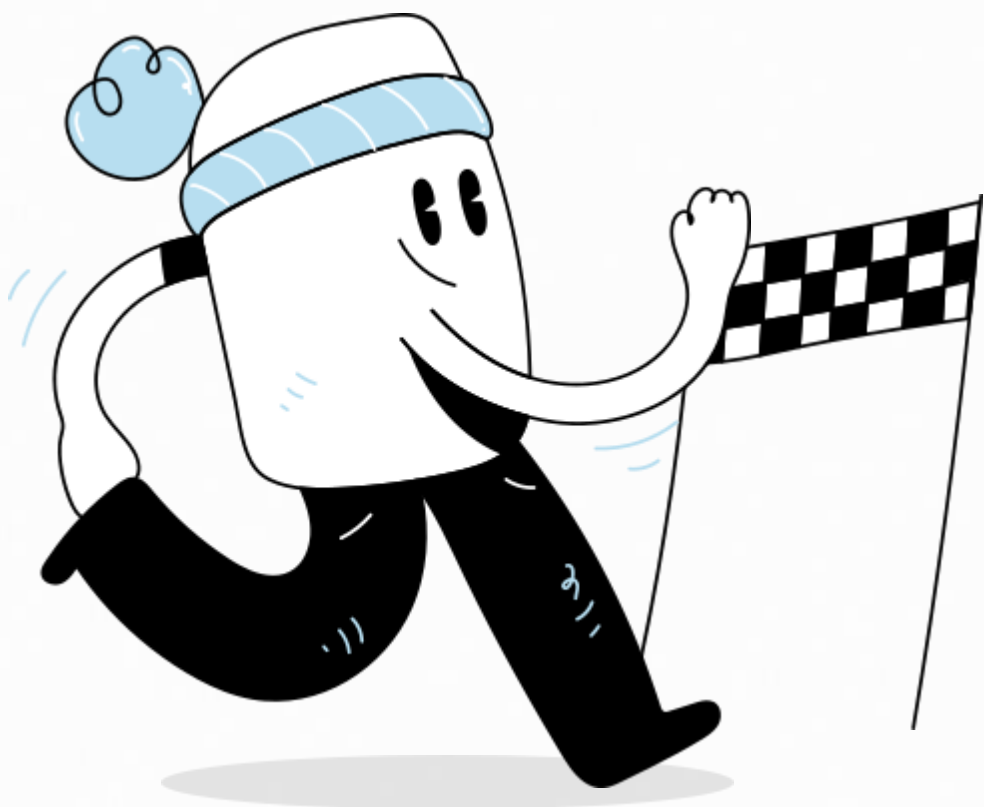


Python 기초 맛보기

Week 5



오늘의 학습 목표



1. 지난 주 수업 복습 퀴즈

1) 횡수 제어 반복문 2) 조건 제어 반복문

2. 지난 주 실습

1) 로그인하기 2) 곱셈 퀴즈 맞추기

3. 함수

1) 함수란? 2) 함수 정의 & 특징 3) 인수, 매개변수

4. 실습

1) get_sum 함수 작성 2) 원의 넓이와 둘레 구하기

1. 지난 주 수업 복습 퀴즈

1 Quiz!

Quiz!

오른쪽의 코드 실행한 후에
0을 입력했더니
"음수"라고 출력되었다.
수정이 필요한 부분은?

```
num = int(input("정수를 입력하세요: "))  
  
if num > 0:  
    if num == 0:  
        print("0")  
    else:  
        print("양수")  
else:  
    print("음수")
```



정수를 입력하세요: 0
음수



Correct answer

오른쪽의 코드 실행한 후에
0을 입력했더니
"음수"라고 출력되었다.
수정이 필요한 부분은?

```
num = int(input("정수를 입력하세요: "))  
if num > 0:  
    if num == 0:  
        print("0")  
    else:  
        print("양수")  
else:  
    print("음수")
```


1 Quiz!

```
num = int(input("정수를 입력하세요: "))
```

```
if num > 0:
```

```
    if num == 0:
```

```
        print("0")
```

```
    else:
```

```
        print("양수")
```

```
else:
```

```
    print("음수")
```



```
num = int(input("정수를 입력하세요: "))
```

```
if num >= 0:
```

```
    if num == 0:
```

```
        print("0")
```

```
    else:
```

```
        print("양수")
```

```
else:
```

```
    print("음수")
```

if의 조건이 "num > 0"인 경우, 0이 해당이 되지 않음!
⇒ 중첩 if문으로 들어가지 않고 아래의 else로 바로 이동

Quiz!

횟수 제어 반복문은 'for 문'을 사용한다.

① O

② X



Correct answer


횟수 제어 반복문은 'for 문'을 사용한다.



② X

Correct answer

횟수 제어 반복(for 문):  정해진 횟수만큼 반복

조건 제어 반복(while 문):  특정한 조건이 만족되면 계속 반복

Quiz!

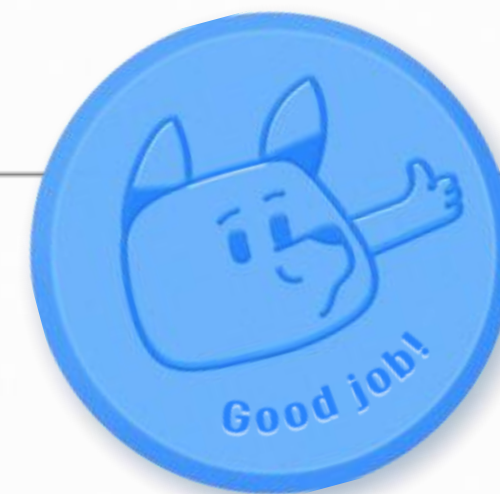
for문에서 횟수를 정하기 위해
사용하는 함수 이름은?

① str()

② print()

③ range()

④ input()



Correct answer

for문에서 횟수를 정하기 위해
사용하는 함수 이름은?

① str()

② print()

③ range()

④ input()

range() 함수

```
for 변수 in range(종료 값):  
    문장
```

```
for i in range(5)  
    print("안녕하세요")
```



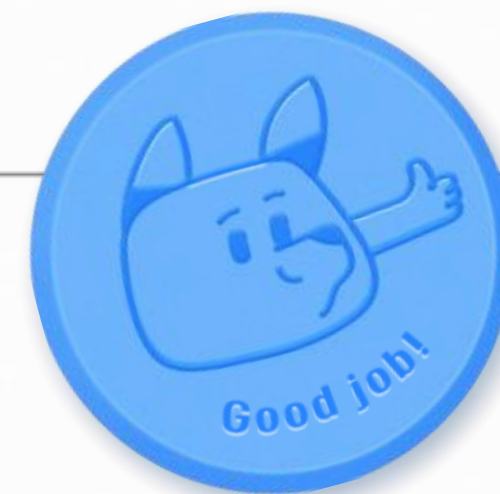
안녕하세요
안녕하세요
안녕하세요
안녕하세요
안녕하세요

***주의할 점: range(종료 값) = 0~(종료값-1)까지의 숫자를 반환한다.**
따라서, i값은 1,2,3,4,5로 변하는 것이 아니라 0,1,2,3,4로 변하면서 반복문은 돈다.
(결과적으로 횟수는 같음)

Quiz!

오른쪽의 코드에서
while문의 반복이 끝날 때
count의 값은?

```
count = 1
sum = 0
while count <= 10 :
    sum = sum + count
    count = count + 1
print("합계는", sum)
```



Correct answer

오른쪽의 코드에서
while문의 반복이 끝날 때
count의 값은?

```
count = 1
sum = 0
while count <= 10 :
    sum = sum + count
    count = count + 1
print("합계는", sum)
```

★ 정답 : 11



2. 지난 주 실습

실습 #1: 로그인하기

```
id = ""
password = ""

while id != "hash" or password != "1234":
    id = input("아이디를 입력하시오: ")
    password = input("비밀번호를 입력하시오: ")
print("로그인 성공")
```



```
아이디를 입력하시오: 12ds4
비밀번호를 입력하시오: 1234
아이디를 입력하시오: hash
비밀번호를 입력하시오: 1234
로그인 성공
```

실습 #1: 로그인하기

```
id = ""  
password = ""
```

-----> 변수(id, password) 선언

```
while id != "hash" or password != "1234":
```

-----> "!=" : ~이 아니다

```
    id = input("아이디를 입력하시오: ")  
    password = input("비밀번호를 입력하시오: ")  
print("로그인 성공")
```

and가 아닌 or을 사용하는 이유는?

And : 두 가지 조건을 모두 만족시켜야 OK.

Or : 두 가지 조건 중 하나만 만족 OK

⇒ 둘 중 하나라도 틀리면 로그인되지 않도록 해야하니까 or을 사용!

실습 #2: 곱셈 퀴즈 맞추기

사용자에게 곱셈 퀴즈를 내고 답을 사용자로부터 받는 프로그램에서 사용자가 올바른 답을 입력할 때까지 반복하도록 수정하여 보자.

```
num=0;
while num != 72:
    num=int(input("8*9의 결과는?"))
print("정답입니다!")
```



```
8*9의 결과는?56
8*9의 결과는?564
8*9의 결과는?72
정답입니다!
```


실습 #2: 곱셈 퀴즈 맞추기

사용자에게 곱셈 퀴즈를 내고 답을 사용자로부터 받는 프로그램에서 사용자가 올바른 답을 입력할 때까지 반복하도록 수정하여 보자.

`num=0;` -----> 변수(num) 선언 & 변수에 0을 저장

`while num != 72:`

`num=int(input("8*9의 결과는?"))`

`print("정답입니다!")`

-----> 올바른 답이 아니면 계속 반복



3. 함수

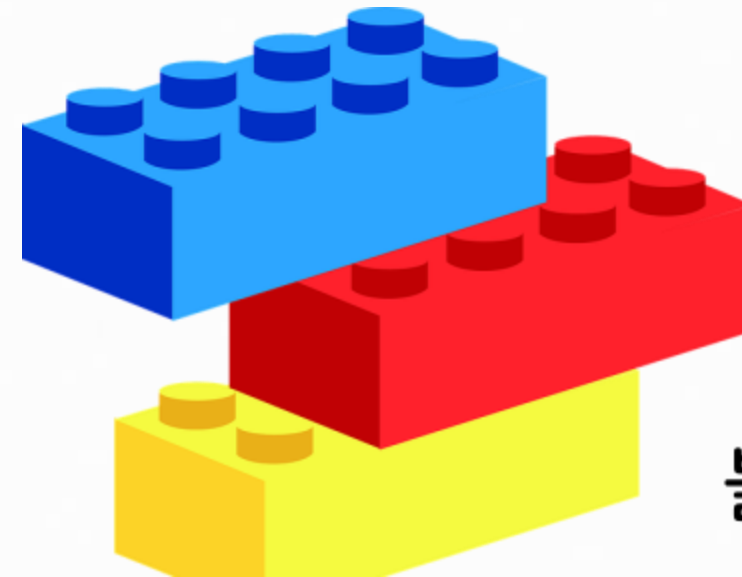
함수 (function)

일을 수행하는 코드의 덩어리

⇒ 더 큰 프로그램을 구축하는 데 사용할 수 있는 작은 조각



블록 한 개: 함수



블록 여러 개: 프로그램

함수 (function)

함수가 필요한 이유?

복잡한 프로그램을 만들게 될 경우,
이해하기 쉽고 관리하기 쉽도록 하기 위해서

함수를 사용하게 되면,

한 번만 코드를 작성해두면
언제든지 필요할 때 불러와서 사용 가능!



함수 정의(생성) : def

함수 생성
키워드

함수의 이름

```
def say_hello():  
    print('Hello, world!')  
    :  
    :
```

} 함수가
수행할
코드

함수 정의를 위해 필요한 요소

- 함수 생성 키워드 def
- 함수의 이름
- 함수가 실행할 코드

def는 파이썬에서 함수 생성을 위해 미리 지정해둔 키워드!

함수 정의(생성) : def

[유의할 점]

1. 함수 이름 뒤에는 괄호 표시()와 콜론(:) 사용!
2. 함수가 수행할 코드들은 반드시 들여쓰기

함수 생성
키워드

함수의 이름

```
def say_hello():  
    print('Hello, world!')  
    :  
    :
```

} 함수가
수행할
코드

예제를 통해 알아보는 '함수의 특징'

[주소를 출력하는 함수 정의]

```
def print_address():  
    print("서울특별시 종로구 1번지")  
    print("파이썬 빌딩 7층")  
    print("홍길동")
```



코드를 작성하고
Run 버튼을 눌러 실행시켜도
아무런 변화가 생기지 X

함수 안의 코드들은 **자동으로 실행X**
정의한 함수를 **호출해야(불러줘야)** 코드가 실행된다!

예제를 통해 알아보는 '함수의 특징'

[주소를 출력하는 함수 정의]

```
def print_address():  
    print("서울특별시 종로구 1번지")  
    print("파이썬 빌딩 7층")  
    print("홍길동")
```



```
서울특별시 종로구 1번지  
파이썬 빌딩 7층  
홍길동
```

```
print_address()
```

정의한 함수를 실행하고 싶을 때는 **함수 이름**을 사용하면 된다!
⇒ 함수 호출 (function call)

함수 : 예제 실습 #1

[주소를 출력하는 함수 정의]

```
def print_address(name):  
    print("서울특별시 종로구 1번지")  
    print("파이썬 빌딩 7층")  
    print(name)
```

```
print_address("홍길동")
```



```
서울특별시 종로구 1번지  
파이썬 빌딩 7층  
홍길동
```

함수 : 예제 실습 #1

[주소를 출력하는 함수 정의]

* 앞에서 했던 함수 선언과 **달라진 부분?**

```
def print_address(name): -----> 괄호() 안에 name이 있음  
    print("서울특별시 종로구 1번지")  
    print("파이썬 빌딩 7층")  
    print(name)
```

```
print_address("홍길동") -----> name 자리에 "홍길동"이 있음
```

함수 : 예제 실습 #1

[주소를 출력하는 함수 정의]

```
def print_address(name):  
    print("서울특별시 중랑구 1번지")  
    print("파이썬 빌딩 7층")  
    print(name)
```

```
print_address("홍길동")
```

함수가 호출되어 실행될 때 name에 문자열 "홍길동"이 전달됨.

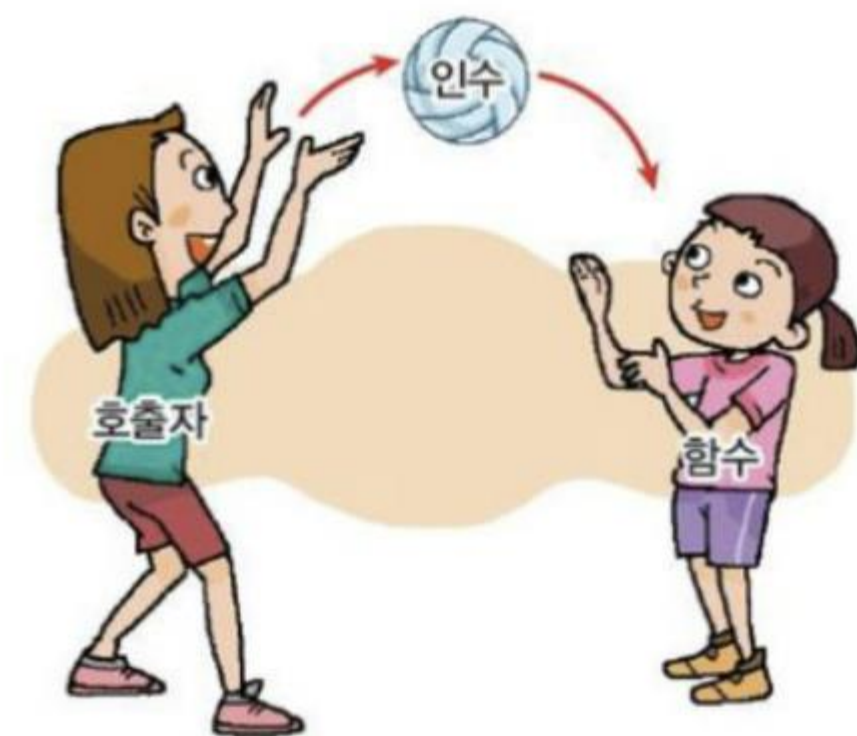
인수 & 매개변수

```
def print_address(name):  
    print("서울특별시 중로구 1번지")  
    print("파이썬 빌딩 7층")  
    print(name)
```

매개변수

```
print_address("홍길동")
```

인수



매개변수(parameter) : 값을 전달 받는 변수
인수(argument) : 전달되는 값

함수 : 예제 실습 #2

[공간 넓이 구하기]

```
def calculate_area(radius):  
    area = 3.14 * radius**2  
    return area  
  
c_area = calculate_area(5.0)  
  
print(c_area)
```



78.5

함수 : 예제 실습 #2

[공간 넓이 구하기]

* 앞에서 했던 함수 선언과 **달라진 부분?**

```
def calculate_area(radius):  
    area = 3.14 * radius**2  
    return area
```

-----> return이라는 키워드가 새롭게 등장!

```
c_area = calculate_area(5.0)
```

-----> 함수를 변수에 저장??

```
print(c_area)
```

반환 값

함수 실행이 끝날 때
변수 area에 담긴 값을 반환

```
def calculate_area(radius):  
    area = 3.14 * radius**2  
    return area
```

```
c_area = calculate_area(5.0)
```

```
print(c_area)
```

함수가 반환한 값을
변수 c_area에 저장



반환 값 : 함수로부터 되돌아오는 값

4. 실습

실습 #1: get_sum 함수 작성

[start에서 end까지의 합을 계산하는 함수]

```
def get_sum(start, end):  
    sum = 0  
    for i in range(start, end+1):  
        sum += i  
    return sum  
  
print(get_sum(1,10))
```



55

실습 #1: get_sum 함수 작성

[start에서 end까지의 합을 계산하는 함수]

```
def get_sum(start, end):  
    sum = 0  
    for i in range(start, end+1):  
        sum += i  
    return sum  
  
print(get_sum(1,10))
```

```
def get_sum(start, end):  
    sum = 0
```

- * get_sum이라는 이름을 가진 함수 선언
- * start와 end는 매개변수

```
for i in range(start, end+1):
```

- * 반복문의 실행 범위 : start부터 end+1

실습 #1: get_sum 함수 작성

[start에서 end까지의 합을 계산하는 함수]

```
def get_sum(start, end):  
    sum = 0  
    for i in range(start, end+1):  
        sum += i  
    return sum  
  
print(get_sum(1, 10))
```

```
        sum += i  
    return sum
```

* `sum += i` 는 `sum = sum+i` 와 동일

* `sum`에 저장되어 있는 값을 함수가 반환
(함수가 일을 끝냈을 때 전달하는 값)

실습 #1: get_sum 함수 작성

[start에서 end까지의 합을 계산하는 함수]

```
def get_sum(start, end):  
    sum = 0  
    for i in range(start, end+1):  
        sum += i  
    return sum  
  
print(get_sum(1,10))
```

```
print(get_sum(1,10))
```

* start 값으로 1, end 값으로 10을 전달

* 함수 실행이 끝난 후,

sum에 저장되어 있는 값을 출력

실습 #1: get_sum 함수 작성

[start에서 end까지의 합을 계산하는 함수]

```
def get_sum(start, end):  
    sum = 0  
    for i in range(start, end+1):  
        sum += i  
    return sum  
  
print(get_sum(1, 10))
```


2개

2개

매개변수의 개수와 전달하는 인자의 개수가 다르면 오류 발생!

실습 #2: 원의 넓이와 둘레 구하기

```
def circleArea(r):  
    print("반지름이 " + str(r) + "인 원의 면적:", pi*r*r)  
  
def circlePerimeter(r):  
    print("반지름이 " + str(r) + "인 원의 둘레:", 2*pi*r)  
  
pi = 3.14  
circleArea(4)  
circlePerimeter(4)
```



반지름이 4인 원의 면적: 50.24
반지름이 4인 원의 둘레: 25.12

실습 #2: 원의 넓이와 둘레 구하기

```
def circleArea(r):  
    print("반지름이 " + str(r) + "인 원의 면적:", pi*r*r)
```

* circleArea라는 이름을 가진 함수 선언

* r은 매개변수

```
def circlePerimeter(r):  
    print("반지름이 " + str(r) + "인 원의 둘레:", 2*pi*r)
```

* circlePerimeter라는 이름을 가진 함수 선언

* r은 매개변수

실습 #2: 원의 넓이와 둘레 구하기

```
pi = 3.14
```

* 전역변수 pi 선언

(함수 외부에서 선언한 변수)

```
circleArea(4)  
circlePerimeter(4)
```

* circleArea 함수의 매개변수에 4 전달

* circlePerimeter 함수의 매개변수에 4 전달

```
def circleArea(r):  
    print("반지름이 " + str(r) + "인 원의 면적:", pi*r*r)  
  
def circlePerimeter(r):  
    print("반지름이 " + str(r) + "인 원의 둘레:", 2*pi*r)
```

```
pi = 3.14  
circleArea(4)  
circlePerimeter(4)
```



오늘의 수업은 끝! 감사합니다~

다음 시간에 만나요!!

