10. 패턴 (Pattern) //src: ch13_pattern 참고

- 객체지향 언어의 장점들을 모아 가장 효율적으로 개발 할 수 있게 만들어 놓은 틀(프레임)
- 보다 빠르게 개발하고 효율적인 유지보수를 위해 정형화 시켜 놓은 것(패턴, 디자인 패턴)

1) 싱글톤 패턴(Singleton pattern)

(1) 싱글톤 패턴

; 클래스에 유일한 객체를 만들어 여러가지 상황에서 동일한 객체에 접근하도록 만들어진 패턴

- 싱글톤 패턴을 따르는 클래스는 객체를 유일하게 하나만 만들 수 있음
- 생성자가 여러 차례 호출 되어도 실제로 생성되는 객체는 하나
- 해당 객체에 접근할 수 있는 모든 곳에서 하나의 객체에만 접근(**전역적인 접촉점**을 제공)

(2) 싱글톤 클래스 예

```
public class SingletonClass {
	private int i;
	private static SingletonClass INSTANCE; //생성자 대신할 변수 생성
	private SingletonClass() {//생성자 함수가 private이므로 외부에서 new 이용해서 생성X
		i=10;
	}
	public static SingletonClass getInstance() {
	//객체가 생성되기 전 데이터 영역의 클래스 상태에서 바로 접근가능한 메소드
		if(INSTANCE==null) {
			INSTANCE = new SingletonClass(); //객체 생성(여기서만 가능)
		}
		return INSTANCE;
	}
	public int getI() {
		return i;
	}
	public void setI(int i) {
		this.i = i;
	}
}
```