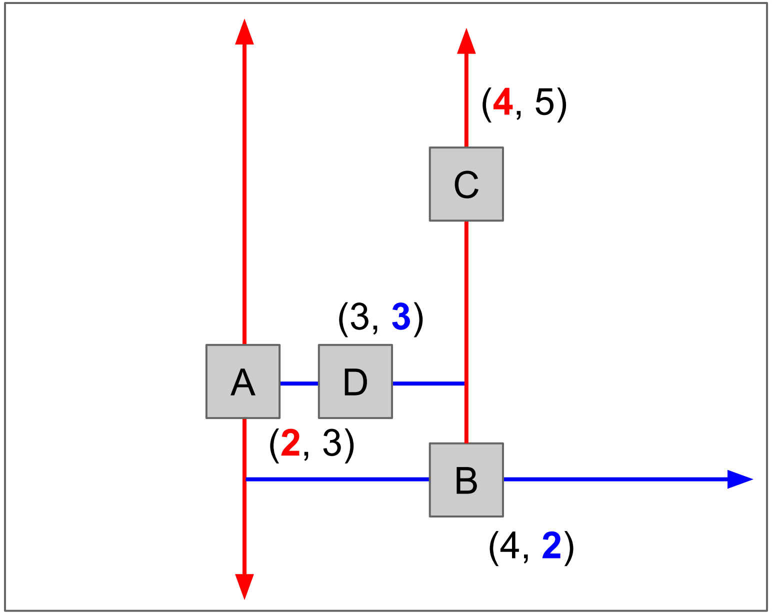
- D is to the right of A, because 3 > **2**.
- D is to the up of B, because 3 > **2**.
- D is to the left of C, because 3 < **4**.



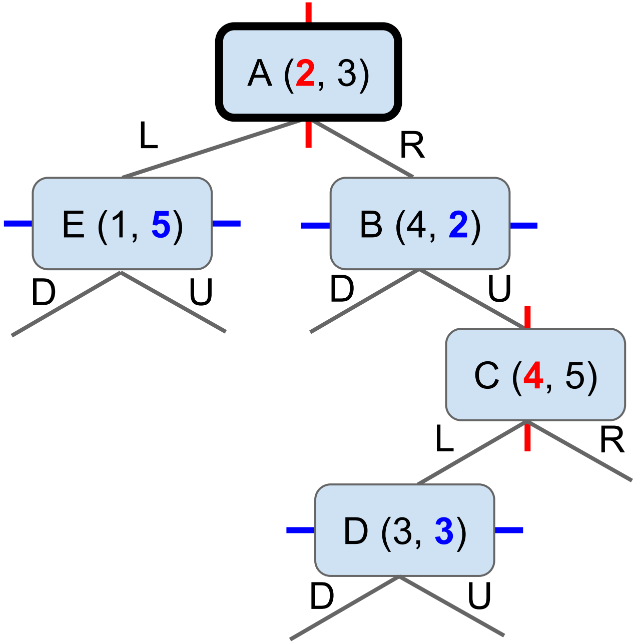
K-d Tree Insertion Demo



Insert E (1, 5) ?

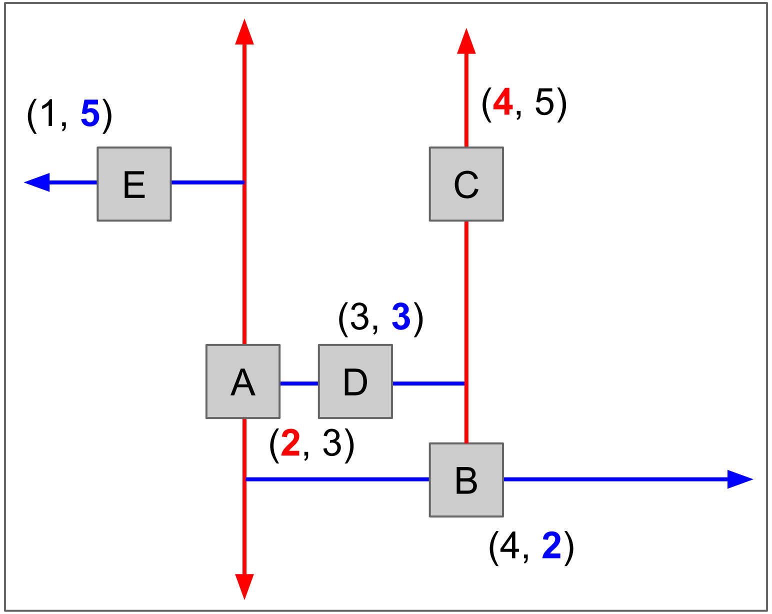


K-d Tree Insertion Demo

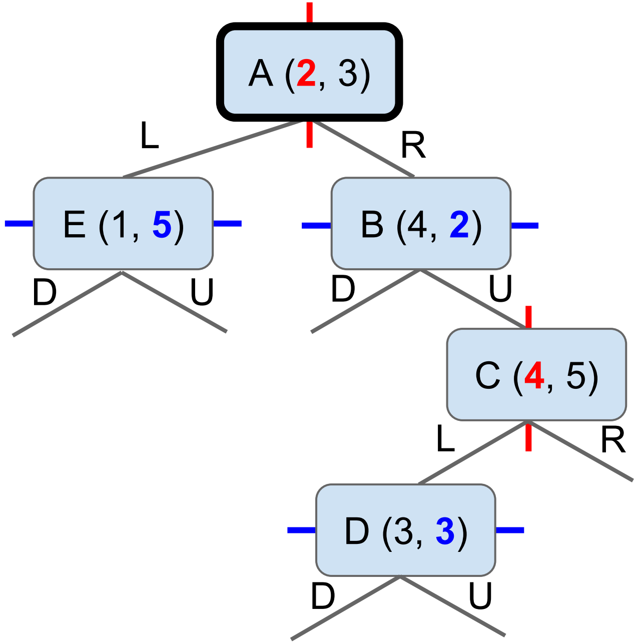


Insert E (1, 5) ?

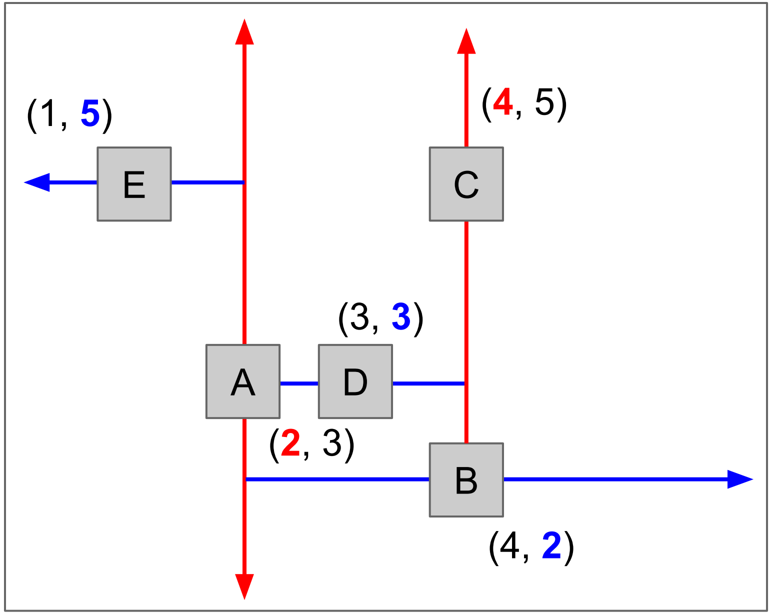
- E is left of A because (1 < 2).



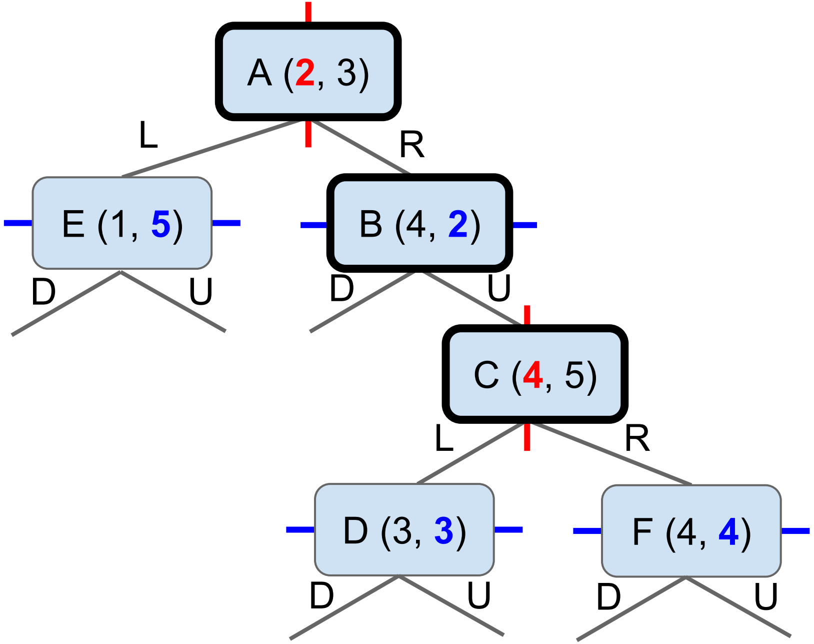
K-d Tree Insertion Demo



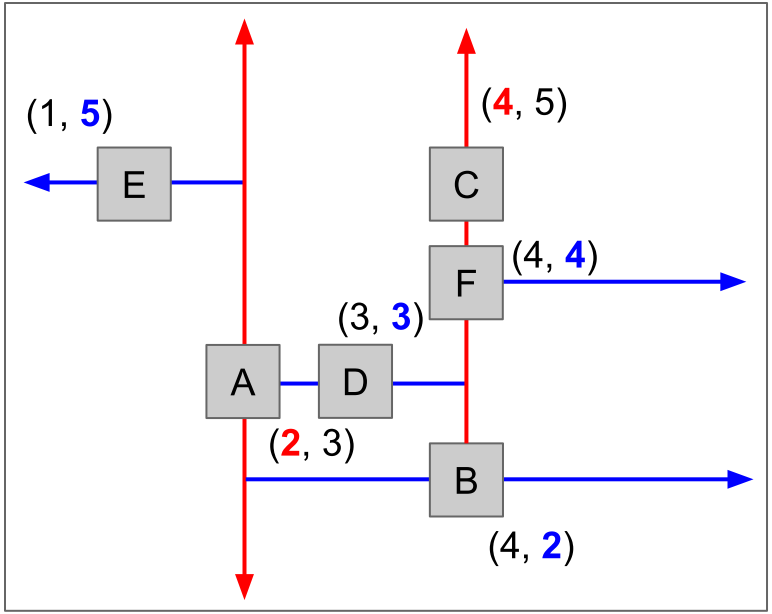
Insert F (4, 4) ?



K-d Tree Insertion Demo



Have to break ties somehow. We'll say items that are equal in one dimension go off to the right (or up) child of each node.



Insert F (4, 4) ?

- F is right of A because 4 > 2.
- F is up of B because 4 > 2.
- F is right of C because 4 >= 4.