Homework 7

CSS 142 – 2018

Instructions: Please submit your files to the Canvas submission page for this assignment by the due date. Late homework will not be graded.

Grading Rubric:
70% for correctness - includes: following the requirements of each problem as stated (it is your responsibility to ask for clarifications); using the concepts learned from class; testing your code for correctness; having code that works.
30% for style - includes: commenting (block and in-line); spacing between operator and operands; obeys style guidelines indicated during lectures.

Purpose: In this homework you will write two classes meant to be used to instantiate objects of those types, but where the instantiated objects are themselves unique. This homework will continue your practice of creating classes and objects, as well as provide valuable exercises using arrays. You will also write a test class containing a main method in order to test your objects and their methods.

Point

Part a: Write an immutable class called Point to represent a point on the two dimensional Cartesian plane, with the following variables and methods:

1. *Private data fields*: x of type double and y of type double;
2. *Public constructor*: only one constructor that takes a double x_coord and a double y_coord as arguments;
3. *Public methods*:
    - boolean equals(Point anotherPoint) : returns true if the invoking Point is equal to the argument Point. Two points are equal if they have the same x coordinate and the same y coordinate;
    - boolean lessThan(Point anotherPoint) : returns true if a Point is less than another Point by the following rule: first compare points by their $y$coordinates, breaking ties by their x-coordinates.
      Formally, the invoking Point (x_0,y_0) is less than the argument Point (x_1,y_1) if and only if either $y_0 < y_1$ or if $y_0 = y_1$ and $x_0 < x_1$;
    - double slopeTo(Point anotherPoint) : returns the slope between the invoking Point and the argument Point. The slope between two points is given by the formula: $(y_1 - y_0)/(x_1 - x_0)$. The slope of a vertical line segment should be

positive infinity, while the slope between a point and itself should be negative infinity.

- int compareSlopes(Point anotherPoint) : returns −1 if the invoking Point has slope less than the argument Point from the origin, 0 if both points have equal slopes from the origin, and +1 if the invoking Point has slope greater than the argument Point from the origin. *Note*: the origin is the point (0,0);
- String toString() : returns the string representation of a Point objects as follows: (x, y).

PointArray

Part b: Write a class called PointArray to represent an array of Point objects containing the following variables and methods:

1. *Private data field*: points an array of type Point;
2. *Public constructors*:
   - a constructor that takes an array of doubles containing the *x* and *y* values of each point consecutively. That is, if we have the following array of doubles:

     {2.4, 3.5, 4.7, 1.0, 6.4, 11.01} then we can construct the array of three points:

     {new Point(2.4, 3.5), new Point(4.7, 1.0), new Point(6.4, 11.01)} Note: this constructor should only accept arrays of even length.
   - a constructor that takes an argument of type FileInputStream, and constructs the points array by reading a text file containing the points. For example, if the file has the following content:

     3
     2 4
     3.5 -1 0
     17

     then the first number is the number of points, and below it, each line is a point with *x* being the first number and *y* being the second. So we can read the first number in the file to construct a points array of that size, then loop through the file to construct each new Point and store it into the points array.
3. *Public methods*:
   - boolean contains3collinear() : returns true if a PointArray contains three points which are collinear. For example, the points P_0 : (x_0,y_0), P_1 : (x_1,y_1), and P_2 :

(x_2,y_2) are collinear if the slope between P_0 and P_1 is the same as the slope between P_1 and P_2;

- void sort() : sorts the invoking PointArray object using selection sort: the sort is by the ascending order specified by the lessThan method from the Point class above;
- boolean equals(PointArray anotherPointArray) : returns true if the invoking PointArray is equal to the argument PointArray. Two arrays of points are equal if all the points are the same. *Note*: order does not matter for equality;
- String toString() : returns the string representation of a PointArray. For example, a PointArray with 3 points P_0 : (x_0,y_0), P_1 : (x_1,y_1), and P_2 : (x_2,y_2) would be printed as follows: {(x0, y0), (x1, y1), (x2,y2)}.

Testing

Part c: In a class called PointArrayTester, write a main method to test each of your constructors and methods from the PointArray class. Make at least three distinct PointArray objects and one non-distinct (to test the equals method). To test the constructor that takes a FileInputStream argument, download some of my test files from the Canvas assignment page. *Note*: if you wish you may pass in the test files to main as command line arguments, however this is not required.

Important: Please follow the API provided above. That is, use the method names and signatures as provided. I will be testing your code automatically with my own test files and classes.