

- description :
- author :
- email : hylee@repia.com
- lastupdate : 2020-06-25

```
# Section06
#                               (lambda)

#
# def function_name(parameter):
#     code

#
# function_name()
#

#     1
print("#====      ====")
print("#===      ===")
def hello(world):
    print("Hello, ", world)

param1 = "Niceman"

hello(param1)

print()

#     2
print("#====      ====")
def hello_return(world):
    value = "Hello, " + str(world)
    return value

str = hello_return("Niceman")

print(str)
```

```
print()

#      3(      )
print("#===      ===")

def func_mul1(x):
    y1 = x * 2
    y2 = x * 4
    y3 = x * 6
    return y1, y2, y3

val1, val2, val3 = func_mul1(3)

print(val1, val2, val3)

print()

#
print("#===      ===")
def func_mul2(x):
    y1 = x * 2
    y2 = x * 4
    y3 = x * 6
    return (y1, y2, y3)

tup = func_mul2(4)

print(type(tup), tup, list(tup))

print()

#
print("#===      ===")
def func_mul2(x):
    y1 = x * 2
    y2 = x * 4
    y3 = x * 6
    return [y1, y2, y3]

lis = func_mul2(6)

print(type(lis), lis, set(lis))

print()
```

```

#
print("#===          ===")
def func_mul3(x):
    y1 = x * 2
    y2 = x * 4
    y3 = x * 6
    return {'ret1': y1, 'ret2': y2, 'ret3': y3}

#
# return [y1, y2, y3]

#
# return (y1, y2, y3)

dic = func_mul3(8)
print(type(dic), dic, dic.get('ret3'), dic.items(), dic.keys(),
dic.values())

print()

#      4
# *args, **kwargs
print("#=== *args, **kwargs      ===")

# *args
#
#      가
print("#== *args      ==")
def args_func(*args):
    for i, v in enumerate(args): # enumerate -> index
        print('{}'.format(i), v, end=' ')

args_func('Kim')
args_func('Kim', 'Park')
args_func('Kim', 'Park', 'Lee')

print('\n')

# kwargs
print("#=== **kwargs      ==")
def kwargs_func(**kwargs): #
    for v in kwargs.keys(): #      가
        print('{}'.format(v), kwargs[v], end=' ')

kwargs_func(name1='Kim')
kwargs_func(name1='Kim', name2='Park')
kwargs_func(name1='Kim', name2='Park', name3='Lee')

```

```
print('\n')

#
print("#==      ==")
def example(arg_1, arg_2, *args, **kwargs):
    print(arg_1, arg_2, args, kwargs)

example(10, 20)
example(10, 20, 'park', 'kim', 'lee')
example(10, 20, 'park', 'kim', 'lee', age1=33, age2=34, age3=44)

print()

#      5
#      (      )
print("#===      (      ) ===")
def nested_func(num):
    def func_in_func(num):
        print(num)

    print("In func")
    func_in_func(num + 100)

nested_func(1)

print()

#      가
#      func_in_func(1)

#      6
#      Hint
print("#=== Hint      ===")
def tot_length1(word: str, num: int) -> int:
    return len(word) * num

# word: str, num: int
#      : Hint      : Hint

print('hint exam1 : ', tot_length1("i love you", 10))

def tot_length2(word: str, num: int) -> None:
    print('hint exam2 : ', len(word) * num)
```

```

tot_length2("niceman", 10)

print('\n\n')

print("#====          =====")

#
#           ,가          ,
#           ->          (      )
#           (Heap      ) ->

#       7
# def mul_10(num):
#     return num * 10

# def mul_10_one(num): return num * 10
#
# lambda x: x * 10

#           ->
print("#===          ->          =====")
def mul_10(num):
    return num * 10

mul_func = mul_10

print(mul_func(5))
print(mul_func(6))

print()
#           ->

print("#===          ->          =====")
lambda_mul_func = lambda x: x * 10

def func_final(x, y, func):
    print(x * y * func(10))

func_final(10, 10, lambda_mul_func)

```

```

#====          =====
#===          ===
Hello, Niceman

#===          ===

```

Hello, Niceman

```
# ===      ===  
6 12 18
```

```
# ===      ===  
<class 'tuple'> (8, 16, 24) [8, 16, 24]
```

```
# ===      ===  
<class 'list'> [12, 24, 36] {24, 12, 36}
```

```
# ===      ===  
<class 'dict'> {'ret1': 16, 'ret2': 32, 'ret3': 48} 48 dict_items([('ret1', 16), ('ret2', 32), ('ret3', 48)]) dict_keys(['ret1', 'ret2', 'ret3']) dict_values([16, 32, 48])
```

```
#=== *args, **kwargs      ===  
#== *args      ==  
0 Kim 0 Kim 1 Park 0 Kim 1 Park 2 Lee
```

```
#== **kwargs      ==  
name1 Kim name1 Kim name2 Park name1 Kim name2 Park name3 Lee
```

```
# ==      ==  
10 20 () {}  
10 20 ('park', 'kim', 'lee') {}  
10 20 ('park', 'kim', 'lee') {'age1': 33, 'age2': 34, 'age3': 44}
```

```
# ===      (      ) ===  
In func  
101
```

```
#=== Hint      ===  
hint exam1 : 100  
hint exam2 : 70
```

```
# =====      =====  
# ===      ->      ===  
50  
60
```


```
# ===      ->      ===  
10000
```

Tip

, [python](#), , [lambda](#)

From:
<http://rwiki.repia.com/> -

. - 2023.12

Permanent link:
http://rwiki.repia.com/doku.php?id=wiki:ai:python:%ED%95%A8%EC%88%98_%EB%B0%8F_%EB%9E%8C%EB%8B%A4&rev=1593063888 

Last update: **2022/03/10 19:52**