CSI323 Algorithms   Assignment 1   Monday 24 September 2018

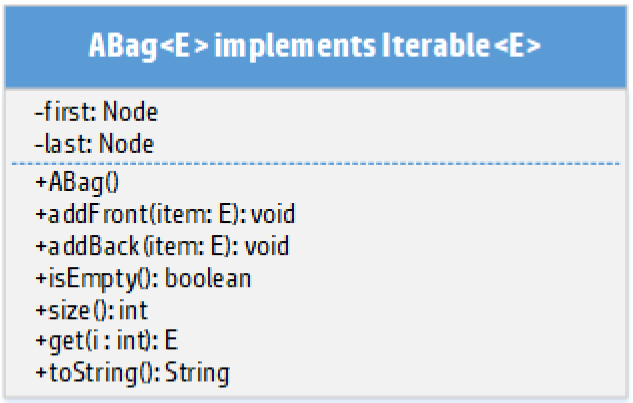
Due: Sunday 04 October 2018: 11:59pm

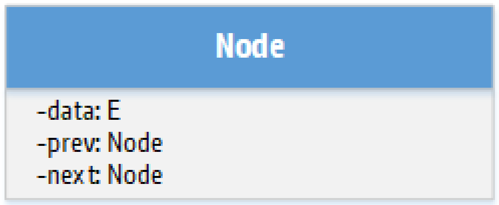
Instructions

- You may work in groups of maximum size 3.
- Only 1 of the group members need to submit. Ensure that work submitted DOES NOT violate UB and course policies. All group members must ensure that submitted work is indeed the group's work.

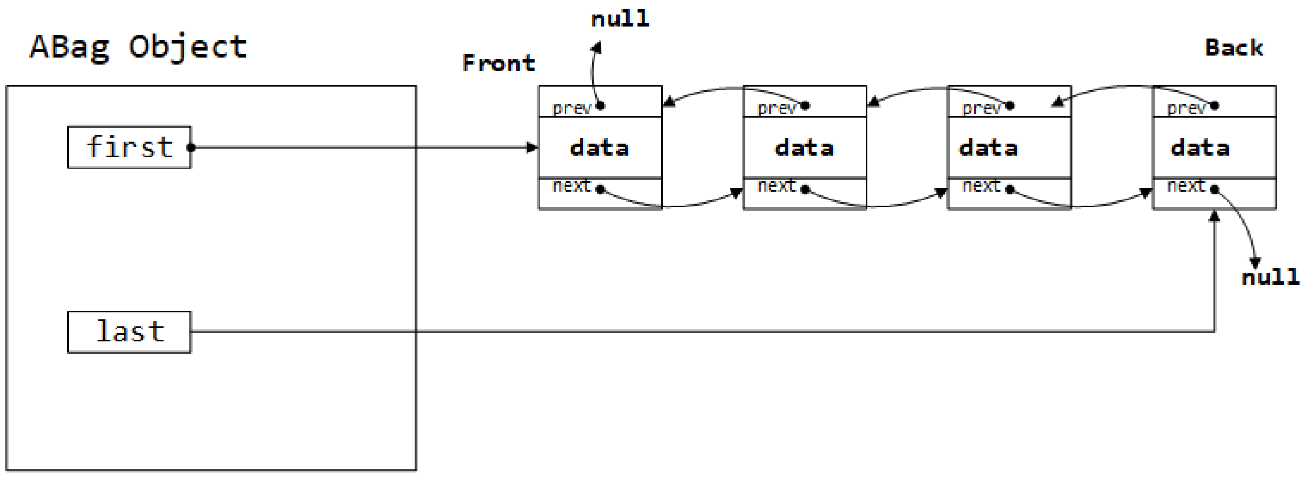
Question 1

In this question, you are to implement a variation of the Bag class, called `ABag`, with what is called a **doubly-linked list**. This is similar to the singly-linked implementation except that a node contains an additional reference/address to the node behind it, if there is any. The UML diagrams of the `ABag` and Node classes are shown below. The Node is a private inner class of class `ABag`,





The diagram below shows `ABag` object with four items (in four nodes) and how the node pointers (**`prev`** and **`next`**) are to be used.

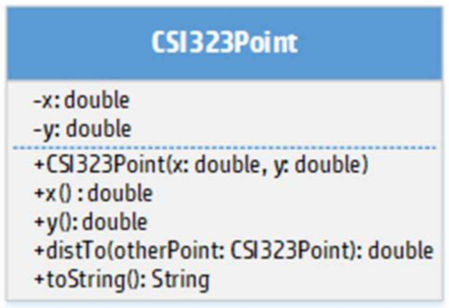


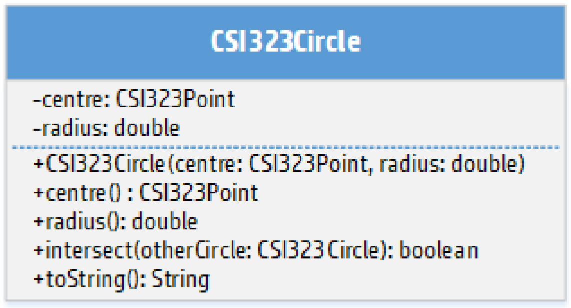
Note that we DO NOT have any other instance variables, so DO NOT add any under any circumstances. The ABag class also implements the Java interface `Iterable`. The implemented iterator **should** iterate through objects from the back to the front.

The `toString` method should use the format similar to the one from the `CSI323LLBag` class. The `get()` method returns (without removing!) the item at specified index. The item at the front has index 0.

Question 2

In this question we investigate a graph made from a set of points in the plane (2-dimensions). Your first task is to implement the following classes.



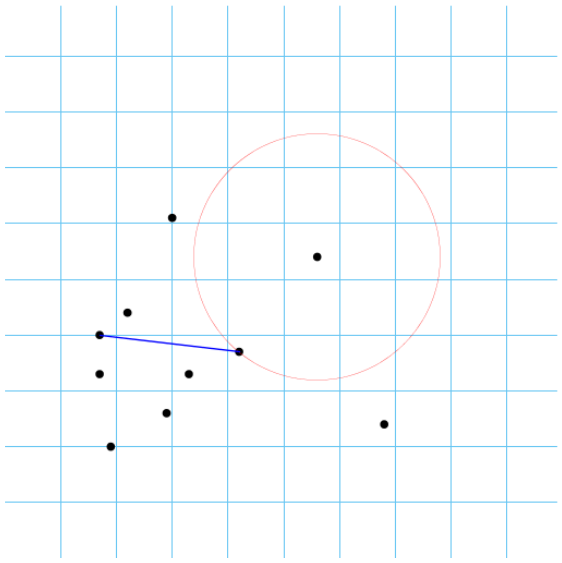


Method	**Operation**
distTo(otherPoint)	Returns the eucledian distance between this point and the given point, otherPoint.
toString() (in CSI323Point)	Returns a string representing the point. If the point is x=2.0 and y=12.0, the string "(2.0, 12.0)" is returned.
Intersect(otherCircle)	Returns true if this circle intersect with the given circle, otherCircle. Note that it does not matter whether it's at one point (just touching) or two points. We DO NOT care about the intersection points, only whether they intersect at all.
toString() (in CSI323Circle)	Returns a string representing the circle, in the form "[(2.0, 12.0), 3.45]". (2.0, 12.0) is the centre and 3.45 is the radius.

After this classes are implemented, you should implement a Java application called **`CSig.java`** with the following methods.

`public static int closestPoint(ABag<CSI323Point> points, int  pos)`	*Returns the index of the closest (in terms of distance) point to the point at index pos.*
`public static ABag<CSI323Circle> circles(ABag<CSI323Point> points)`	*Returns a ABag object of circles. For each point p in the given points, find its closest point q amongst other points, then create a circle with centre p and radius equal to the distance between p and q, its closest point.*
`public static void drawGrid(int N, int step)`	*Draws a grid of size NxN and lines at intervals defined by step. Sample program shows how to do this.*
`public static void plotPoints(ABag<CSI323Point> points)`	*Plots the points.*
`public static void drawCircles(ABag<CSI323Circle> circles)`	*Draws the circles.*
`public static int drawLines(ABag<CSI323Circle> circles)`	*Draw a line between centres of every two pairs of circles that intersect and returns the number of lines drawn*

The main method of class CSig should get an input file as a command line argument (not interactively typed). Each line of the input file contains two doubles, separated by space. Each line represents an input point. The points should be stored in a ABag object. Using methods of the class, your application should draw a grid (we will use a grid size of 1000x1000 and draw lines at step intervals of 20); plot the points, draw circles computed by the circles method, and lastly draw the lines between every pair of intersecting circles. The sample Java program produced the following diagram (note that points are random – every execution gives you a different result!)



Question 3

Implement a Java application, call it Experiment.java, to generate 50 points. Each coordinate of a point should be a random integer between 150 and 850. The points should be stored in an ABag object. The objective is to count how many lines will be used to connect those points whose circles intersect. You do not have to draw anything for this question. Actually it will be more efficient that way. Keep track of the point set that gives the largest number of lines drawn. You should repeat this experiment at least 1000 times. The point set that produces the highest number of lines between points should be written to a file called bestPointset.txt.

What to submit

- ABag.java, CSI323Point.java, CSI323Circle.java, CSig.java, Experiment.java, and bestPointset.txt