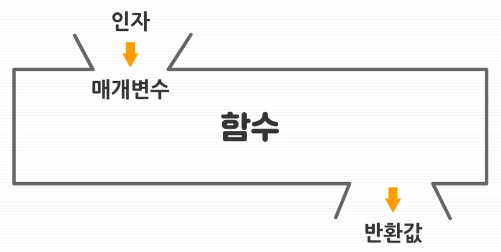
함수란?

: 프로그램에서 어떤 특정 기능을 수행할 목적으로 만들어진 재사용 구조의 코드 부분

함수의 개념과 목적

* 장점
  + 하나의 큰 프로그램을 여러 부분으로 나눌 수 있기 때문에 구조적 프로그래밍이 가능해짐
  + 동일 함수를 여러 곳에서 필요할 때마다 호출할 수 있음
  + 수정이 용이함
* 사용 방법



* 순수 함수 pure function: 결과값 반환 외에 외부에 영향을 주지 않는함수
  + 함수형 프로그래밍 지원 언어에서는 순수 함수를 인자, 반환값으로 사용 가능

함수 호출 및 선언

* print()
  + 괄호 안에 인자 전달
* def 함수명 (필요할 경우 매개변수):
* 명령문 1
* 명령문 2
* 예시)
  + def calc\_sum(x, y): *#매개변수에 인자값 전달*
    - return x + y  *#calc\_sum() 호출한 위치에 반환값이 전달*
    - a, b = 2, 3
    - c = calc\_sum(a, b)  *#반환값 5가 변수 c에 저장*
    - d = calc\_sum(a, c)  *#반환값 7시 변수 d에 저장*
    - print(“결과: {}”.format(c))
    - print(“결과: {}”.format(d))
    - >>> 결과: 5
    - >>> 결과: 7

함수의 유형

* 매개변수, 반환값의 유무로 달라짐

1. 매개변수와 반환값이 있는 함수
   * (parameter) 존재, return 존재
2. 매개변수는 없고 반환값이 있는 함수
   * () 안이 비어있음, return 존재
3. 매개변수는 있고 반환값이 없는 함수
   * (parameter) 존재, no return
4. 매개변수와 반환값이 없는 함수
   * no parameter, no return

함수와 매개변수

* 매개변수: 함수 호출 시 입력값을 전달 받기 위한 변수
  + 전달받은 인자의 값에 의해 타입이 결정됨
* 언팩 연산자(\*): 매개변수의 개수를 가변적으로 사용할 수 있도록 언팩 연산자 제공됨; 매개변수에 적용 시 인자를 튜플 형식으로 처리함
  + def calc\_sum(\*parameter):
    - total = 0
    - for val in parameter:
      * total += val
    - return total
    - ret\_val = calc\_sum(1, 2)
    - ret\_val = calc\_sum (1, 2, 3)  *#1, 2, 3이 인자로 전달되어 for문 3번 반복* ***→*** *6 반환*
    - ret\_val = calc\_sum(1, 2, 3, 4)  *#1, 2, 3, 4가 인자로 전달되어 반환값 10을 변수 ret\_val에 저장*
  + 가변형 매개변수는 하나만 지정할 수 있으며, 가장 마지막 매개변수로 지정해야 부작용 없이 사용가능
  + 명시적 매개변수와 가변 매개변수의 혼합 사용
    - def calc\_sum(precision, \*params):
      * if precision ==0:
        + total = 0
      * elif 0 < precision < 1:
        + total = 0.0
      * for val in params:
        + total += val
      * return total
    - ret\_val = calc\_sum(0, 1, 2)  *#0이 precision에, 1, 2가 \*params에 인자로 전달됨*
    - print(ret\_val)
    - >>> 3
    - ret\_val = calc\_sum(0.0, 1, 2)
    - print(ret\_val)
    - >>> 3.0
  + 언팩 연산자를 사용하는 튜플 형식의 반환값
    - def calc\_sum(precision1, precision2, \*params):
      * if precision1 == 0:
        + total1 = 0
      * elif 0 < precision1 < 1:
        + total1 = 0.0
      * for val in params:  *# 튜플 형식의 매개변수 반복 접근*
        + total1 += val
        + total2 += val
      * return total1, total2
    - ret\_val = calc\_sum(0, 0.1, 1, 2)  *#매개변수 precision1이 0, precision2가 0.1, Params가 1, 2* ***→*** *튜플 (3, 3.0) 반환하여 ret\_val에 저장*
* 키워드 언팩 연산자(\*\*): 매개변수의 개수를 가변적으로 사용할 수 있도록 함; 키워드 인자들을 전달해 매개변수를 딕셔너리 형식으로 처리함
  + 키워드 언팩 연산자를 사용하는 딕셔너리 형식의 가변 매개변수
    - def use\_keyword\_arg\_unpacking(\*\*params):
      * for k in params.keys():
        + print(“{}: {}”.format(k, poarams[k]))
* 기본값을 갖는 매개변수

고급 함수

* 중첩 함수
* Lambda 람다식 함수
  + 사용법) Lambda 매개변수: 반환값