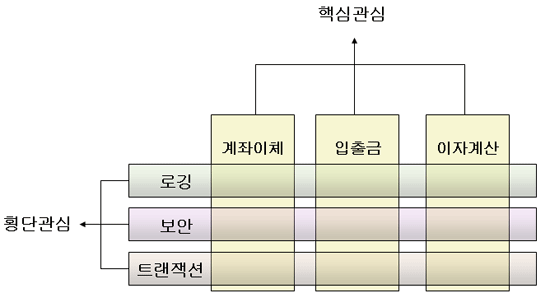
**관점 지향 프로그래밍**

**(AOP: Aspect Oriented Programming)**



: OOP를 더욱 발전시킨 개념으로, 핵심 관심에서 가지는 공통 모듈(횡단 관심)을 분리시켜 외부의 다른 클래스에 존재하게 하도록 하는 것.

• 핵심 관심: 비즈니스 로직을 통해 구현해야 하는 핵심 기능

• 횡단 관심: 핵심 관심에서 중복되는 기능들

**장점)**

- 같은 기능 구현을 위한 복붙을 방지해 중복 코드가 사라짐.

- 자신의 비즈니스 코드에 집중할 수 있음.

- 재활용성이 높아짐.

**구성요소)**

**①JoinPoint**: 횡단 관심을 삽입할 시점 -> When?

②**PointCut**: 어떤 클래스의 어떤 함수에 삽입할지 -> Where?

③**Aspect**: JointPoint + PointCut

④**Advice**: 무엇을 삽입할지 즉, 횡단 관심에 해당하는 코드 -> What?

⑤**Weaving**: Aspect + Advice 를 통해 새 proxy 객체 생성

⑥**Proxy**: 횡단 관심을 실행하는 대리자

출처:

<https://greendreamtrre.tistory.com/601>,

<https://velog.io/@gillog/AOP%EA%B4%80%EC%A0%90-%EC%A7%80%ED%96%A5-%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D>