정보 은닉 그리고 캡슐화

정보 은닉 그리고 캡슐화

- 정보(클래스 인스턴스 변수) 은닉의 이유

```
public class Circle {
	double rad;   // 원의 반지름
	final double PI = 3.14;
	public Circle(double r) {
		setRad(r);
	}
	public void setRad(double r) {
		if (r < 0) {     // 반지름은 0보다 작을 수 없음
			rad = 0;
		} else {
			rad = r;
		}
	}
	public double getArea() {
		return (rad * rad) * PI;
	}
}
```

정보 은닉 그리고 캡슐화

```
public class UnsafeCircle {
	public static void main(String args[]) {
		Circle c = new Circle(1.5);
		System.out.println(c.getArea());
		c.setRad(2.5);
		System.out.println(c.getArea());
		c.setRad(-3.3);
		System.out.println(c.getArea());
		c.rad = -4.5;		// 옳지 않은 접근 방법, 문제되는 부분
		System.out.println(c.getArea());
	}
}
```

- 원의 반지름은 음수가 될 수 없어 Circle 클래스에서 0보다 작을 경우 0으로 처리
- UnsafeCircle 클래스에서 rad에 음수 저장
- 외부에서 클래스의 인스턴스 변수에 직접 접근하면 잘못된 값의 저장이 발생할 수 있어 직접 접근하지 못하도록 설계 하는 것을 정보 은닉이라 함
- 정보 은닉을 위해 클래스의 인스턴스 변수를 private 키워드로 설정하고 메소드를 이용하여 접근하도록 설계

정보 은닉 그리고 캡슐화

- 정보 은닉 적용

```
public class Circle {
    private double rad;// 원의 반지름
    private final double PI = 3.14;
    public Circle(double r) {
        setRad(r);
    }
    public void setRad(double r) {
        if (r < 0) {  // 반지름은 0보다 작을 수 없음
            rad = 0;
        } else {
            rad = r;
        }
    }
    public double getRad() {
        return rad;
    }
    public double getArea() {
        return (rad * rad) * PI;
    }
}
```

정보 은닉 그리고 캡슐화

```
public class InfoHideCircle {
    public static void main(String args[]) {
        Circle c = new Circle(1.5);
        System.out.println("반지름 : " + c.getRad());
        System.out.println("넓 이 : " + c.getArea());

        c.setRad(3.4);
        System.out.println("반지름 : " + c.getRad());
        System.out.println("넓 이 : " + c.getArea());
    }
}
```

- 클래스 Circle의 rad 인스턴스 변수를 privated으로 선언 -> 변수 rad는 클래스 내부에서만 접근 허용
- 클래스 Circle 외부에서는 setRad() 메소드를 통해 변수에 값을 설정하며, getRad() 메소드를 통해 변수 값을 참조
- 값의 참조를 위한 메소드를 getter, 값의 설정을 위한 메소드를 setter 함
- getter
  - 인스턴스 변수의 값을 참조하는 용도로 정의된 메소드
  - 변수의 이름이 name일 때, 메소드의 이름은 getName으로 짓는 것이 관례
- setter
  - 인스턴스 변수의 값을 설정하는 용도로 정의된 메소드
  - 변수의 이름이 name일 때, 메소드의 이름은 setName으로 짓는 것이 관례

정보 은닉 그리고 캡슐화

- 접근 수준 지시자 : public, protected, private, default
  - default는 아무 선언을 하지 않았을 경우
  - 클래스 정의 대상 : public, default
  - 인스턴스 변수, 메소드 대상 : public, protected, private, default

지시자	클래스 내부	동일 패키지	상속 받은 클래스	이외의 영역
private	o	X	X	X
default	o	o	X	X
protected	o	o	o	X
public	o	o	o	o

정보 은닉 그리고 캡슐화

- 캡슐화
  - 정보 은닉과 더불어 객체지향 기반의 클래스 설계에 있어 가장 기본이면서 중요한 원칙
  - 하나의 목적을 이루기 위해 관련 있는 것들만 하나의 클래스에 정의
  - 잘못된 캡슐화
    - 사람 클래스에 "하늘을 날다" 메소드 정의 -> "하늘을 날다" 메소드는 새 클래스에 정의하는 것이 어울림
    - 삼각형 클래스에 "원주율" 인스턴스 변수 정의 -> "원주율" 인스턴스 변수는 원 클래스에 정의하는 것이 어울림

정보 은닉 그리고 캡슐화

```
public class Point {
    int xPos;
    int yPos;
    public Point(int x, int y) {
        xPos = x;
        yPos = y;
    }
    public void showPointInfo() {
        System.out.println("[" + xPos + ", " + yPos + "]");
    }
}
```

1. 위 클래스를 참고하여 원을 의미하는 TestCircle 클래스를 정의하시오. TestCircle 클래스는 좌표 상의 위치 정보(원의 중심 좌표)와 반지름의 길이 정보를 저장할 수 있어야 합니다. 그리고 다음 수준의 main 메소드를 기반으로 Circle 클래스를 테스트 하시오.

```
public static void main(String[] args) {
    Circle c = new Circle(2, 2, 4);          // 좌표 [2, 2] 반지름 4인 원의 생성
    c.showCircleInfo();
}
```

위의 main 메소드에서 showCircleInfo 메소드 호출을 통해서 원의 정보를 출력했을 때, 원의 좌표 정보와 반지름 정보는 반드시 출력이 되도록 구현해야 합니다.