#7 코드의 흐름을 바꾸는 흐름제어와 조건을 합쳐 판단하는 논리연산자

✔ return
함수를 종료하고 값을 반환하는 역할

✔ break
반복문 내의 구문이 실행되는 중간에 즉시 반복문을 종료하고 다음 구문으로 넘어가는 역할

✔ continue
다음 반복 조건으로 즉시 넘어가는 역할

```
for main() {
    for (i in 1..10){
        if (i == 3) break
        println(i)
    }
}
```

► i가 3이 될 경우 중단됨
► 실행결과: 1
2

```
for main() {
    for (i in 1..10){
        if (i == 3) continue
        println(i)
    }
}
```

► I가 3이 되는 시점에 println을 하지 않고 바로 for문으로 돌아가 다음 반복 조건인 4로 진행되어 실행시켰을 때 3을 제외한 모든 숫자가 찍히게 된다
► 실행결과: 1 2 4 5 6 7 8 9 10

✔ label 기능
코틀린에서는 더 나아가 다중 반복문에서 break나 continue가 적용되는 반복문을 label을 통해서 지정할 수 있는 기능이 추가되었다.

<예시> i가 1이고 j가 2이면 모든 반복문을 종료해야 하는 식

■ **기존 언어**

```
for main() {
    for (i in 1..10){
        for (j in 1..10) {
            if(i==1 && j==2) break
        }
        if(i==1 && j==2) break
    }
}
```

*&&연산자: 두 조건을 모두 만족해야 하는 논리연산자
고전적인 언어에서는 내부 반복문에서 조건을 체크하여 break를 하더라도 외부 반복문에서 또 다시 조건을 체크하여 모든 반복문을 수동으로 종료해야만 한다.

■ **코틀린**

```
for main() {
    loop@for (i in 1..10){
        for (j in 1..10) {
            if(i==1 && j==2) break@loop

            println ("i:$i, j:$j")
        }
    }
}
```

► 외부 반복문에 레이블 이름과 @ 기호를 달고 **continue/break** 문에서 @과 레이블 이름을 달아주면 레이블이 달린 반복문을 기준으로 즉시 **continue/break**를 시켜준다.
*레이블 이름은 원하는 이름으로 달기

►따옴표 안에서 변수를 출력할 때는 변수명 앞에 $표기를 해주면 변수 내용으로 대체되어 출력된다.

► 수행결과: I : 1, j : 1

✔ **논리연산자**

논리 값을 연산하여 새로운 논리값을 도출할 때 쓰는 연산자

&&	**and 연산자**	**true && true => true** 둘다 true 인 경우 true
||	**or 연산자**	**true || false => true** 둘 중 하나라도 true 인 경우 true
!	**not 연산자**	**!true => false** **!false => true** 뒤에 붙은 값을 반전시킴

```
for main() {
    println(true && true)
    println(true || false)
    println(!true)
    println(!false)
}
```

► 수행결과:
false
true
false
true

실제 사용시에는 조건식과 조건식을 연산하는 경우가 더 많다

```
for main() {
    var a = 6
    var b = 4
    println(a > 5 && b > 5)
}
```

► 수행결과:
false