Abstract classes

public abstract class Shape

Circle
getArea

Rectangle
getArea
public abstract class Shape {
    public abstract double getArea();
}

public class Rectangle extends Shape {
    // must define getArea
    public double getArea() {
        return width * height;
    }
}
Shape s = new Shape(x, y);

Shape s = new Rectangle(x, y, w, h);

ok
Shape[] shapes = new Shape[10];

shape[i] = new Circle(x, y, r);

Circle c = (Circle)shape[i];
c.getRadius(); \(\text{risky}\)

for (int i = 0; i < shapes.length; i++) {
  if (shape[i] instanceof Circle) {
    Circle c = (Circle)shape[i];
  } else if (shape[i] instanceof Rectangle) {

}
final modifier
- instance var
- static var
\{ constants
- classes - cannot make a sub-class
- methods - cannot be overridden
interfaces:
- define method signatures
- do not implement the methods
- do not have instance variables
- have constants
**Fillable.java**

```java
public interface Fillable {
    public void fill();
    public Color getColor();
}
```

**Circle.java**

```java
public class Circle extends Shape
    implements Fillable {
    draw() // from Shape
    getArea() // from Shape
    fill() // from Fillable
    getColor() // from Fillable
}```