Classes Project
Complete 2 of the following 3 projects

Place your files in a folder named LastnameClassesProject and submit the folder with your files as a zip file.

Include a class comment each class file and use the descriptions below for the method comments. For the class comment, include your name, the date, and a brief description of the class. For example:

```
/**
 * A Point object represents an ordered pair of (x, y) coordinates in the
 * 2D Cartesian plane.
 *
 * Jane Doe
 * December 12, 2014
 */
```

1. Write a class called RationalNumber that represents a fraction with an integer numerator and denominator.

   A RationalNumber object should have the following methods:

   Constructs a new rational number to represent the ratio (numerator / denominator). The denominator cannot be 0, so throw an IllegalArgumentException if 0 is passed.
   public RationalNumber(int numerator, int denominator)

   Constructs a new rational number to represent the ratio (0/1).
   public RationalNumber()

   Returns this rational number's denominator value; for example, if the ratio is (3/5), return 5.
   public int getDenominator()

   Returns this rational number's numerator value; for example, if the ratio is (3/5), return 3.
   public int getNumerator()

   Returns a String representation of this rational number, such as "3/5". Omit denominators of 1, returning "4" instead of "4/1".
   public String toString()

   Adds two rational numbers and returns the result as a new RationalNumber.
   public RationalNumber add(RationalNumber other)

   Subtracts two rational numbers and returns the result as a new RationalNumber.
   public RationalNumber subtract(RationalNumber other)

   Use RationalNumberMain to test your class.

   The output from RationalNumberMain should be (if you don't reduce the numbers):

Number 0 is 0
Number 1 denominator is 8
Number 1 numerator is 3
Number 1 is 3/8
Addition Result is 18/-48
Subtraction Result is 20/-32

Optional:
Eliminates any common factors and ensures that the denominator is greater than 0. For example, 3/6 would reduce to 1/2 and 2/-3 would change to -2/3.
private void reduce()

The output from RationalNumberMain should be (if you reduce the numbers):

Number 0 is 0
Number 1 denominator is 8
Number 1 numerator is 3
Number 1 is 3/8
Addition Result is -3/8
Subtraction Result is -5/8

2. Define a class named RandomWalker. A RandomWalker object should keep track of its (x, y) location. Random walkers can start at (0, 0) or can start at the coordinates (x, y). When a walker is asked to move, it randomly moves either left, right, up or down. Each of these four moves should occur with equal probability. The resulting behavior is known as a "random walk."

   Each RandomWalker object should have the following public methods. You may add whatever fields or methods you feel are necessary to implement these methods:

   Instructs this random walker to randomly make one of the 4 possible moves (up, down, left, or right).
   move()

   Returns this random walker's current x-coordinate.
   getX()

   Returns this random walker's current y-coordinate.
   getY()

   Returns the number of steps this random walker has taken.
   getSteps()

   Random walks have interesting mathematical properties. For example, given infinitely many steps, a random walker approaches 100% chance of reaching a particular (x, y) coordinate. To learn more about random walks, visit http://en.wikipedia.org/wiki/Random_walk and http://mathworld.wolfram.com/RandomWalk.html .

   Test your RandomWalker by running it with the RandomWalkerMain test class. The RandomWalkerMain will run your random walker in a loop and animate its position as it moves.

Be sure to include DrawingPanel.java in the same folder.

3. Write a class named GroceryList that represents a list of items to buy from the market, and another class named GroceryItemOrder that represents a request to purchase a particular item in a given quantity (example: four orders of cookies). The GroceryList class should use an array field to store the grocery items and to keep track of its size (number of items in the list so far). Assume that a grocery list will have no more than 10 items. A GroceryList object should have the following methods:

   Constructs a new empty grocery list.
   public GroceryList()

   Adds the given item to this list if the list has fewer than 10 items.
   public void add(GroceryItemOrder item)

   Returns the total sum cost of all grocery item orders in this list.
   public double getTotalCost()

   The GroceryItemOrder class should store an item quantity and a price per unit. A GroceryItemOrder object should have the following methods:

   Constructs an item order to purchase the item with the given name, in the given quantity, which costs the given price per unit.
   public GroceryItemOrder(String name, int quantity, double pricePerUnit)

   Return the total cost of this item in its given quantity. For example, four orders of cookies that cost 2.30 per unit have a total cost of 9.20.
   public double getCost()

   Sets this grocery item's quantity to be the given value.
   public void setQuantity(int quantity)