CS 129 HW 3
Rectangles

Write a program with 3 classes: Driver, Rect, and Point. Do **not** use the built-in Java Point or Rectangle classes, though we will use them later on.

Point has two integer fields: x and y, and represents a point in 2D space. You should also make a toString method for Point that prints something like “Point: (xX, yY)”, where X and Y are the values of the x and y coordinate.

In this coordinate system, the top-left corner is (0, 0). Moving right increases x, and moving down **increases** y. That is, up is negative. See the image on the next page.

Rect maintains an array of 4 points. The first point is the top-left corner, then the top-right, bottom-left, and bottom-right, in that order. Rect also maintains the width and height of the rectangle. Rect has two constructors:

- Takes an x, y, width, and height. The x and y coordinates are the top-left corner, and the width and height tell you the rest (see image in next page for help)
- Takes 8 integers, corresponding to the 4 (x, y) pairs you need. Like so: (xTopLeft, yTopLeft, xTopRight, yTopRight, xBottomLeft….So forth)

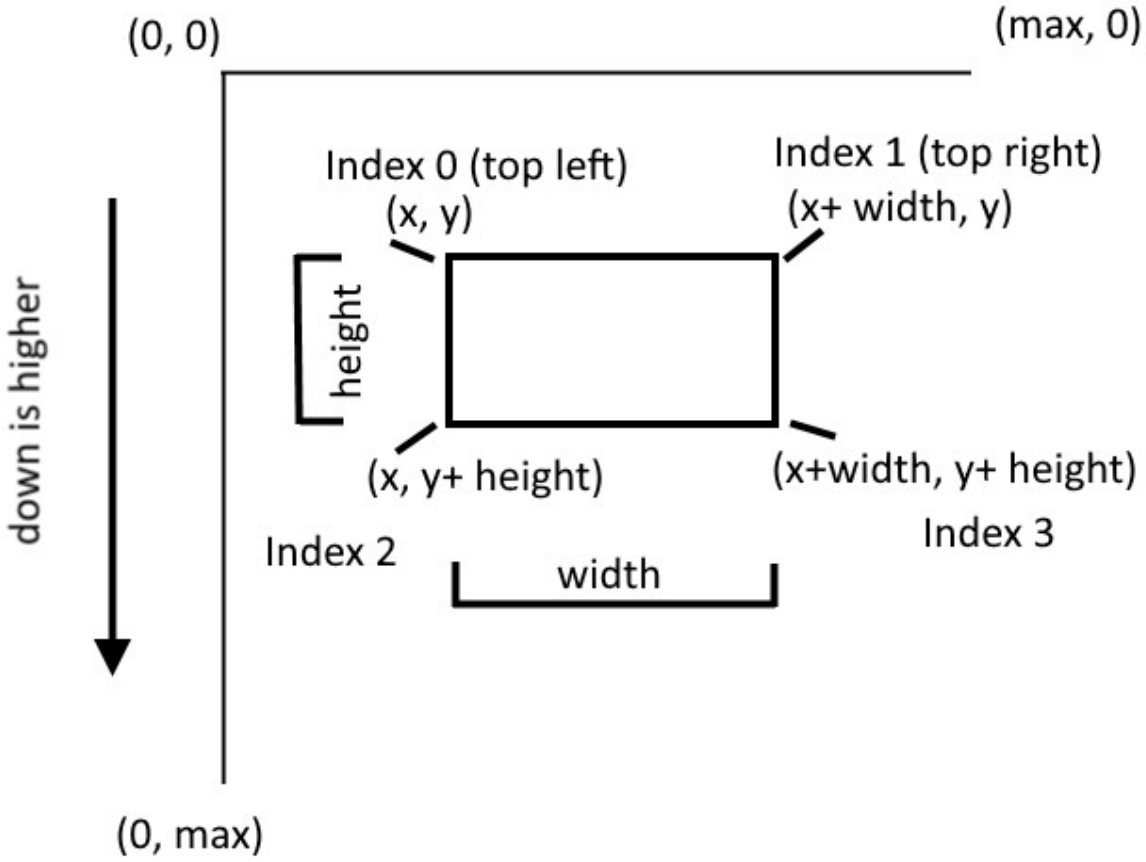
Extra credit (10%), print an error if a Rect is created that is not a rectangle. Squares are fine, but other quads don’t count. A “rotated” square, like a diamond, also doesn’t count as a rectangle.
Extra credit (5%) also accept diamonds, and only reject shapes that actually aren’t rectangles at all. (hint: trig stuff probably required)

Rect also has following methods:

- void translate(int x, int y) moves the whole shape by x, y units For example, to move left 2 units, you would do myRect.translate(-2, 0). Remember that all the points have to move!
- Rect scale(float xScale, float yScale) returns a new Rect that’s width * xScale wide and height * yScale tall, and has the same location for the top-left corner.
- Boolean contains(Point p) returns true if the Rect contains the given point (ties return true)
- Boolean contains(Rect other) returns true if this Rect completely contains Rect “other”. That is, the entire Rect other must be inside this one, it doesn’t count if any points stick out.
- String toString() prints the points that make up this Rect in a readable way.
- Getters for anything that needs a getter

Extra credit (up to 5%) for exceptional documentation, especially good method and class headers (/** comments).

See the next two pages for a visual guide, a sample main method you can use for testing, and the sample output.


(0, 0)
(max, 0)
Index 0 (top left)
(x, y)
Index 1 (top right)
(x+ width, y)
down is higher
height
(x, y+ height)
(x+width, y+ height)
Index 2
Index 3
width
(0, max)

```
public static void main(String[] args) {

		Point origin = new Point(0, 0);
		System.out.println("The origin is at: " + origin.x + " " + origin.y);

		// A 5 by 5 rectangle whose top-left corner is the origin (0, 0)
		Rect alpha = new Rect(0, 0, 5, 5);
		// A 2 by 4 rectangle whose top-left corner is the origin (0, 0)
		Rect beta = new Rect(0, 0, 2, 4);
		System.out.println("beta is a..." + beta.toString());
		System.out.println("The top left corner of beta is at: " + beta.getX() + " " + beta.getY());
		System.out.println("beta is " + beta.getWidth() + " units wide and " + beta.getHeight() + " units tall");
		System.out.println();

		Point oneOne = new Point(1, 1);
		if (alpha.contains(oneOne)) {
			System.out.println("alpha contains oneOne: " + oneOne.toString());
		}
		if (alpha.contains(beta)) {
			System.out.println("alpha contains beta: " + beta.toString());
		}
		System.out.println();

		beta.translate(2, 2);
		System.out.println("beta is a..." + beta.toString());
		System.out.println("The top left corner of beta is at: " + beta.getX() + " " + beta.getY());
		System.out.println("beta is " + beta.getWidth() + " units wide and " + beta.getHeight() + " units tall");
		System.out.println();

		if (alpha.contains(beta)) { // prints nothing, beta is no longer inside alpha...
			System.out.println("alpha still contains: " + beta.toString());
		}

		Rect biggerBeta = beta.scale(2, 4);
		System.out.println("bigger beta is a..." + biggerBeta.toString());
		System.out.println("The top left corner of bigger beta is at: " + biggerBeta.getX() + " " + biggerBeta.getY());
		System.out.println("bigger beta is " + biggerBeta.getWidth() + " units wide and " + biggerBeta.getHeight()
				+ " units tall");
		System.out.println();

		Rect delta = new Rect(3, 3, 6, 3, 3, 5, 6, 5);
		if (biggerBeta.contains(delta)) {
			System.out.println("biggerBeta contains delta: " + delta.toString());
		}

		// (Bonus)
		Rect notARect = new Rect(3, 3, 4, 5, 3, 5, 6, 5);
}
```

```
The origin is at: 0 0
beta is a...Rect with: Point: (x0, y0), Point: (x2, y0), Point: (x0, y4), Point: (x2, y4),
The top left corner of beta is at: 0 0
beta is 2 units wide and 4 units tall

alpha contains oneOne: Point: (x1, y1)
alpha contains beta: Rect with: Point: (x0, y0), Point: (x2, y0), Point: (x0, y4), Point: (x2, y4),

beta is a...Rect with: Point: (x2, y2), Point: (x4, y2), Point: (x2, y6), Point: (x4, y6),
The top left corner of beta is at: 2 2
beta is 2 units wide and 4 units tall

bigger beta is a...Rect with: Point: (x2, y2), Point: (x6, y2), Point: (x2, y18), Point: (x6, y18),
The top left corner of bigger beta is at: 2 2
bigger beta is 4 units wide and 16 units tall

biggerBeta contains delta: Rect with: Point: (x3, y3), Point: (x6, y3), Point: (x3, y5), Point: (x6, y5),
Point: (x3, y3)
Point: (x4, y5)
Point: (x3, y5)
Point: (x6, y5)
Warning, not a rectangle!
```