왕초보 코틀린 #4 똑똑한 타입추론과 편리한 함수

✔ 타입 추론 type inference

변수나 함수들을 선언할 때나 연산이 이루어질 때 자료명을 코드에 명시하지 않아도 코틀린이 자동으로 자료형을 추론해주는 기능

변수가 선언될 때 할당된 값의 형태로 해당 변수가 어떤 자료형을 가지는지 추론이 가능하기 때문에 가능

```
var a = 1234    // Int형 변수로 지정
var b = 1234L   // Long형 변수로 지정
var c = 12.45   // Double형 변수로 지정
var d = 12.45f  // Float형 변수로 지정
```

```
var e = 0xABCD
var f = 0b0101010
```

*16진수와 2진수는 Int형으로 추론됨

```
var g = true
var h = 'c'
```

*Boolean과 Char 역시 자료형 없이도 추론 가능

대부분은 코틀린의 타입추론 기능을 이용하여 코드량을 줄일 수 있음.

✔ 함수형 function

특정한 동작을 하거나 원하는 결과값을 연산하는데 사용

예) main() println('')

▪ 값을 입력하면 더해주는 함수 만들기

```
fun add(a:Int, b:Int, c:Int) : Int {
    return a + b + c
}
fun main() {
    println(add(5, 6, 7))
}
```

①함수를 지칭하는 fun명령문과 함수 이름을 써준다.

② 괄호 안에 함수가 받아야 할 Int 정수값 a, b, c를 써준다.

③ 괄호를 닫고 a, b, c를 더해서 반환하는 값의 자료형을 써준다. ► 반환형
(반환값이 없으면 생략해도 됨)

④ 다음은 함수의 내용을 만들기 위해 중괄호를 사용한다.
중괄호 안에는 함수가 해야할 구문을 적는다.

⑤ return a + b + c 로 기술한다.
함수 안에서 return은 뒤에 오는 값을 반환하는 키워드로 return이 발생하면 함수의 중간이더라도 값을 반환하고 함수를 종료하게 된다.

⑥ 실제로 사용하기 - println(add(5, 6, 7))

위의 함수는 여러 가지 일을 하는 것이 아닌 a, b, c를 단순히 더해서 반환하는 역할만 하는데 이럴 때는 함수를 좀 더 간단하게 기술할 수 있도록 **단일 표현식 함수 single-expression function**를 지원한다.

```
fun add(a:Int, b:Int, c:Int) = a + b + c
```

또한 단일 표현식 함수에서는 반환형의 타입 추론이 가능하므로 반환형을 생략할 수 있다.
함수는 내부적으로 기능을 가진 형태이지만 외부에서 볼 때 파라미터를 넣는다는 점 외에는 자료형이 결정된 변수라는 개념으로 접근하는 것이 좋다.