12. 예외처리(Exception handling) //src: ch15_exception 참고

1) 예외(Exception)

(1) 문법 에러: 문법적으로 나타나는 에러, 컴파일 불가하므로 실행 불가

(2) 실행 에러: 논리 에러, 시스템 에러, 예외 사항(프로그램 실행 중 발생하는 예기치 못한 사건)

2) 예외처리 필요성

- 예외가 발생하더라도 계속해서 프로그램이 작동되도록 하기 위해(보험)

3) 예외처리 문법

(1) try ~ catch, (finally)

```
try {
        try 블럭; //예외 발생 가능성이 있는 명령문(문제가 발생할 수 있는 로직)
    } catch(예외type e) {//e는 예외 변수
        예외 발생시 처리할 명령문(try 블록 안에 문제가 발생했을 때 대처방안);
    } finally {
        예외 발생 여부와 상관 없이 마지막에 실행할 명령문;
    }                                                    ※ catch문 예외처리 여러 개 가능
```

(2) throws

```
public void 메소드() throws Exception {
        if(예외가 발생하는 조건) {
                throw new Exception("에러 메시지"); //강제로 예외 발생
        }//예외가 발생하지 않았을 때 실행할 내용
```

※throw의 경우 해당 메소드를 호출한(실행하는) 곳에서 try~catch 문으로 예외처리 필요

(단, 예외가 발생하지 않는다는 전제하에 메인함수에서 throw 가능)

※자주 나오는 예외 종류

- ArrayIndexOutOfBoundsException; 배열 사용시 존재하지 않는 index값 호출시 발생
- NullPointerException; 존재하지 않는 객체 가리킬 때 발생
- NumberFormatException; 숫자로 변경할 수 없는 문자열 변경할 시 발생
- ClassNotFoundException; (DB관련) 드라이브 이름을 찾지 못했을 시 발생
- SQLException; db url, id, pw가 올바르지 않을 때 발생