w11-Lec1

Functions Part III

for 204111

by Kittipitch Kuptavanich

Default Arguments Example

อยู่หลังเสมอ

```
>>> def f(x, y=10):
...     return (x, y)
...
>>> print(f(5))
(5, 10)
>>> print(f(5, 6))
(5, 6)
```

- **Non-default arguments ต้องมาก่อน Default arguments เสมอ**

อยู่หลังเสมอ Always!!

```
>>> def f(y=10, x):
...     return (x, y)
...
  File "<stdin>", line 1
SyntaxError: non-default argument follows default argument
```

Do Not Use Mutable Default Args

ไม่นิยมใช้

```python
def f(x, list_x=[]): #lists, sets, dicts, etc.
    list_x.append(x)
    return list_x


print(f(1))
print(f(2))   # why is this [1, 2]?
```

Think Python: How to Think Like a Computer Scientist

One Workaround for Mutable Default Args

```
def f(x, list_x=None):
    if (list_x == None):
        list_x = []
    list_x.append(x)
    return list_x

print(f(1))
print(f(2))   # [2] (that's better)
```

Functions as Parameters

```
def derivative(f, x):
    h = 10**-8
    return (f(x+h) - f(x))/h


def f(x):
    return 4*x + 3

print(derivative(f, 2))  # about 4

def g(x):
    return 4*x**2 + 3

print(derivative(g, 2))  # about 16 (8*x at x==2)
```

lambda Functions

```
print(derivative(lambda x: 3*x**5 + 2, 2))
# about 240, 15*x**4 at x==2

def my_func(x):
    return 10*x + 42

print(my_func(5))  # 92
print(derivative(my_func, 5))  # about 10

# instead of this
~~b = list(map(lambda x: str(x), [1, 2, 3]))~~
#  Do this
b = list(map(str, [1, 2, 3]))
```

เป็น Built in อยู่แล้ว

Misuse of lambda expressions

```
# The official python style guide PEP8, strongly
discourages the assignment of lambda expressions as shown
in the example below.

sum = lambda x, y: x + y
print(type(sum))      # <class 'function'>

x1 = sum(4, 7)
print(x1)                    # 11

# instead of this
func = lambda x, y, z: x*y + z

# it is recommended to write a one-liner function as
def func(x, y, z): return x*y + z

```

https://www.geeksforgeeks.org/overuse-of-lambda-expressions-in-python/

Variable length args (*args)

ใช้บอกการ call มากกว่า 1 ตัวแปร

```python
def longest_word(*args):
    if (len(args) == 0):
        return None
    result = args[0]
    for word in args:
        if (len(word) > len(result)):
            result = word
    return result

print(longest_word("this", "is", "really", "nice"))
# really

mywords = ["this", "is", "really", "nice"]
print(longest_word(mywords))
# ['this', 'is', 'really', 'nice’]

print(longest_word(*mywords))  # really
```

https://www.cs.cmu.edu/~112/notes/notes-functions-redux.html

Keyword Arguments

มี Default >1

```python
def f(x=1, y=2):
    return (x, y)

print(f())  # (1, 2)
print(f(3))  # (3, 2)
```

มีตัวเดียวจะแทนในตัวแรก

```python
# [here is where we use a keyword arg]
print(f(y=3))  # (1, 3)
```

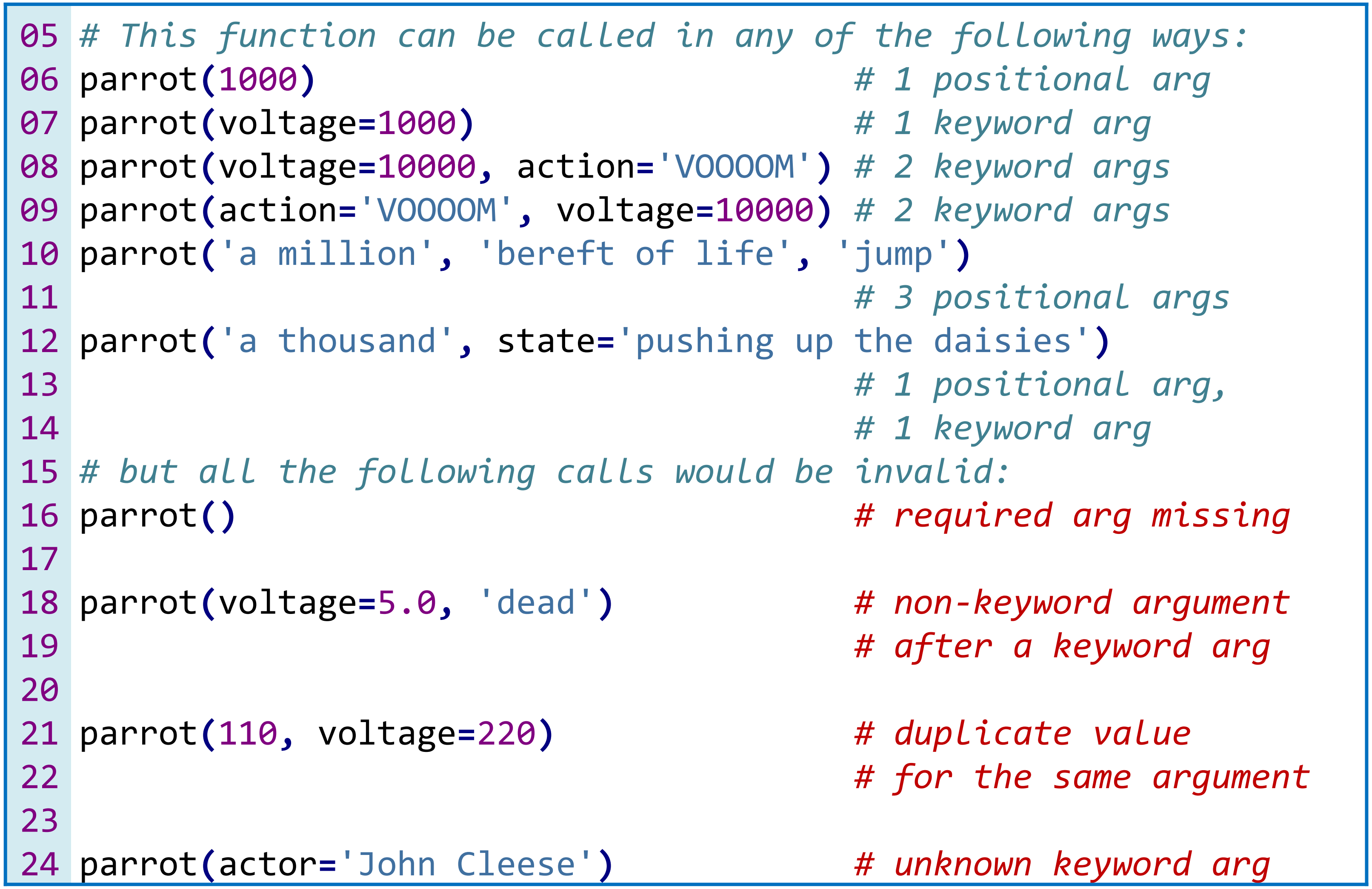
ไม่เปลี่ยน x

```python
def parrot(voltage, state='a stiff', action='voom',
           type='Norwegian Blue'):
    print("-- This parrot wouldn't", action, end=' ')
    print("if you put", voltage, "volts through it.")
    print("-- Lovely plumage, the", type)
    print("-- It's", state, "!")

# accepts one required argument (voltage)
# and three optional arguments (state, action, and type).
```

https://www.cs.cmu.edu/~112/notes/notes-functions-redux.html

Keyword Arguments [2]



```
# This function can be called in any of the following ways:
parrot(1000)                           # 1 positional arg
parrot(voltage=1000)                   # 1 keyword arg
parrot(voltage=10000, action='VOOOOM') # 2 keyword args
parrot(action='VOOOOM', voltage=10000) # 2 keyword args
parrot('a million', 'bereft of life', 'jump')
                                       # 3 positional args
parrot('a thousand', state='pushing up the daisies')
                                       # 1 positional arg,
                                       # 1 keyword arg
# but all the following calls would be invalid:
parrot()                               # required arg missing

parrot(voltage=5.0, 'dead')            # non-keyword argument
                                       # after a keyword arg

parrot(110, voltage=220)               # duplicate value
                                       # for the same argument

parrot(actor='John Cleese')            # unknown keyword arg
```

https://www.cs.cmu.edu/~112/notes/notes-functions-redux.html

References

- https://www.cs.cmu.edu/~112/notes/notes-functions-redux.html
- https://www.geeksforgeeks.org/overuse-of-lambda-expressions-in-python/
- https://docs.python.org/3/tutorial/controlflow.html