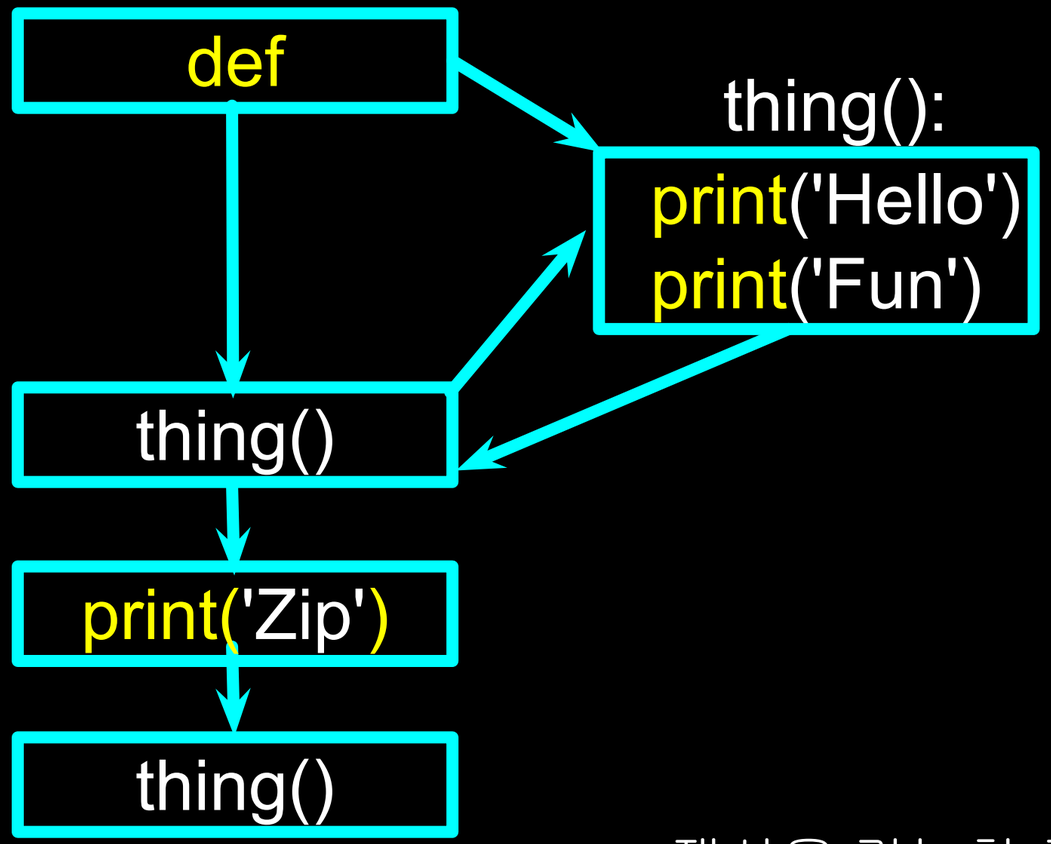
함수

제4장

Python for Everybody

www.py4e.com

저장 (그리고 재사용) 단계



프로그램:

```
def thing():
    print('Hello')
    print('Fun')

thing()
print('Zip')
thing()
```

출력:

```
Hello
Fun
Zip
Hello
Fun
```

재사용 가능한 코드 조각을 “함수”라고 부름

파이썬 함수

- 파이썬에는 두 종류의 함수가 존재

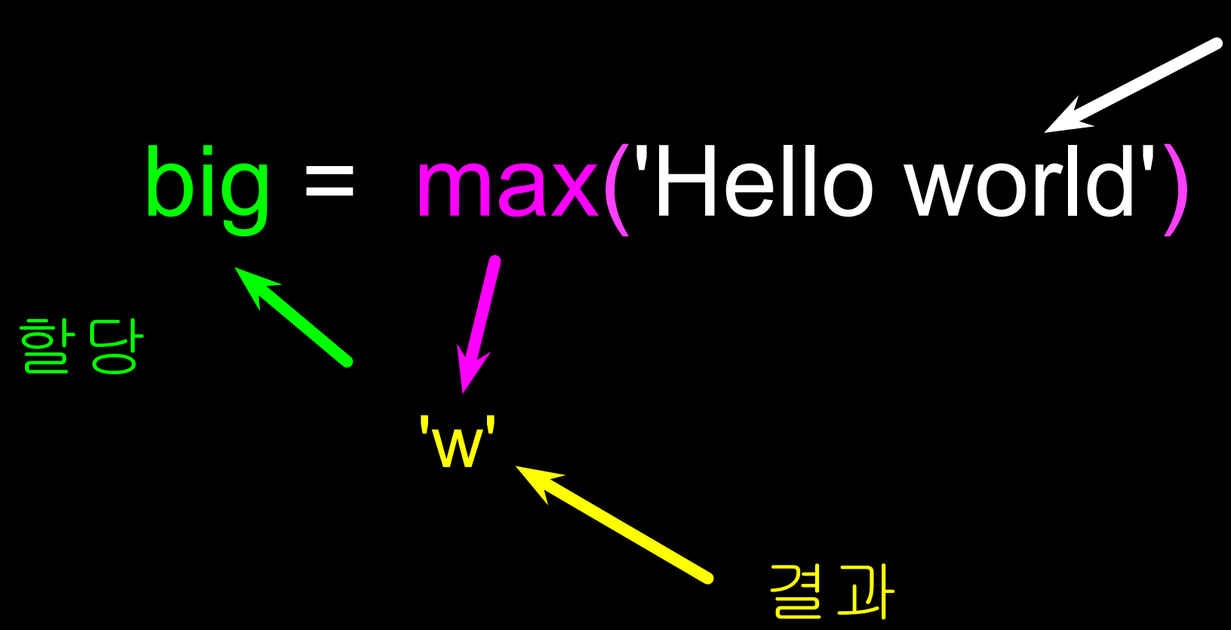
  - 내장 함수 파이썬의 한 부분으로 제공됨 - 예: print(), input(), type(), float(), int() ...

  - 우리가 정의하고 사용하는 함수

- 내장 함수의 이름은 “새로운” 예약어라고 생각
  (i.e., 변수명으로 사용할 수 없음)

함수의 정의

- 파이썬에서 함수는 인자를 입력 받고, 계산을 하고, 결과를 반환하는 재사용 가능한 코드
- 함수를 정의할 때 def 예약어를 이용
- 함수 이름, 괄호 그리고 인자를 이용해 함수를 호출함




```
>>> big = max('Hello world')
>>> print(big)
w
>>> tiny = min('Hello world')
>>> print(tiny)

>>>
```

Max 함수

```
>>> big = max('Hello world')
>>> print(big)
w
```

함수는 우리가 사용할
저장된 코드임

함수는 입력값을 받아서
출력값을 생산

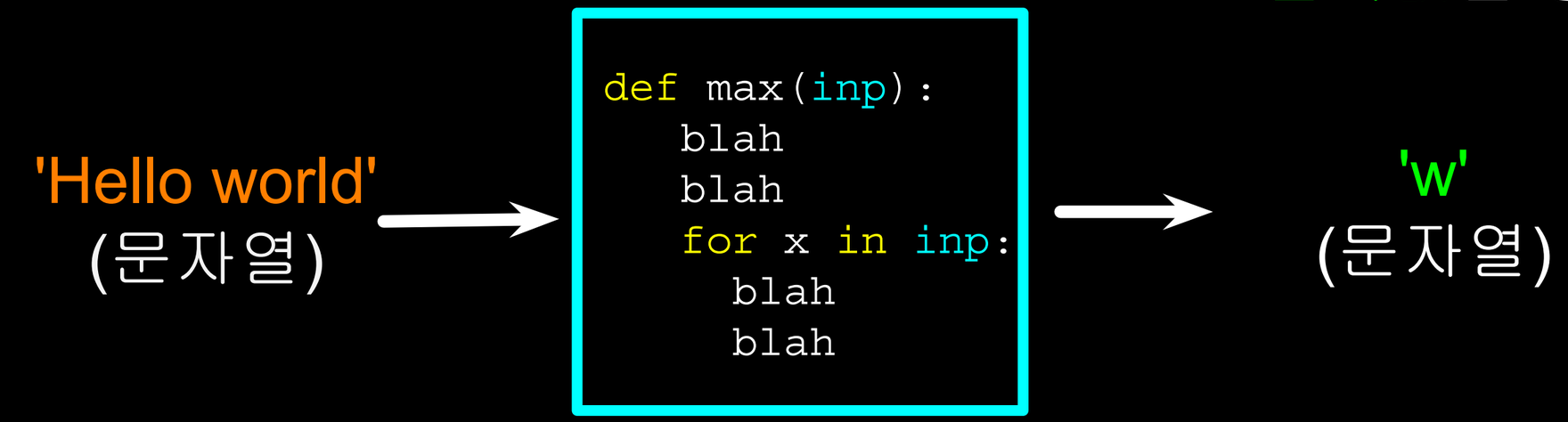
Guido가 작성한 코드

Max 함수

함수는 우리가 사용할
저장된 코드임

```
>>> big = max('Hello world')
>>> print(big)
w
```

함수는 입력값을 받아서
출력값을 생산



Guido가 작성한 코드

자료형 변환

- 정수형과 실수형을
  표현식에 동시에 사용할 때
  정수는 암시적으로
  실수형으로 변환됨

- 내장 함수인 int()과 float()을
  이용하여 조정 가능

```
>>> print(float(99) / 100)
0.99
>>> i = 42
>>> type(i)
<class 'int'>
>>> f = float(i)
>>> print(f)
42.0
>>> type(f)
<class 'float'>
>>> print(1 + 2 * float(3) / 4 - 5)
-2.5
>>>
```

문자열 변환

- int() 와 float()를 문자열에서 정수형으로 변환할 때도 사용할 수 있음
- 문자열이 숫자를 포함하지 않으면 에러

```
>>> sval = '123'
>>> type(sval)
<class 'str'>
>>> print(sval + 1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str'
and 'int'
>>> ival = int(sval)
>>> type(ival)
<class 'int'>
>>> print(ival + 1)
124
>>> nsv = 'hello bob'
>>> niv = int(nsv)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int()
```

직접 정의한 함수....

우리만의 함수 만들기

- def 키워드, 괄호, 그리고 선택적으로 매개 변수를 적어서 새로운 함수를 만들 수 있음
- 함수 본문은 들여쓰기를 함
- 이는 함수를 정의하지만 함수의 본문을 실행하지는 않음

```
def print_lyrics():
    print("I'm a lumberjack, and I'm okay.")
    print('I sleep all night and I work all day.')
```

print_lyrics():

```
print("I'm a lumberjack, and I'm okay.")
print('I sleep all night and I work all day.')
```

```
x = 5
print('Hello')

def print_lyrics():
    print("I'm a lumberjack, and I'm okay.")
    print('I sleep all night and I work all day.')

print('Yo')
x = x + 2
print(x)
```

```
Hello
Yo
7
```

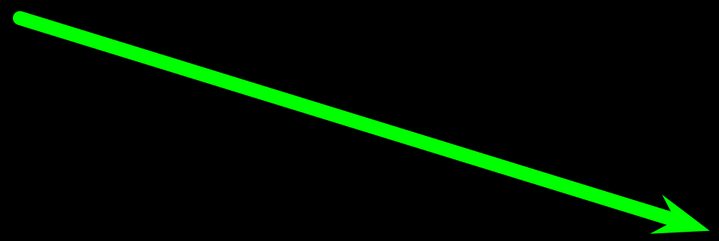
정의와 사용

- 함수를 한 번 정의하면, 원하는 만큼 호출 (또는 실행) 가능
- 이는 저장과 재사용 패턴

```
x = 5
print('Hello')

def print_lyrics():
    print("I'm a lumberjack, and I'm okay.")
    print('I sleep all night and I work all day.')

print('Yo')
print_lyrics()
x = x + 2
print(x)
```



Hello
Yo
I'm a lumberjack, and I'm okay.
I sleep all night and I work all day.
7

인자

- 인자는 함수를 호출 할 때 입력값으로 전달하는 값
- 인자는 함수가 다른 조건에서 호출 되었을 때 각각 다른 일을 수행할 수 있도록 지시하는 역할
- 인자는 함수 이름 다음의 괄호 안에 씀

```
big = max('Hello world')
```

Argument

매개 변수

매개 변수는 함수 정의에
사용되는 변수임. 특정 함수
호출에서 함수 안의 코드가
인자에 접근하기 위한 “손잡이”
역할.

```
>>> def greet(lang):
...     if lang == 'es':
...         print('Hola')
...     elif lang == 'fr':
...         print('Bonjour')
...     else:
...         print('Hello')
...
>>> greet('en')
Hello
>>> greet('es')
Hola
>>> greet('fr')
Bonjour
>>>
```

반환값

함수는 종종 인자를 받아서 계산을 하고 함수 호출 구문이 사용할 수 있도록 값을 반환. 이를 위해 return 키워드를 사용.

```
def greet():
    return "Hello"

print(greet(), "Glenn")
print(greet(), "Sally")
```

```
Hello Glenn
Hello Sally
```

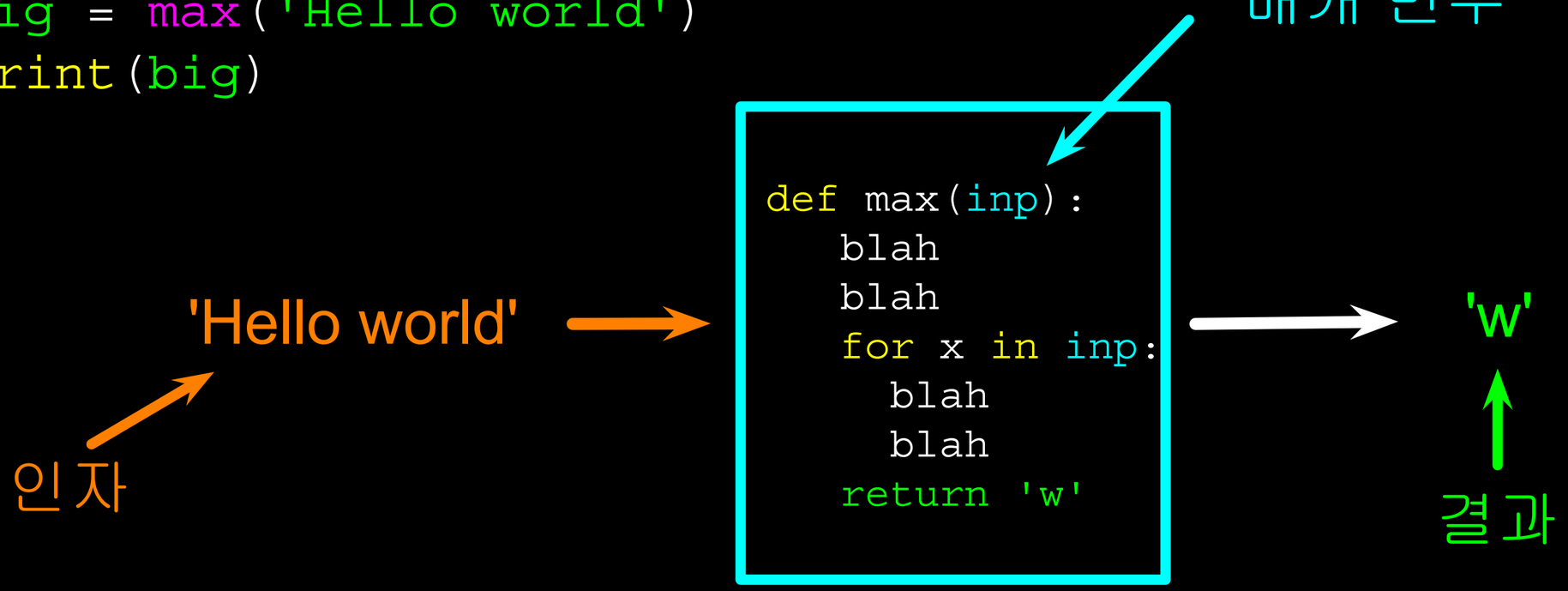
반환값

- “fruitful” 함수는 결과 (또는 반환값)을 생성
- return 구문은 함수 실행을 끝내고 함수의 결과를 “반환”

```
>>> def greet(lang):
...     if lang == 'es':
...         return 'Hola'
...     elif lang == 'fr':
...         return 'Bonjour'
...     else:
...         return 'Hello'
...
>>> print(greet('en'),'Glenn')
Hello Glenn
>>> print(greet('es'),'Sally')
Hola Sally
>>> print(greet('fr'),'Michael')
Bonjour Michael
>>>
```

인자, 매개 변수, 그리고 결과

```
>>> big = max('Hello world')
>>> print(big)
w
```



다중 매개 변수 / 인자

- 함수 정의에서 한 개 이상의 매개 변수를 정의할 수 있음
- 단순히 함수를 호출 할 때 인자를 추가
- 숫자는 인자의 순서에 따라 매개 변수와 매칭

```
def addtwo(a, b):
    added = a + b
    return added

x = addtwo(3, 5)
print(x)
```

```
8
```

Void (non-fruitful) 함수

- 함수가 값을 반환하지 않으면, “void” 함수라고 함
- 값을 반환하는 함수를 “fruitful” 함수라고 함
- Void 함수는 “not fruitful” 함수

함수를 쓰느냐 마느냐...

- 코드를 한 “문단”으로 정리 - 완전한 아이디어를 저장해두고 “명명”
- 번복하지 말기 - 한 번에 작동하도록 만들어서 재사용
- 코드가 너무 길어지거나 복잡해지면, 논리 상 나누어서 각 조각을 함수 안에 집어 넣어넣기
- 일반적으로 자주 사용 하는 것은 라이브러리화 - 동료와 공유해도 좋음

요약

- 함수
- 내장 함수
- 자료형 변환 (int, float)
- 문자열 변환
- 매개 변수
- 인자
- 결과 (fruitful 함수)
- Void (non-fruitful) 함수
- 함수를 사용하는 이유

예제

월급 계산 프로그램을 다시 프로그램하기.
근무 시간이 40시간이 넘으면 초과 근무 시간에는
시급의 1.5배를 지급합니다.이름이 computepay이고
2개의 매개 변수(시간과 시급)를 받는 함수를
작성하기.

Enter Hour: 45
Enter Rate: 10

Pay: 475.0

475 = 40 * 10 + 5 * 15

Acknowledgements / Contributions

Initial Development: Charles Severance, University of Michigan School of Information

Contributor:
- Seung-June Lee (plusjune@gmail.com)
- Connect Foundation

Translator:
- Hakyeong Kim
- Jeungmin Oh (tangza@gmail.com)

...