Exercise 1. Suppose we have a user-defined data type called `Circle` that represents a circle of radius r centered at (h, k) and supports the following API:

`public Circle(double h, double k, double r)`	constructs a circle of radius r centered at (h, k)
`public double area()`	returns the area of *this* circle
`public boolean contains(double x, double y)`	returns `true` if *this* circle contains† (x, y) and `false` otherwise
`public boolean less(Circle other)`	returns `true` if *this* circle is area-wise smaller than *other*, and `false` otherwise
`public boolean equals(Object other)`	returns `true` if *this* and *other* represent the same circle, and `false` otherwise
`public String toString()`	returns a string representation of *this* circle, as `(h, k, r)`

† A point (x, y) is contained in a circle of radius r centered at (h, k) if $(x - h)^2 + (y - k)^2 \leq r^2$

a. Is the `Circle` data type immutable?

b. How do you create a `Circle` object `c1` representing a circle centered at $(1, 1)$ and having radius 2?

c. How do you create a `Circle` object `c2` representing a unit circle centered at the origin?

d. How do you obtain the area of `c1`?

e. How do you check if the point $(1.2, 2.2)$ is contained in `c1`?

f. How do you check if the the circle `c` is area-wise smaller than the circle `d` without invoking the `area()` method explicitly?

g. How do you check if two `Circle` objects `c` and `d` represent the same circle?

h. How do you obtain the string representation of `c1`?

i. Provide code that creates an array `a` of 100 `Circle` objects, each representing a circle centered at the origin and having a random radius from the interval $[0, 1)$.

Exercise 2. Write a program called `Filter.java` that accepts three doubles h, k, and r as command-line inputs, creates a `Circle` object `c` representing a circle centered at (h, k) and having radius r, reads in pairs (x, y) of doubles from standard input representing points on a 2D plane, and writes the fraction of points that fall inside the circle `c`. For example

```
$ python3 filter.py 0 0 3
1 2
3 4
1 5
1 3
<ctrl-d>
0.25
```

Solutions

Solution 1.

a. Yes

b. `Circle c = Circle(1, 1, 2);`

c. `Circle d = Circle(0, 0, 1);`

d. `c.area()`

e. `c.contains(1.2, 2.2)`

f. `c.less(d)`

g. `c.equals(d)`

h. `c.toString()`

i.

```
Circle[] circles = new Circle[100];
for (int i = 0; i < circles.length; i++) {
    Circle c = new Circle(0, 0, StdRandom.uniform(0.0, 1.0));
    circles[i] = c;
}
```

Solution 2.

```
import stdlib.StdIn;
import stdlib.StdOut;

public class Filter {
    public static void main(String[] args) {
        double h = Double.parseDouble(args[0]);
        double k = Double.parseDouble(args[1]);
        double r = Double.parseDouble(args[2]);

        Circle c = new Circle(h, k, r);
        int inside = 0, total = 0;
        while (!StdIn.isEmpty()) {
            double x = StdIn.readDouble();
            double y = StdIn.readDouble();
            total += 1;
            inside += c.contains(x, y) ? 1 : 0;
        }
        StdOut.println(1.0 * inside / total);
    }
}
```