CS 401 Algorithms
Assignment2 Convex Hull

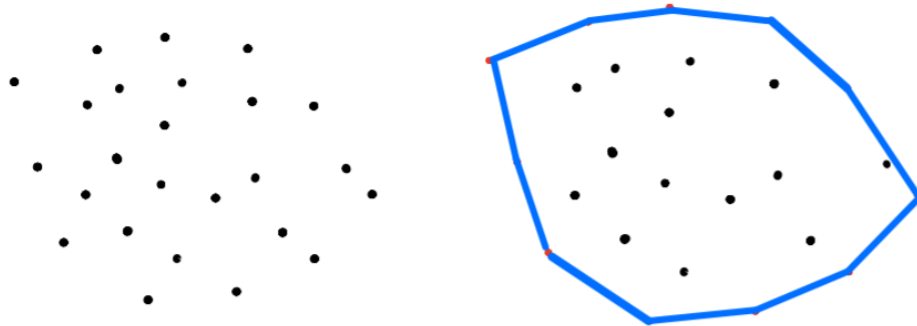**Purpose:** In this assignment, you are asked to develop a Java program which computes the convex hull or convex envelope of N points using Graham's scan algorithm. The convex hull of N points is the smallest convex polygon that contains all the points of it. It finds the minimum area convex region that contains every point.

The Graham's algorithm depends on the two important properties of a convex hull:

- Can traverse the convex hull by making only counterclockwise turns
- The vertices of convex hull appear in **increasing order of polar angle** with respect to point p with lowest y-coordinate.

Sample Applications: Convex hull is widely used in machine learning, robot motion planning, pattern recognition, etc.
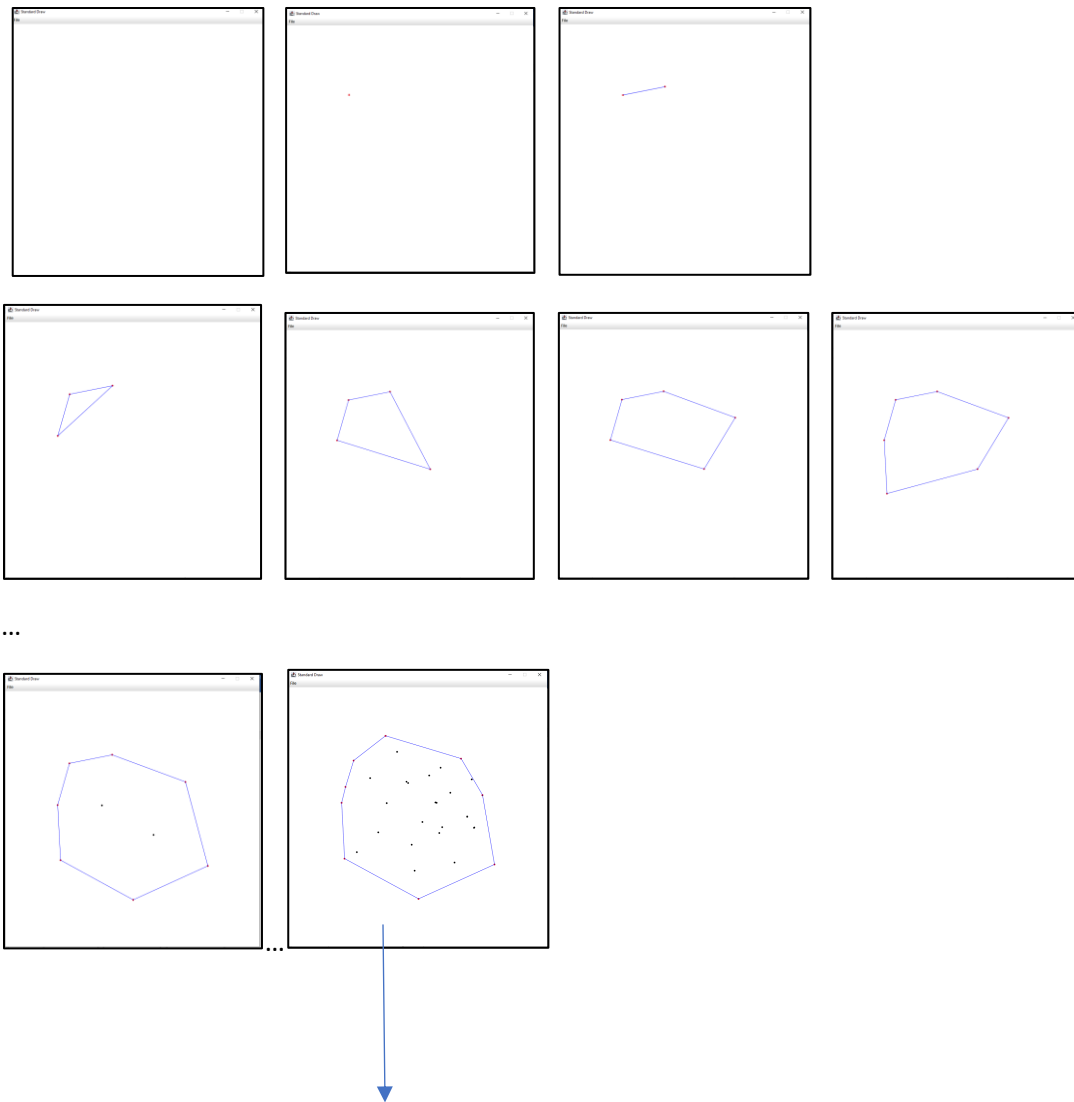
Description of the Assignment

You are provided with `ConvexHullGUI` class which plots the convex hull. Your task is to create `ConvexHullBuilder` class, implement Graham's scan algorithm to find the hull, and make it work with GUI class. You are also asked to create `ConvexHullBuilderTest` Junit class and test your implementation for `ConvexHullBuilder`.
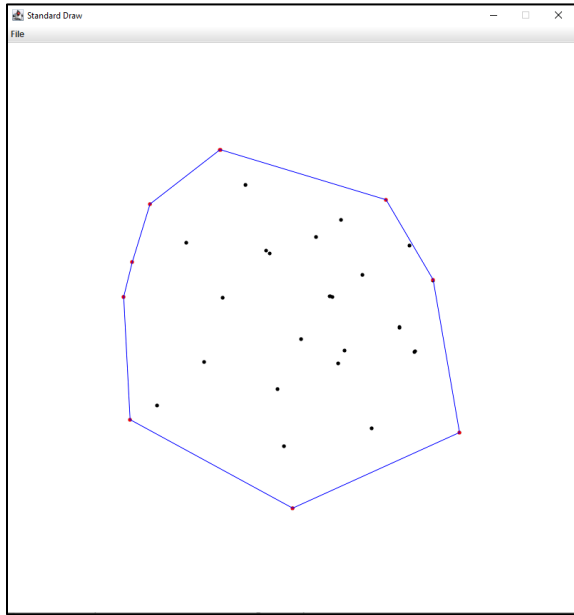
When GUI application is executed, it'll show an empty panel. Once the user clicks any point on the GUI, it'll draw the point, compute convex hull, and repaint the convex hull including all the points that user clicked.

Sample Run: The application starts with an empty canvas. When a user performs mouse clicks, it plots the point and draws the convex hull of points

...

...

Here are a couple of online resources that explains Graham's Scan:

- https://en.wikipedia.org/wiki/Graham_scan
- http://www.cs.ucr.edu/~eldawy/19SCS133/slides/CS133-04-ConvexHull.pdf

Hint: Here is useful information about `Point2D` class that comes with alg4.jar. To compare points, sort by polar degrees or determine if the points are counter clock wise etc., you can use these methods provided with `Point2D` class in the library.

- `Point2D` is a `Comparable` class
- `Point2D`'s `compareTo` method by default compares two points by y-coordinate
- `Point2D`'s `polarOrder` method returns a `Comparator` to compare two points by polar angle
- `Point2D`'s ccw method returns -1 it three points is a clockwise, 0 if it is a collinear and -1 if it is a counterclockwise

What to Submit

- `ConvexHullBuilder.java`
- `ConvexHullBuilderTest.java`