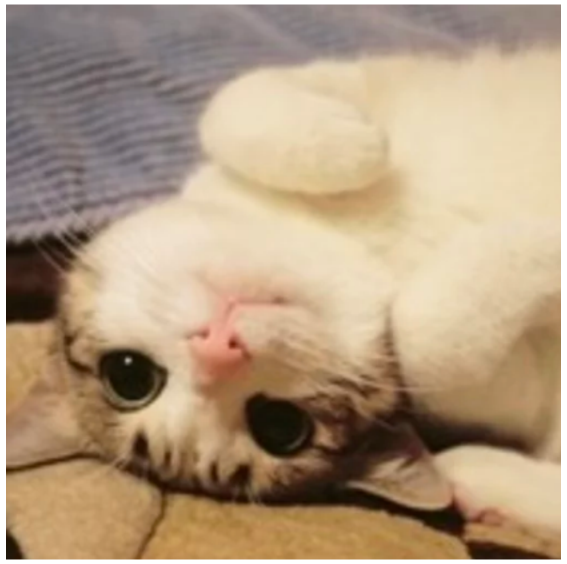
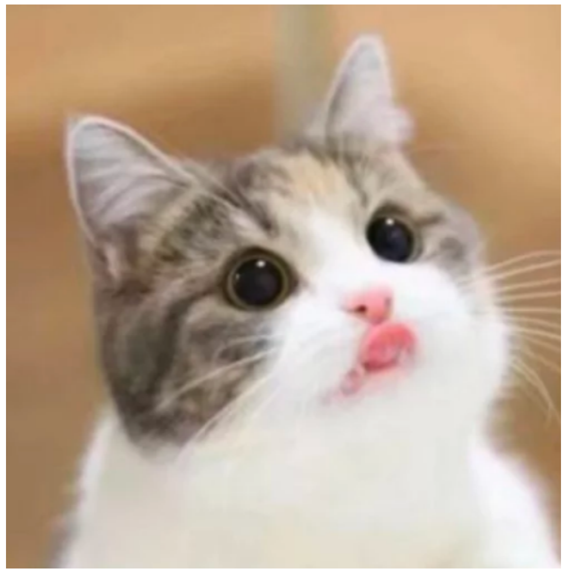


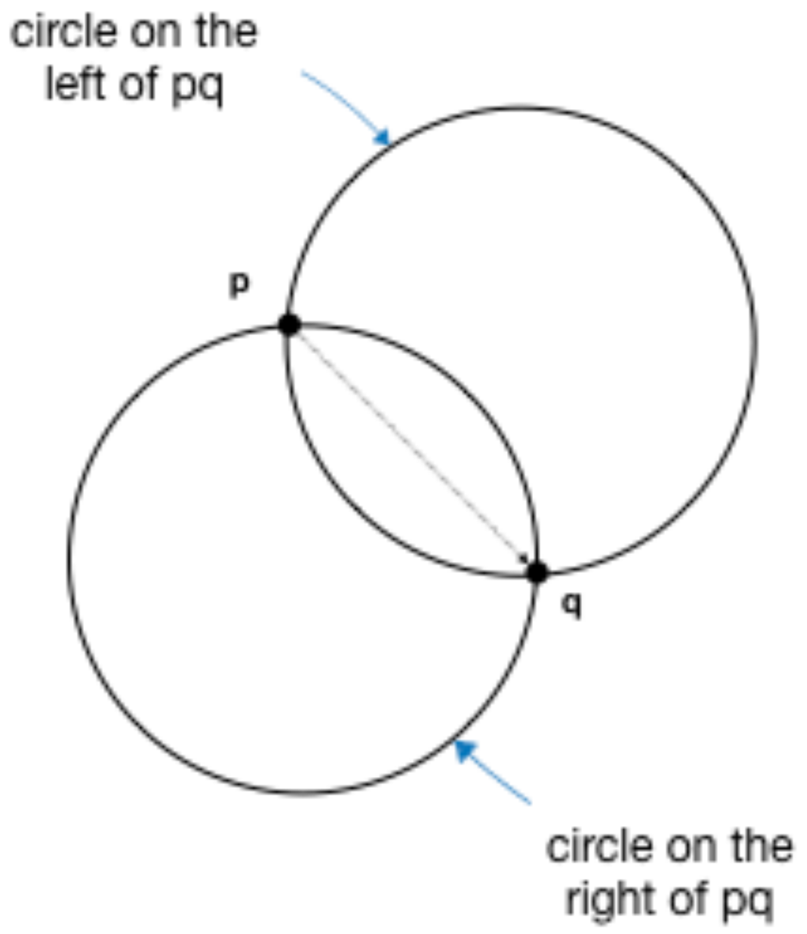
CS2030 LAB #0 & #1

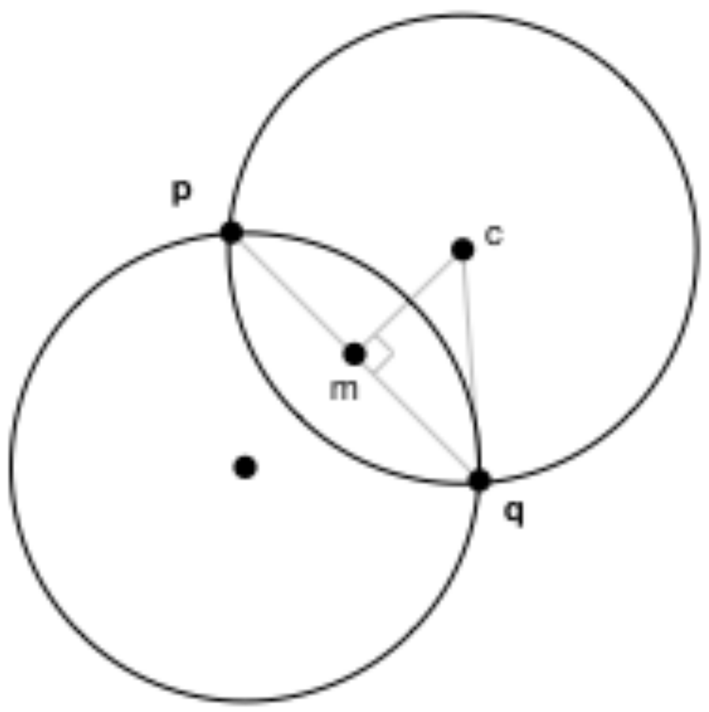
“Points on the Edge of a Circle” &
“Maximum Disk Coverage”



Problem 1

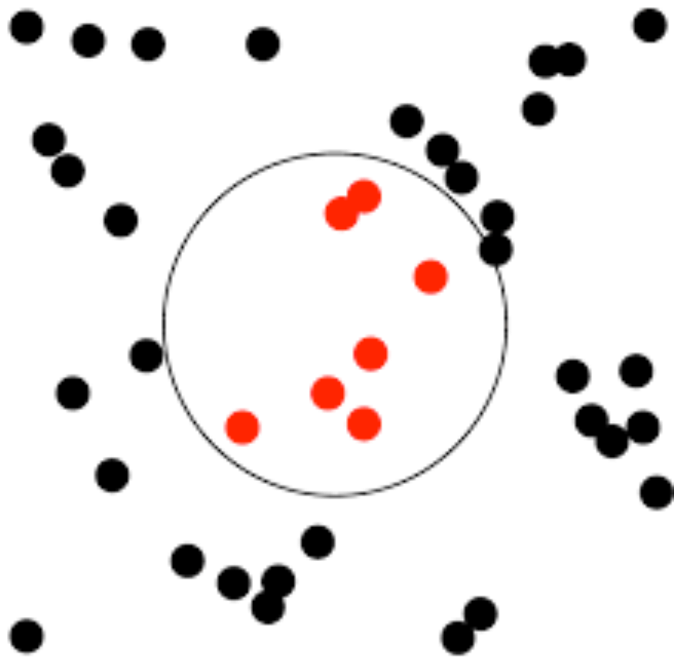
Given two points p and q, construct a circle of radius r such that the two points lie on the edge of the circle.





Problem 2

Given a set of points on a 2D plane, we want to place a unit disc (i.e., a circle of radius 1) so that it covers as many points as possible.



Level 1

Representing a point object with each pair of x- and y-coordinates

Things to do:

1. Write a `Point` class

2. Change the string representation of a `Point` object

```
jshell> /open Point.java

jshell> new Point(0.0, 1.0)
$.. ==> point (0.000, 1.000)

jshell> /exit
```

Level 2

Finding the mid-point and angle (in radians) between two consecutive points

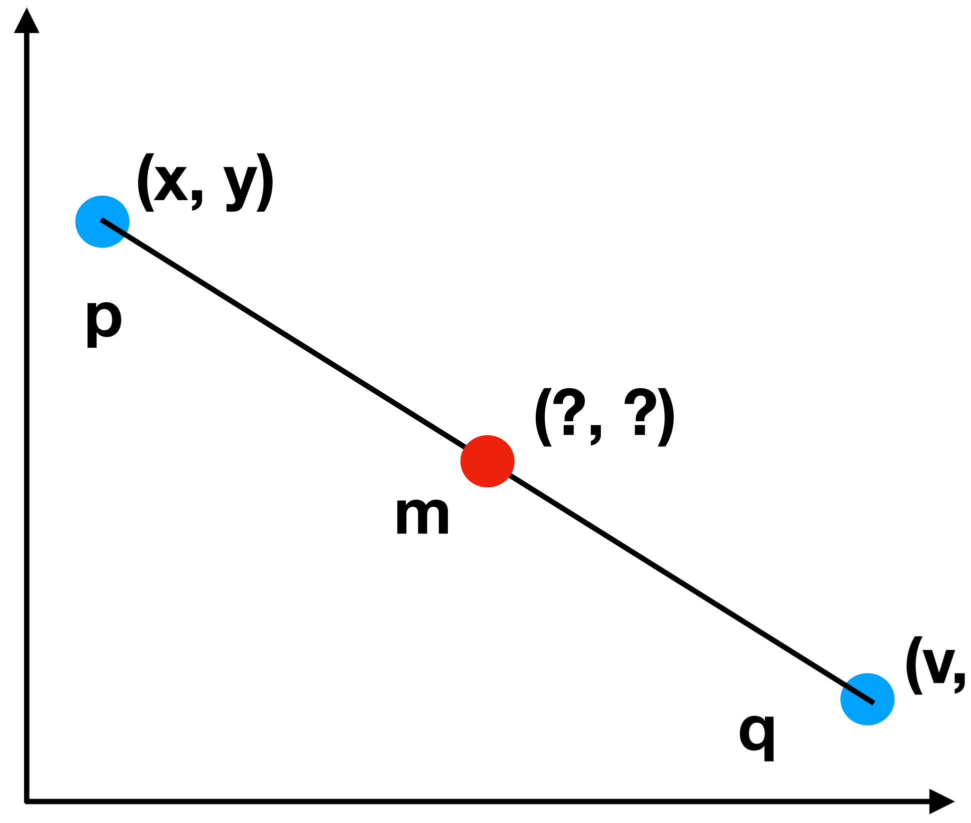
Things to do:

1. Extend the `Point` class with a `midPoint()` method and an `angleTo()` method

```
jshell> /open Point.java

jshell> new Point(0.0,
0.0).midPoint(new Point(1.0, 1.0))
$.. ==> point (0.500, 0.500)

jshell> new Point(0.0,
0.0).angleTo(new Point(1.0, 1.0))
$.. ==> 0.7853981633974483

jshell> new Point(0, 0).angleTo(new
Point(-1, -1))
$.. ==> -2.356194490192345

jshell> /exit
```

`midPoint()`

- Takes in a `Point` as argument; the other `Point` is itself (or `this`)
- Returns another `Point`

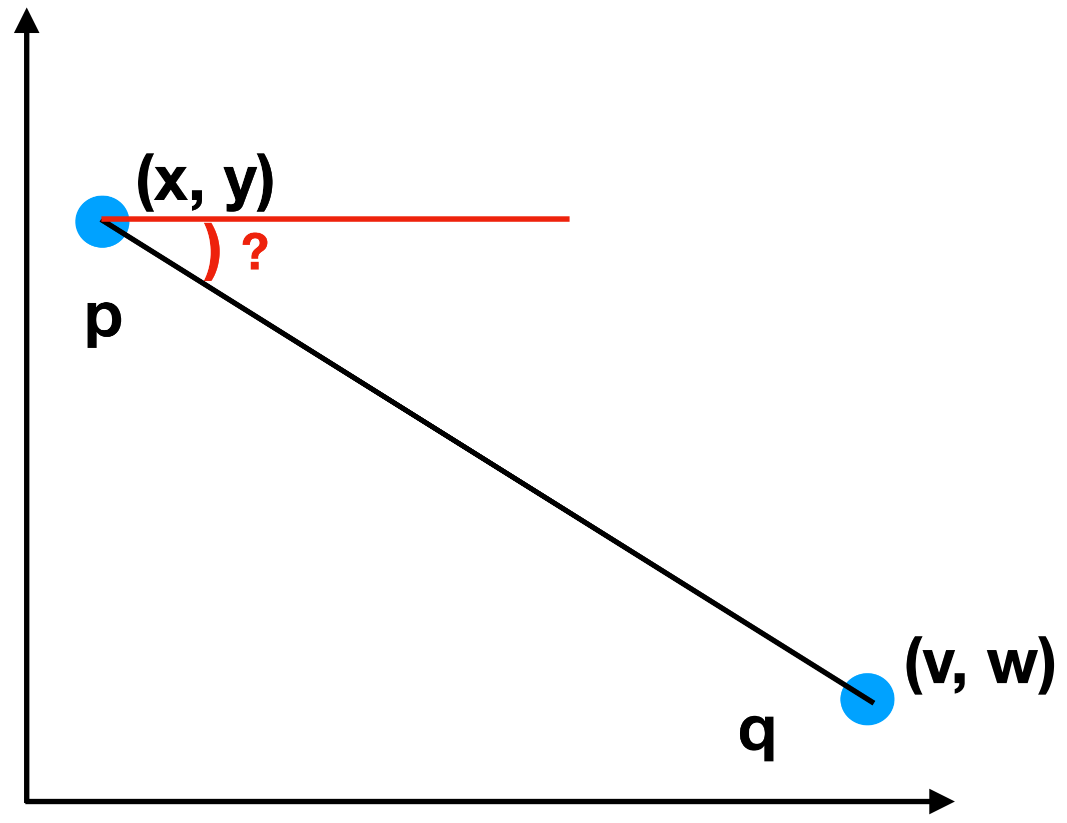


```
class Point {
  private double x;
  private double y;

  public Point(double x, double y) {
    this.x = x;
    this.y = y;
  }

  public Point midPoint(Point that) {
    …
  }
}
```

angleTo()

- Takes in a `Point` as argument; the other `Point` is itself (or `this`)

- Returns a `double`



```
class Point {
  private double x;
  private double y;

  public Point(double x, double y) {
    this.x = x;
    this.y = y;
  }

  public double angleTo(Point that) {
    …
  }
}
```

Level 3

Moving a point at an angle θ and distance d

Things to do:

1. Extend the `Point` class with a `moveTo()` method using the given hint

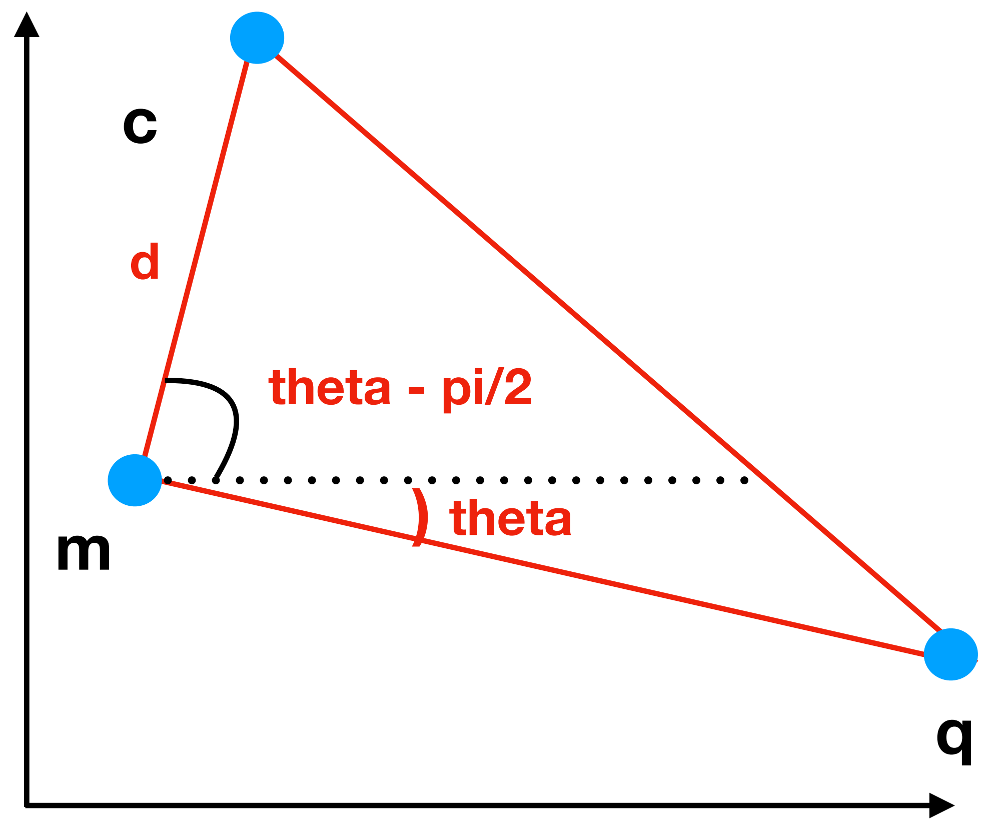
```
jshell> /open Point.java

jshell> new Point(0,
0).moveTo(Math.PI / 2, 1.0)
$.. ==> point (0.000, 1.000)

jshell> /exit
```

Hint: if a point is at (x, y), then moving m at an angle θ and distance d, would result in the new position having the coordinates (x + d cosθ, y + d sinθ)

moveTo()

- Takes in 2 arguments; the angle in radians (`double`) and distance (also a `double`)

- Returns a new `Point`



```
class Point {
  private double x;
  private double y;

  public Point(double x, double y) {
    this.x = x;
    this.y = y;
  }

  public Point moveTo(double theta,
    double d) {
    …
  }
}
```

Level 4 pt 1

Getting to a Circle

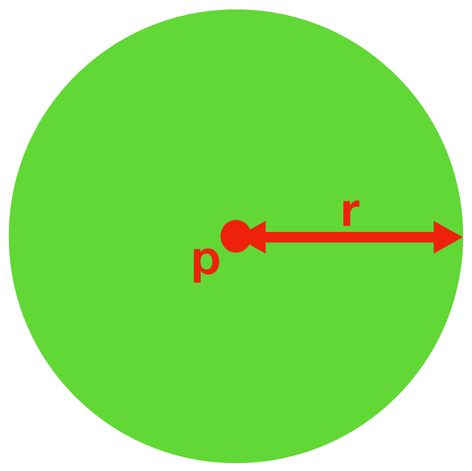
Things to do:

1. Write a `Circle` class
2. Change the string representation of a `Circle` object
3. Return null on invalid `Circles`

```
jshell> /open Point.java

jshell> /open Circle.java

jshell> new Circle(new Point(0.0, 0.0), 1.0)
|  Error:
|  Circle(Point,double) has private access in
Circle
|  new Circle(new Point(0.0, 0.0), 1.0)
|  ^---------------------------------^

jshell> Circle.getCircle(new Point(0.0, 0.0),
1.0)
$.. ==> circle of radius 1.0 centered at point
(0.000, 0.000)

jshell> Circle.getCircle(new Point(0.0, 0.0),
-1.0)
$.. ==> null

jshell> Circle.getCircle(new Point(0.0, 0.0),
0.0)
$.. ==> null
```

getCircle()

- Takes in a `Point` and a radius (double) as argument
- Returns a `Circle`



```
class Circle {
  private Point center;
  private double radius;

    why?
  private Circle(Point p, double r) {
    center = p;
    radius = r;
  }

  public Circle getCircle(Point p,
    double r) {
    …
  }
}
```

Level 4 pt 2

Creating a Circle passing through two consecutive `Points`

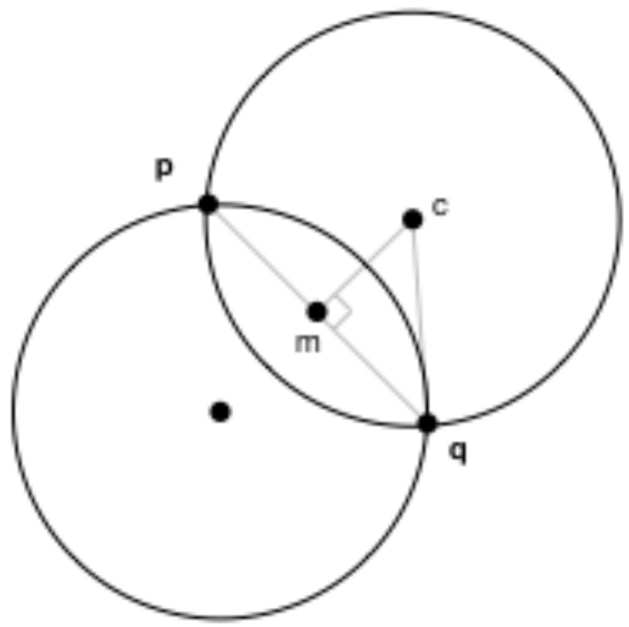
Things to do:

1. Write a `Main` class
2. Have a `createCircle` method
3. Reject null on invalid `Circles`

```
jshell> /open Main.java

jshell> Main.createCircle(new Point(0, 0), new
Point(1, 0), 1)
$.. ==> circle of radius 1.0 centered at point
(0.500, 0.866)

jshell> Main.createCircle(new Point(0, 0), new
Point(1, 0), 2)
$.. ==> circle of radius 2.0 centered at point
(0.500, 1.936)

jshell> Main.createCircle(new Point(0, 0), new
Point(2, 0), 1)
$.. ==> circle of radius 1.0 centered at point
(1.000, 0.000)

jshell> Main.createCircle(new Point(0, 0), new
Point(0, 0), 2)
$.. ==> null

jshell> Main.createCircle(new Point(0, 0), new
Point(3, 0), 1)
$.. ==> null

jshell> /exit
```

createCircle()

- Takes in 2 `Points` and a radius (double) as argument
- Returns a `Circle`



```
class Main {
  public static Circle createCircle(Point p,
    Point q, double r) {
    …
  }
}
```

why?

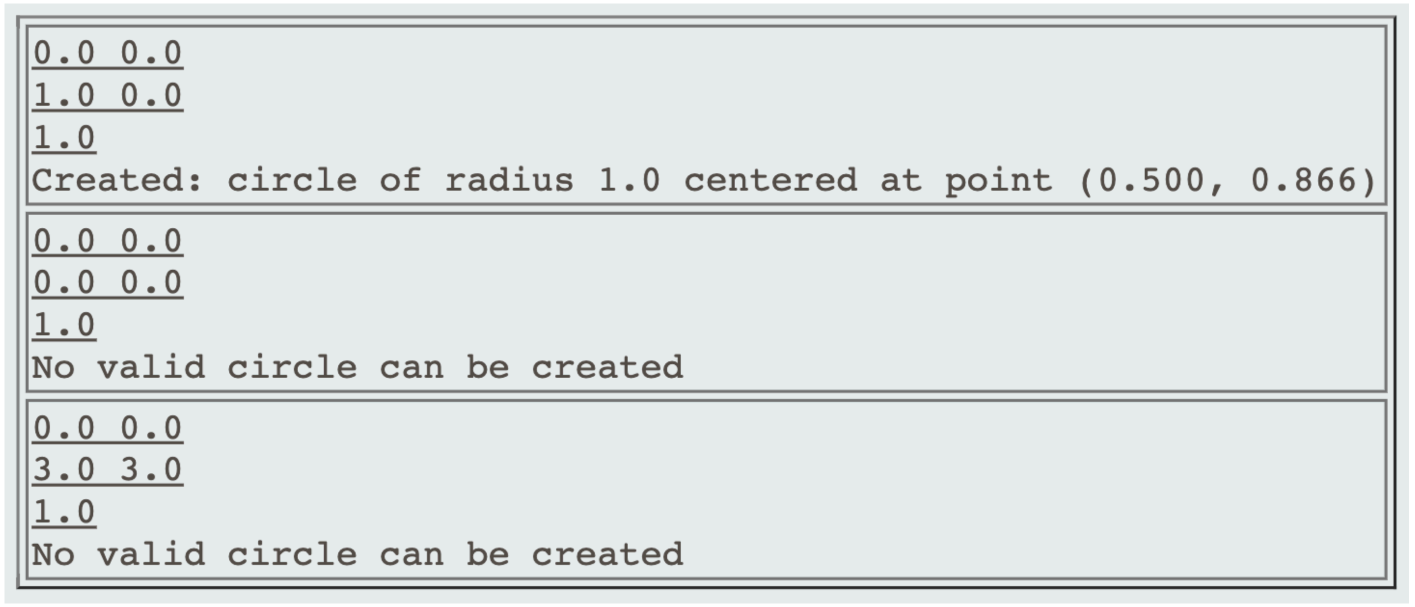
Now, use the methods you have created i.e `midPoint()`, `angleTo()`, `moveTo()` and `getCircle()` to complete the task!

Level 5

Including functionality for input and output; solving for maximum disk coverage

Things to do:

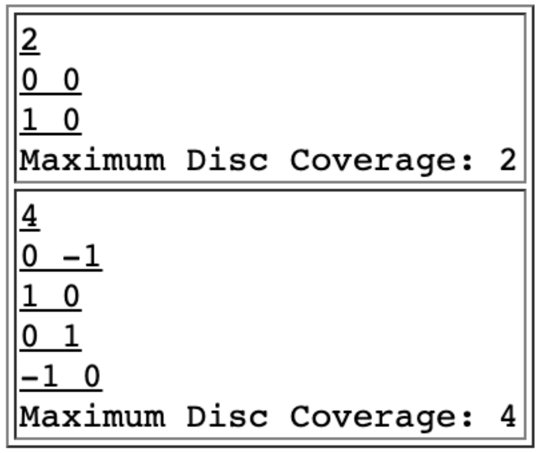
1. Using the `Scanner` class, take in input from the user and return output accordingly



```
0.0 0.0
1.0 0.0
1.0
Created: circle of radius 1.0 centered at point (0.500, 0.866)
```

```
0.0 0.0
0.0 0.0
1.0
No valid circle can be created
```

```
0.0 0.0
3.0 3.0
1.0
No valid circle can be created
```



```
2
0 0
1 0
Maximum Disc Coverage: 2
```

```
4
0 -1
1 0
0 1
-1 0
Maximum Disc Coverage: 4
```

main()

- Takes in nothing initially
- Returns nothing also (void)

```
class Main {
  public static void main() {
    Scanner sc = new Scanner(System.in);
    Point p = new Point(sc.nextDouble(), sc.nextDouble());
    .
    .
    .
  }
}
```

Read the docs!
https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Scanner.html

Maximum Disk Coverage

One way:

1. Use 2 `Points` to construct a `Circle` (if possible)
2. Check how many points are inside the `Circle`
3. Record the disk coverage if it is a new maximum
4. Repeat for all combinations of `points`

All the best!

Wai Lun - wailun@u.nus.edu
Si Mun - simun@u.nus.edu