06/20 월

220617-01 exception

복습 예외처리

**Try-catch 흐름 기억하기**

기본

→ 프로그램 문장 흐름에서 예외가 발생하면 그 즉시 프로그램이 종료됨.

예외처리

→ 예외가 발생할 수 있는 가능성이 있는 문장들을 try-catch문을 사용,

예외 객체를 처리함으로써 원래 흐름으로 흘러 갈 수 있게끔 만듦.

예외가 발생하면 catch문장으로 이동;

그렇지 않으면 catch문 건너뛰고 위에서 밑으로 정상적인 문장흐름으로 흘러감.

예외가 발생해도, catch문장에 그 예외 타입이 정의되어 있지 않다면 catch에서 잡지 못한다.

printStackTrace(); → 예외가 발생되었던 경로를 콘솔 창에 출력해준다.

Catch {}블록에서는 각각의 지역 변수를 가진다.

Try catch {} 블록 바깥에 더 큰 변수를 선언하면 양쪽 블록 안에서 모두 사용이 가능하다.

타입에 맞춰서, 해당되는 예외를 잡는 것이 가장 베스트이지만, 범용적으로 매우 큰 범위를 잡을 때 하나의 타입으로 잡을 수 있다.

모든 예외 객체들은 up-casting이 가능하다.

→ 상위 타입으로 잡아 놓으면 모든 타입의 예외가 잡힌다.

→ Exception

→슈퍼 클래스인 Exception을 catch문의 위에 적으면, 밑에 있는 예외들은 자식이기 때문에 도달할 수 없게 되어 컴파일에러가 발생한다.

try catch를 적지 않아도 보이는 예외들은 un-checked 라고 부른다.

**Finally** → 예외의 발생 유무에 상관없이 무조건 실행.

그 이후 원래 흐름대로.

Try catch finally 에서 catch가 없이도 finally 문장 사용가능.

String builder

220617-02 string builder

Append는 void를 리턴하는 것이 아니라 자기 자신을 참조하여 오른쪽에 붙여준다.

String str = "원본";

str.concat(",one="); // 새로운 참조를 만들어, 그 참조를 반환하는 것, 원본 = str 은 유지된다. 불변.