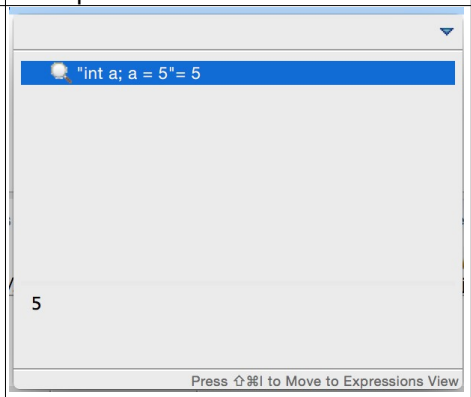
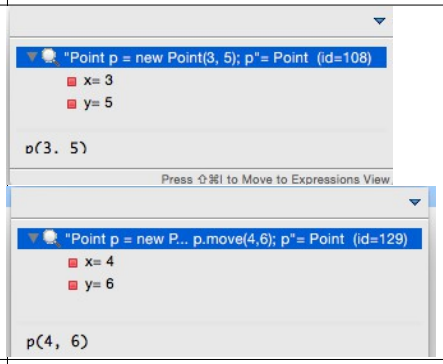
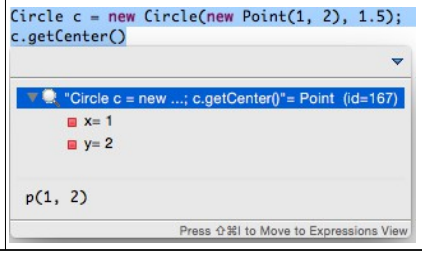
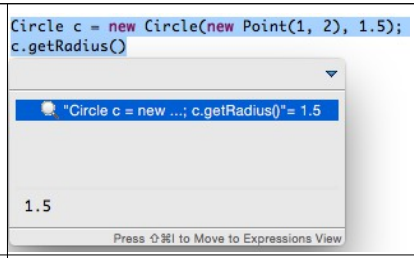
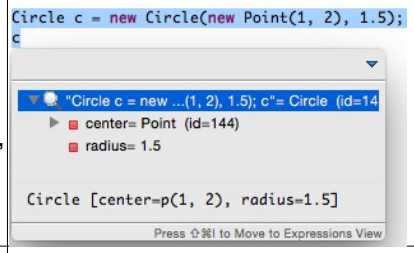
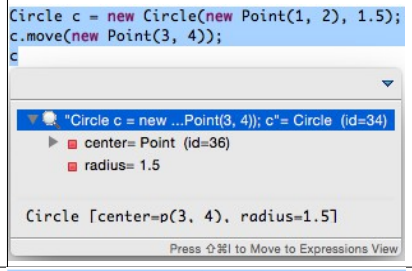
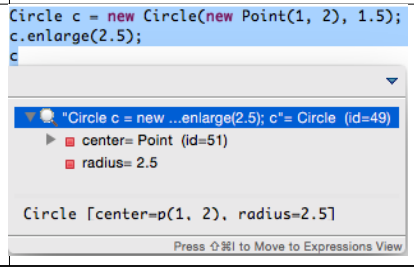
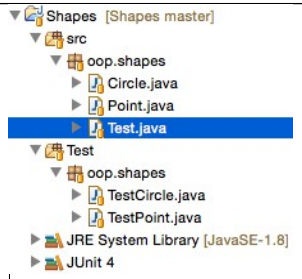
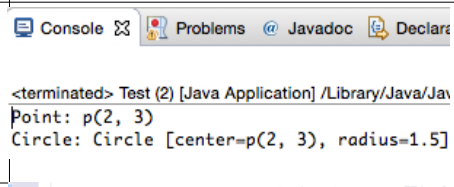
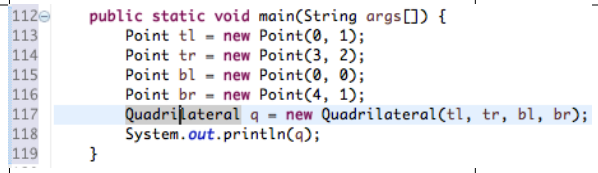
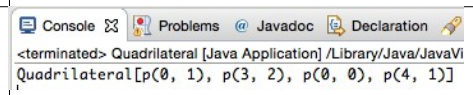
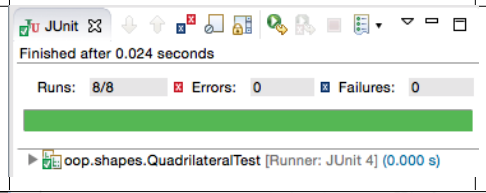
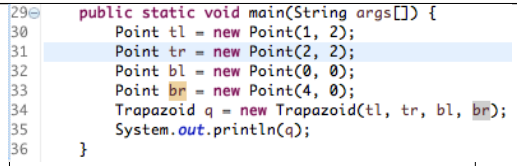
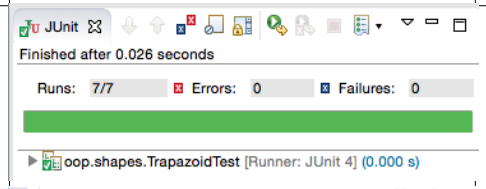
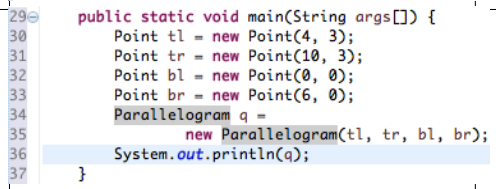
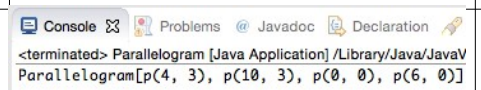
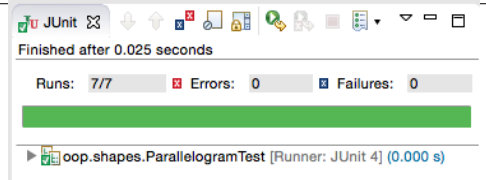
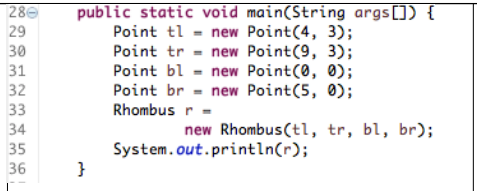
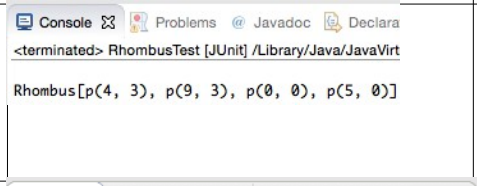
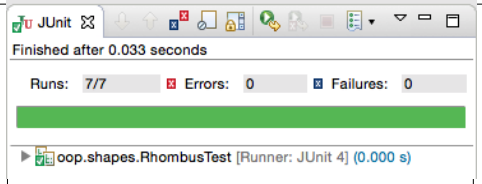
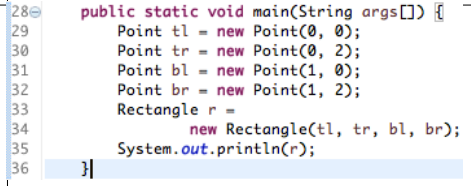
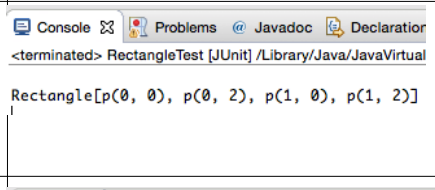
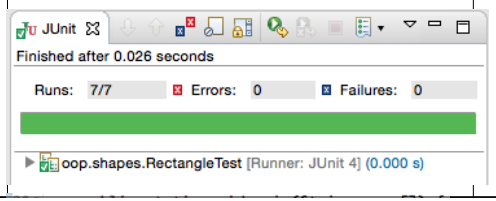
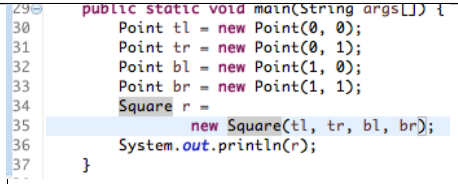
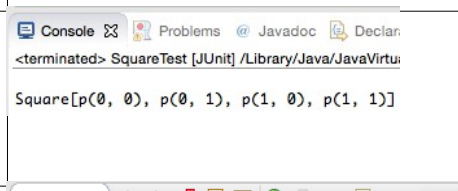
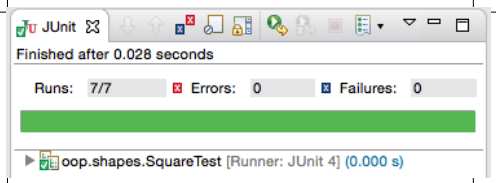
S.No	Prio.	Story	Test	Example	Notes
1	1	Evaluate expressions in a Scrapbook page.	1. Declare an integer. 2. Set variable. 3. Declare an integer array 4. Write a loop to initialize array 5. Write a loop to calculate average 6. Write a loop to revers an array. 7. Write a lop to calculate first 10 Fibonocci numbers 8. Write a loop that will put the first 50 integers divisible by the first five integers in five arrays.		Lab 0 gives instructions on setting up a Scrapbook page.
2	2	Create a Point class as described in Lecture. Member variables x and y. Methods: Point(int int), move(), getX(), getY(). ToString produces p(n, m) where n is the value of x and m is the value of y.	1. Point p = Point(3, 5); 2. p 3. See p(3, 5) 4. p.move(4, 6); 5. p 6. See p(4, 6) 7.System.out.println(p.getX() + ” “ + p.getY()); 8 see 4 6		
3	3	Create a Circle class that uses your Point class as a center point and include a floating point radius. Add a getCenter() method.	1. Point p = new Point(1, 2); 2. Circle c = new Circle(p, 1.5); 3. c.getCenter() 4. See p(1, 2) 5. c.getRadius() 6. See 1.5		

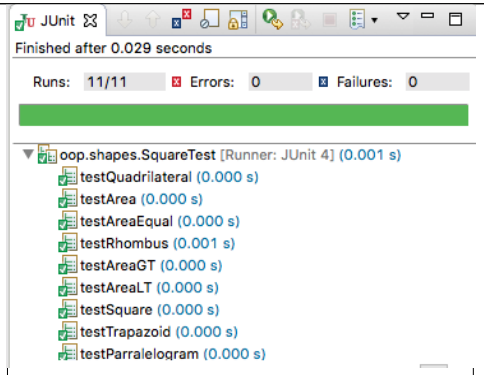
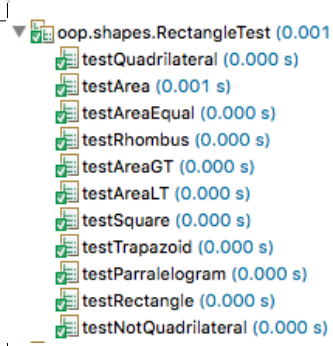
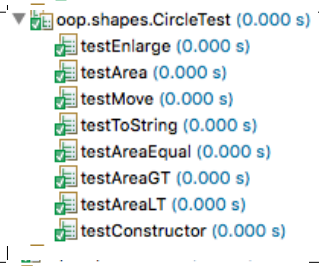
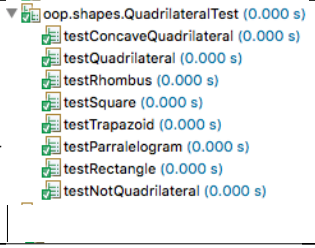
4	4	Add a getRadius method to your Circle Class	1. Point p = new Point(1, 2); 2. Circle c = new Circle(p, 1.5); 3. c.getCenter() 4. See p(1, 2) 5. c.getRadius() 6. See 1.5		
5	5	Add a toString method to your Circle class that prints a c followed by parentheses followed by the center point, followed by a comma and the radius, followed by a close parenthesis.	1. Point p = new Point(1, 2); 2. Circle c = new Circle(p, 1.5); 3. c.toString() 4. See Circle [center=p(1,2), Radius=1.5]		
6	6	Add a move method to your Circle class.	1. Circle c = Circle(new Point(1, 2), 1.5); 2. c.move(new Point(3, 4)); 3. c 4 See Circle [center=p(3,4), Radius=1.5]		
7	7	Add an enlarge method to your Circle class	1. Circle c = Circle(new Point(1, 2), 1.5); 2. c.enlarge(2.5); 3. c 4 See Circle [center=p(3,4), Radius=2.5]		

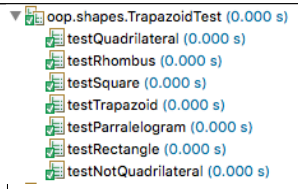
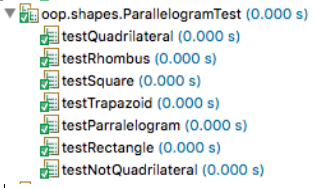
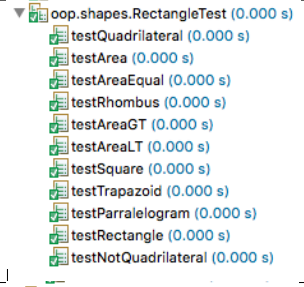
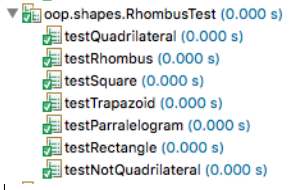
8	8	Save your work in a Git repository on the class AWS server.	1. Create a local Git Repository 2. Add your project to the local Git repository. 3. Create a remote git repository 4. Store your materials in the remote git repository. 5. See the stored files		
9	9	Create a new class called Test that contains only a main() method that will make a Point and Circle object, then print those objects on the consol.	1. Select Test 2. Choose Run as from the context menu (the one you get when you right click) 3. Choose Java Class		
10	10	Create a class, Quadrilateral, that stores four points.	1. Create a main() method in Quadrilateral that creates a Quadrilateral. 2. Pass the constructor four points. 3. See the method compile and run.		
11	11	Overload toString() to print the four points of the Quadrilateral.	1. in the main() method call the toString() method. 2. See `Quadrilateral[p(0, 1), p(3, 2), p(0, 0), p(4, 1)]`		
12	12	Create a test called isRightShape() that returns true when the shape is a convex quadrilateral, false otherwise.	The JUnit test, QuadrilateralTest will run correctly.		

13	13	Create a class, Trapezoid, that is a subclass of Quadrilateral	1. Create a main() method in Trapezoid that creates a Trapezoid. 2. Pass the constructor four points. 3. See the method compile and run.	
14	14	Overload toString() to print the four points of the Trapezoid.	1. in the main() method call the toString() method. 2. See `Trappazoid[p(1, 2), p(2, 2), p(0, 0), p(4, 0)]`	
15	15	Create a test called isRightShape() that returns true when a shape is a trapezoid (trapezium), false otherwise.	The JUnit test, TrapazoidTest test will run correctly.	
16	16	Create a class, Parallelogram that is a subclass of Trapezoid.	1. Create a main() method in Parallelogram that creates a Parallelogram. 2. Pass the constructor four points. 3. See the method compile and run.	
17	17	Overload toString() to print the four points of the Parallelogram.	1. in the main() method call the toString() method. 2. See `Parallelogram[p(4, 3), p(10, 3), p(0, 0), p(6, 0)]`	
18	18	Create a test called isRightShape() that returns true when a shape is a parallelogram, false otherwise.	The JUnit test, ParallelogramTest test will run correctly.	

19	19	Create a class, Rhombus that is a subclass of a Trapezoid.	1. Create a main() method in Rhombus that creates a Rhombus. 2. Pass the constructor four points. 3. See the method compile and run.		
20	20	Overload toString() to print the four points of the Rhombus.	1. in the main() method call the toString() method. 2. See `Rhombus[p(4, 3), p(9, 3), p(0, 0), p(5, 0)]`		
21	21	Create a test called isRightShape() that returns true when a shape is a rhombus, false otherwise.	The JUnit test, RhombusTest test will run correctly.		
22	22	Create a class, Rectangle that is a subclass of a Parallelogram.	1. Create a main() method in Rectangle that creates a Rectangle. 2. Pass the constructor four points. 3. See the method compile and run.		
23	23	Overload toString() to print the four points of the Rectangle.	1. in the main() method call the toString() method. 2. See `Rectangle[p(0, 0), p(0, 2), p(1, 0), p(1, 2)]`		
24	24	Create a test called isRightShape() that returns true when a shape is a rectangle, false otherwise.	The JUnit test, RectangleTest test will run correctly.		

25	25	Create a class, Square that is a subclass of Rhombus.	1. Create a main() method in Square that creates a Square. 2. Pass the constructor four points. 3. See the method compile and run.		
26	26	Overload toString() to print the four points.	1. in the main() method call the toString() method. 2. See `Square[p(0, 0), p(0, 1), p(1, 0), p(1, 1)]`		
27	27	Create a test called isRightShape() that returns true when a shape is a Square, false otherwise.	The JUnit test, SquareTest test will run correctly.		
28	28	Create an interface called AreaComparable that provides three boolean methods areaLT, areaGT and areaEqual that take a Quadrilateral as a parameter. In addition, the interface provides a method called area, that takes no parameters. All methods that print the name of the method has not been implemented for the type of the object. For example, “area() has not been implemented for quadrilateral.” or “areaEqual() has not been implemented for square with a circle parameter.	The JUnit test, AreaComparableTEst will run correctly and produce output on the console.	oop.shapes.AreaComparableTest.area() is not defined. oop.shapes.AreaComparableTest.areaEqual(oop.shapes.AreaComparableTest) is not defined. oop.shapes.AreaComparableTest.areaGT(oop.shapes.AreaComparableTest) is not defined. oop.shapes.AreaComparableTest.areaLT(oop.shapes.AreaComparableTest) is not defined.	

29	29	Add the interface defined in story 28 to square.	The Junit test for Square includes tests for the interface and works correctly.		
30	30	Add the interface defined in story 28 to rectagle.	The Junit test for Rectangle includes tests for the interface and works correctly.		
31	31	Add the interface defined in 28 to circle.	The JUnit test for Circle includes tests for the interface and works correctly.		
32	32	As a programmer, I want the Quadrilateral constructor to throw an exception when the points do not form a quadrilateral instead of returning a null shape, so I can tell when an error occurs.	The constructor for Quadrilateral throws a ShapeException with the message “Quadrilateral Error: “, the points, “ is not a quadrilateral.” when the points do not form a convex quadrilateral.		

33	33	As a programmer, I want the Trapezoid constructor to throw an exception when the points do not form a trapezoid instead of returning a null shape, so I can tell when an error occurs.	The constructor for Trapezoid throws a ShapeException with the message “Trapezoid Error: “, the points, “ is not a trapezoid.” when the points do not form a trapezoid.		
34	34	As a programmer, I want the Parallelogram constructor to throw an exception when the points do not form a parallelogram instead of returning a null shape, so I can tell when an error occurs.	The constructor for Parallelogram throws a ShapeException with the message “Parallelogram Error: “, the points, “ is not a parallelogram.” when the points do not form a parallelogram.		
35	35	As a programmer, I want the Rectangle constructor to throw an exception when the points do not form a rectangle instead of returning a null shape, so I can tell when an error occurs.	The constructor for Rectangle throws a ShapeException with the message “Rectangle Error: “, the points, “ is not a rectangle.” when the points do not form a rectangle.		
36	36	As a programmer, I want the Rhombus constructor to throw an exception when the points do not form a rhombus instead of returning a null shape, so I can tell when an error occurs.	The constructor for Rhombus throws a ShapeException with the message “Rhombus Error: “, the points, “ is not a rhombus.” when the points do not form a rhombus.		

37	37	As a programmer, I want the Square constructor to throw an exception when the points do not form a square instead of returning a null shape, so I can tell when an error occurs.	The constructor for Square throws a ShapeException with the message “Square Error: “, the points, “ is not a square.” when the points do not form a rhombus.