



Định nghĩa

python

- ❑ Function (phương thức) là tập hợp các dòng code được viết để thực hiện một chức năng nào đó.
- ❑ Mục đích xây dựng function là để có thể tái sử dụng khi cần.
- ❑ Function cung cấp module tốt hơn cho ứng dụng và mức độ tái sử dụng code cao hơn.
- ❑ Python cung cấp rất nhiều function xây dựng sẵn (*built-in function*), và người dùng cũng có thể tự xây dựng function (*user-defined function*)



Fundamentals of Python - Lập trình Python cơ bản

3



Nội dung

python

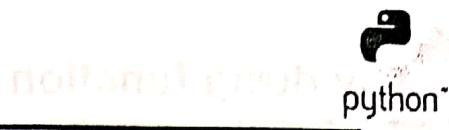
1. Định nghĩa
2. Xây dựng function
3. Gọi function
4. Biến cục bộ và biến toàn cục
5. Tham số (parameter/ argument)
6. Anonymous Function (lambda)
7. Built-in Function



Fundamentals of Python - Lập trình Python cơ bản

4

Xây dựng function



❑ Cú pháp:

```
def function_name(parameters):
    # mô tả về function
    các dòng lệnh
    return [expression]
```

Fundamentals of Python - Lập trình Python cơ bản

5

Xây dựng function



- Ví dụ: Viết function tính điểm trung bình của hk1 và hk2, in kết quả trực tiếp trong function.

Công thức tính: $dtb = (hk1 + hk2 * 2) / 3$

```
def tinh_diem_trung_binh(hk1, hk2):
    dtb = (hk1 + hk2 * 2) / 3
    print("Điểm trung bình:", dtb)
    return
```

Fundamentals of Python - Lập trình Python cơ bản

6

Xây dựng function



- Ví dụ: Viết function tính và trả về chỉ số BMI của một người dựa trên cân nặng và chiều cao được cung cấp. Công thức tính BMI:

$$\text{BMI} = \text{cân nặng} / (\text{chiều cao})^2$$

```
import math
def tinh_bmi(can_nang, chieu_cao):
    bmi = can_nang / math.pow(chieu_cao, 2)
    return bmi
```



Nội dung



1. Định nghĩa
2. Xây dựng function
3. Gọi function
4. Biến cục bộ và biến toàn cục
5. Tham số (parameter/ argument)
6. Anonymous Function (lambda)
7. Built-in Function



Gọi function



Cú pháp:

function_name(values)

Ví dụ:

```
hk1 = eval(input("Nhập điểm HK1: "))
hk2 = eval(input("Nhập điểm HK2: "))
tinh_diem_trung_binh(hk1, hk2)
```

```
Nhập điểm HK1: 7.5
Nhập điểm HK2: 8.5
Điểm trung bình: 8.166666666666666
```

```
bmi = tinh_bmi(52, 1.6)
print("BMI:", bmi)
```

```
BMI: 20.312499999999996
```

Fundamentals of Python - Lập trình Python cơ bản

9

Nội dung



1. Định nghĩa
2. Xây dựng function
3. Gọi function
4. Biến cục bộ và biến toàn cục
5. Tham số (parameter/ argument)
6. Anonymous Function (lambda)
7. Built-in Function

Fundamentals of Python - Lập trình Python cơ bản

10

Biến cục bộ và biến toàn cục



❑ Biến cục bộ (local variable) có thể được truy cập chỉ trong function mà chúng được khai báo

❑ Biến toàn cục (global variable) có thể được truy cập trên toàn chương trình của tất cả các function

Biến cục bộ và biến toàn cục



❑ Ví dụ

• Biến toàn cục

```
s = "Hello Python"  
def in_s():  
    print(s)  
    return  
  
print(s)  
Hello Python  
  
in_s()  
Hello Python
```

Biến cục bộ và biến toàn cục



❑ Ví dụ

• Biến cục bộ

```
s = "Hello Python"  
def in_s():  
    s = "Bonjour Python"  
    print(s)  
    return  
  
print(s)  
Hello Python  
in_s()  
Bonjour Python
```

Fundamentals of Python - Lập trình Python cơ bản

13

Nội dung



1. Định nghĩa

2. Xây dựng function

3. Gọi function

4. Biến cục bộ và biến toàn cục

5. Tham số (parameter/ argument)

6. Anonymous Function (lambda)

7. Built-in Function

Fundamentals of Python - Lập trình Python cơ bản

14



python

Tham số

□ Tham chiếu (reference) hay tham trị (value)?

- Các tham số (parameter/ argument) có kiểu tham trị trong function: nếu ta thay đổi giá trị của nó trong function thì không ảnh hưởng gì đến giá trị của nó bên ngoài function.
- Các tham số (parameter/ argument) có kiểu tham chiếu trong function: nếu ta thay đổi giá trị của nó trong function thì cũng sẽ thay đổi giá trị của nó bên ngoài function.



Fundamentals of Python - Lập trình Python cơ bản

15



python

Tham số

• Ví dụ: Tham trị

```
def tinh_binh_phuong(so):
    so = so * so
    print("Số trong hàm =", so)
    return

so = 8
tinh_binh_phuong(so)
Số trong hàm = 64

print("Số ngoài hàm =", so)
Số ngoài hàm = 8
```



Fundamentals of Python - Lập trình Python cơ bản

16

Tham số

python

• Ví dụ: Tham chiếu

```
def change_list(lst):
    lst.append(10)
    lst.append(20)
    print("List trong hàm:", lst)
    return

lst = [1, 3, 7, 12]
print("List ban đầu:", lst)
List ban đầu: [1, 3, 7, 12]

change_list(lst)
List trong hàm: [1, 3, 7, 12, 10, 20]

print("List ngoài hàm:", lst)
List ngoài hàm: [1, 3, 7, 12, 10, 20]
```

Fundamentals of Python - Lập trình Python cơ bản

17

Tham số

python

□ Phân loại: có 4 loại

- Required argument (tham số bắt buộc)
- Keyword argument (tham số từ khóa)
- Default argument (tham số mặc định)
- Variable-length argument (tham số thay đổi không xác định)

Fundamentals of Python - Lập trình Python cơ bản

18

□ Required argument (tham số bắt buộc)

- Là tham số khi truyền vào function phải đúng vị trí.
- Ở đây số lượng các tham số khi gọi function cần phải trùng khớp chính xác với function đã được xây dựng.



Tham số

- Ví dụ: Ở function tinh_bmi có 2 tham số kiểu số, vì vậy khi gọi function cũng phải truyền vào 2 giá trị kiểu số theo đúng thứ tự định nghĩa.

```
import math
def tinh_bmi(can_nang, chieu_cao):
    bmi = can_nang / math.pow(chieu_cao, 2)
    return bmi
bmi = tinh_bmi(52, 1.6)
print("BMI:", bmi)
BMI: 20.312499999999996
tinh_bmi(50)
Traceback (most recent call last):
  File "C:\Users\LNTRI\eclipse-workspace\Python_CB\src\Demo\Function.py",
    tinh_bmi(50)
TypeError: tinh_bmi() missing 1 required positional argument: 'chieu_cao'
```



Tham số

python

❑ Keyword argument (tham số từ khóa)

- Là tham số liên quan đến lời gọi function.
- Khi ta sử dụng keyword argument thì lời gọi hàm sẽ thực hiện ưu tiên dựa trên tên tham số (không cần phải theo thứ tự).
- Tuy nhiên, số lượng tham số cũng phải chính xác.

Fundamentals of Python - Lập trình Python cơ bản

21

Tham số

python

- Ví dụ: Ở function tinh_bmi có 2 tham số kiểu số, vì vậy khi gọi function cũng phải truyền vào 2 giá trị kiểu số tuy nhiên không cần theo thứ tự mà theo tên tham số

```
import math
def tinh_bmi(can_nang, chieu_cao):
    bmi = can_nang / math.pow(chieu_cao, 2)
    return bmi

bmi = tinh_bmi(chieu_cao = 1.6, can_nang = 52)
print("BMI:", bmi)
BMI: 20.31249999999996

bmi = tinh_bmi(cao = 1.6, nang = 52)
print("BMI:", bmi)
Traceback (most recent call last):
  File "C:\Users\LNTRI\eclipse-workspace\Python C8\src\Demo\Func
    bmi = tinh_bmi(cao = 1.6, nang = 52)
TypeError: tinh_bmi() got an unexpected keyword argument 'cao'
```

Fundamentals of Python - Lập trình Python cơ bản

22

Tham số

python

❑ Default argument (tham số mặc định)

- Là tham số sẽ được function tự động lấy giá trị mặc định để thực hiện nếu khi gọi function người dùng không cung cấp giá trị cho nó.



Tham số

python

- Ví dụ: với function choose_drink, nếu người dùng không truyền vào drink thì mặc định sẽ là 'coffee'

```
def choose_drink(price, drink = "coffee"):
    print("With", price, "vnđ, you can buy", drink)
    return

choose_drink(10000, "tea")
With 10000 vnđ, you can buy tea

choose_drink(10000)
With 10000 vnđ, you can buy coffee
```



Tham số



□ Variable-length argument (tham số có chiều dài thay đổi không xác định)

- Là tham số được sử dụng khi chưa xác định được số các giá trị truyền vào function
- Sử dụng dấu * khi khai báo tham số thay đổi: *parameter_name
- Chỉ có một tham số loại này trong function
- Tham số thay loại này phải đứng sau cùng



Tham số



- Ví dụ: xây dựng function hiển thị lời chào kèm danh sách tên được chào

```
def in_loi_chao(loi_chao, *danh_sach_ten):
    for ten in danh_sach_ten:
        print(loi_chao, ten)
return
in_loi_chao("Hello")
in_loi_chao("Hello", "Lan", "Mai", "Trúc", "Đào")
Hello Lan
Hello Mai
Hello Trúc
Hello Đào
in_loi_chao("Chào mừng", "Nam", "An")
Chào mừng Nam
Chào mừng An
```



Nội dung



python

1. Định nghĩa
2. Xây dựng function
3. Gọi function
4. Biến cục bộ và biến toàn cục
5. Tham số (parameter/ argument)
6. Anonymous Function (lambda)
7. Built-in Function



Fundamentals of Python - Lập trình Python cơ bản

27

Anonymous Function (lambda)

python

Định nghĩa

- Anonymous Function (Phương thức ẩn danh): gọi là ẩn danh vì nó không được khai báo theo cách tiêu chuẩn bằng từ khóa def mà được viết ngắn gọn bằng cách sử dụng từ khóa lambda để tạo ra các phương thức ngắn (1 dòng lệnh).

Cú pháp:

lambda [parameter1[, parameter2,...]]:
expression



Fundamentals of Python - Lập trình Python cơ bản

28

Anonymous Function (lambda)

python

● Chú ý:

- Lambda có thể có một hoặc nhiều tham số truyền vào nhưng chỉ trả về một giá trị duy nhất
- Lambda không thể chứa nhiều command hoặc expression.
- Lambda không thể được gọi trực tiếp để in kết quả bởi vì nó yêu cầu một expression.
- Lambda có local namespace của nó và không thể truy cập các biến khác và biến trong phạm vi global namespace.

bang với sequence ngắn

Fundamentals of Python - Lập trình Python cơ bản

29

Anonymous Function (lambda)

python

● Ví dụ:

```
import math
s = lambda x, n: math.pow(math.pow(x, 2) + 1, n)
print("s =", s(2, 3))
s = 125.0
print("s =", s(3, 3))
s = 1000.0
```

Fundamentals of Python - Lập trình Python cơ bản

30



Nội dung

1. Định nghĩa
2. Xây dựng function
3. Gọi function
4. Biến cục bộ và biến toàn cục
5. Tham số (parameter/ argument)
6. Anonymous Function (lambda)
7. Built-in Function



Built-in Function

map(), reduce(), filter() là những built-in function hỗ trợ việc xử lý các sequence rất hiệu quả



Built-in Function

❑ map()

- Tạo ra một sequence mới dựa trên một phương thức và sequence cũ, mỗi phần tử trong sequence cũ sẽ áp dụng phương thức để thành phần tử mới.
- Nếu có nhiều hơn một sequence được cung cấp thì phương thức sẽ được gọi kết hợp cho từng phần tử của các sequence
- Nếu một sequence ngắn hơn so với sequence khác => kết quả sẽ có số phần tử bằng với sequence ngắn

Built-in Function

❑ map()

- Cú pháp: map(function , sequence , [sequence...]) → list
- Ví dụ:

```
list_one = [1, 2, 3, 4, 5]
list_three = [6, 7, 8, 9, 10]
print('list_one:', list_one)
list_two = list(map(lambda item: item ** 2, list_one))
print('list_second:', list_two)
list_four = list(map(lambda item1, item2: item1 + item2, list_one, list_three))
print('list_four:', list_four)
```

```
list_one: [1, 2, 3, 4, 5]
list_second: [1, 4, 9, 16, 25]
list_four: [7, 9, 11, 13, 15]
```

Built-in Function

- Ví dụ:

```
from operator import add
list_new = list(map(lambda item1, item2: item1 + item2, [1,2,3,4], (2,3,4,5)))
print('list_new:', list_new)

list_new_1 = list(map(lambda item1, item2: item1 - item2, [1,2,3], [2,3,4,5,6]))
print('list_new_1:', list_new_1)

print("List_add:", list(map(add, [1,2,3,4,5], [6,7,8,9,10])))

list_new: [3, 5, 7, 9]
list_new_1: [-1, -1, -1]
list_add: [7, 9, 11, 13, 15]
```

Fundamentals of Python - Lập trình Python cơ bản

35

Built-in Function

□ filter()

- Dùng để lọc các item trong sequence, tạo ra một sequence mới với các item thỏa điều kiện của function

- Cú pháp:

filter(function, sequence) -> list

Fundamentals of Python - Lập trình Python cơ bản

36

Scanned with CamScanner

Built-in Function

- Ví dụ:

```
list_number= [1, 2, 3, 4, 6, 7, 8, 9, 10]
list_even_number = list(filter(lambda item: item % 2==0, list_number))
print('list_even_number', list_even_number)
list_even_number [2, 4, 6, 8, 10]

list_string= ["abc", "def", "abf", "stu", "cba"]
list_string_a = list(filter(lambda item: 'a' in item, list_string))
print('list_string_a', list_string_a)
list_string_a ['abc', 'abf', 'cba']

print('list_filter', list(filter(lambda item: item=='a', "hello abababa")))
list_filter ['a', 'a', 'a', 'a']
```



Built-in Function

- **reduce()**

- Trả về kết quả là một giá trị đơn bằng cách áp dụng phương thức cho các item trong sequence

- Cú pháp:

```
reduce(function, sequence) -> value
```



Built-in Function

● Ví dụ:

```
from functools import reduce
from operator import add
list_number = [1, 2, 3, 4, 6, 7, 8, 9, 10]
sum1 = reduce(lambda item1, item2: item1 + item2, list_number)
print('Tổng:', sum1)
Tổng: 50

sum2 = reduce(add, list_number)
print('Tổng:', sum2)
Tổng: 50

list_words = ['He ', 'has ', 'a ', 'cat.']
print("Sentence:", reduce(lambda item1, item2 : item1 + item2, list_words))
Sentence: He has a cat.
```