Block Project – Part 2

**Description**

Add the following methods to your Block class with **Javadoc comments**. Here’s a link if you need help: <http://www.drjava.org/docs/user/ch10.html>

If done correctly you can generate an HTML file with the comments and methods that you wrote. Your IDE probably has a short cut for this! And most IDE’s will show these comments, so users won’t have to look them up.

// converts the Block to a standard representation with positive width and height

// given a block with x = 10, y = 10, w = 3, h = -4 then executing canonicalize results in

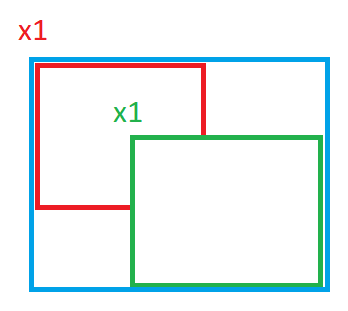
// x = 10, y = 6, w = 3, h = 4

public void canonicalize(){}

// returns a new Block that’s the tightest bounding region that encompasses both blocks

public Block union(Block other){}

Visual Help with Union

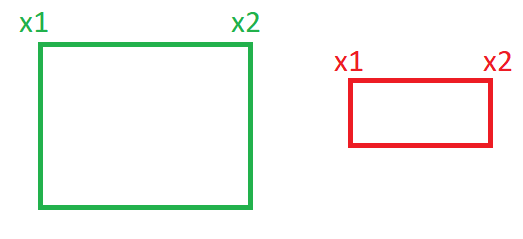


Union forms a block that encompasses both blocks. Which x1 should you use? Follow the same approach for the other 3 sides. Math.max and Math.min will be useful.

// returns a new Block that’s the intersection of the two blocks or null if none exists

public Block intersection(Block other){}

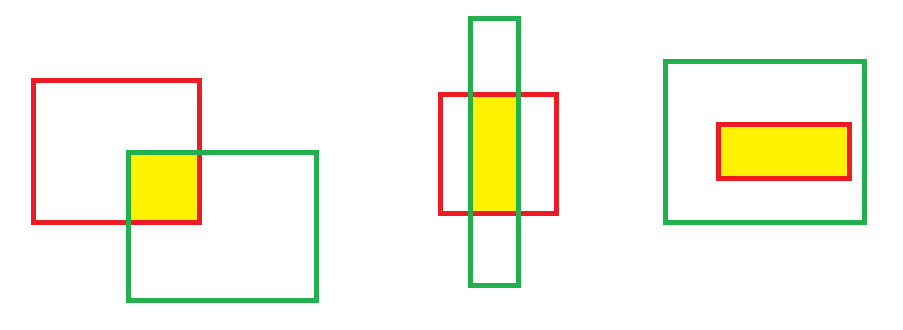
Visual Help with Intersection



Clearly these two blocks don’t intersect. There are 4 general cases where this happens. Check these first and return null if true.

X2 < X1 🡪 therefore return null. Check the other 3 cases.

Now the rectangles are guaranteed to intersect in some manner.



There are lots of ways this could happen, and the coordinates are known for all of them. It follows a very similar pattern as union. Which x1 would you use? Follow the same approach for the other 3 sides. Math.max and Math.min will be useful.