✓ Exception

- ✓ Java 의 Exception 을 이해하고 활용한다.
- ✓ try-catch-finally, throws 등 처리방법을 활용한다.
- ✓ 사용자 정의 Exception 을 작성한다.

*K*OSTA 은행은 새로운 고객 관리 시스템을 구축하고 있다. 이전 ws 코드에 이어 BankService 클래스에 비즈니스 로직과 관련한 2개의 메소드를 추가하고 이를 실행하는 과정에서 발생할 수 있는 예외상황을 처리하고자 한다. BankService 에 추가되는 2개의 메소드와 이를 처리할 때 체크하고 처리할 예외사항 아래와 같다. 이를 사용자 정의 Exception 으로 만들고 적용해 본다.

✓ Exception

1. getUserAccount : 고객의 일련번호와 계좌 일련번호를 입력받아 계좌 정보를 리턴 한다. 이 때, 일치하는 계좌가 없을 경우 UserAccoutNotFoundException 을 발생시키고 이 요청을 수행하는 코드에서 처리한다.
2. withdraw :고객의 일련번호, 계좌 일련번호, 그리고 출금금액을 입력받아 출금 처리한다. 이 때, 잔액이 부족할 경우, BalanceLackException 을 발생시키고, 이 요청을 수행하는 코드에서 처리한다.

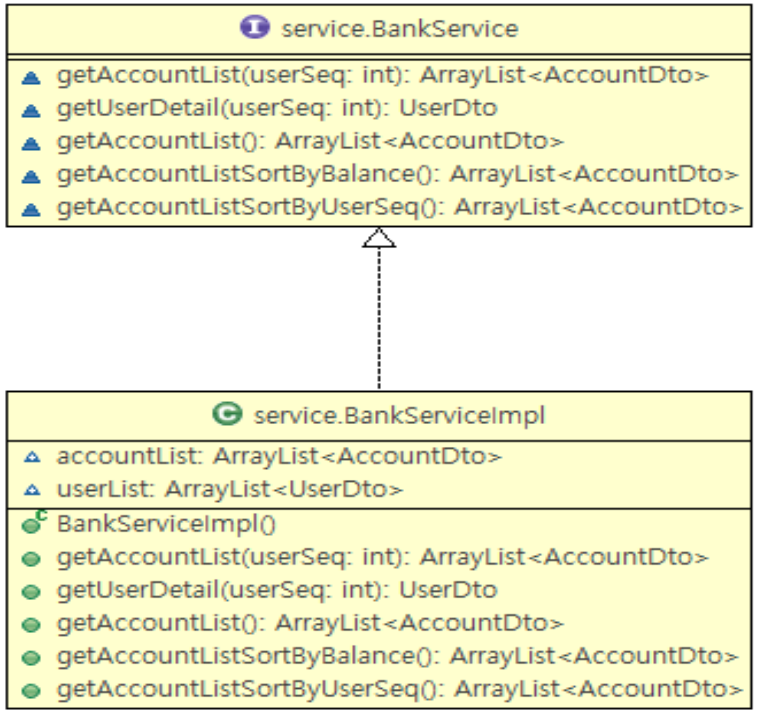
3. Test 클래스에 위 2개의 메소드를 테스트 하는 코드를 추가한다.

Test 출력하는 결과 예시

```
사용자 또는 계좌를 찾을 수 없습니다.
잔액이 부족합니다.
```

✓ Exception

작성하는 코드는 아래의 클래스 다이어그램을 참고한다.



✓ Exception

작성하는 코드는 아래의 클래스 다이어그램을 참고한다.

*** BankService interface에 아래의 두 메소드 추가**

1. AccountDto getUserAccount(**int** userSeq, **int** accountSeq)
   throws UserAccountNotFoundException;

2. int withdraw(**int** userSeq, **int** accountSeq, **int** amount)
   throws BalanceLackException, UserAccountNotFoundException;