Kỹ thuật lập trình:
HÀM VÀ MỘT SỐ KỸ THUẬT HÀM

NỘI DUNG

1. Khái quát về hàm trong Python
2. Hàm đệ quy
3. Một số hàm toán học thông dụng

1. Khái quát về hàm trong Python

Khái niệm

Hàm là ***một khối lệnh thực hiện một công việc hoàn chỉnh*** (module), ***được đặt tên*** và được gọi thực thi nhiều lần tại nhiều vị trí trong chương trình.

Hàm còn gọi là chương trình con (Subroutine)

Phân loại hàm:

- ✓ *Hàm thư viện: phải import thư viện trước khi sử dụng hàm (from … import)*
- ✓ *Hàm do người dùng tự định nghĩa.*

1. Khái quát về hàm trong Python

Ví dụ hàm thư viện, hàm tự định nghĩa

Vd: Hàm thư viện

```
from decimal import *
x = 6.6789
y = Decimal(x)
print(type(x))
print(type(y))
```

Vd: Hàm người dùng định nghĩa:

```
def Tong(x, y):
    return x + y

a = 8
b = 9
c = Tong(a, b)
print(f"Tong = {c}")
```

1. Khái quát về hàm trong Python

Cú pháp hàm

```
def name ( parameter list ) :
    block
```

Hàm được định nghĩa với từ khóa def, *hàm có thể có đối số hoặc không, có thể có kết quả trả về hoặc không.*

1. Khái quát về hàm trong Python

Ví dụ định nghĩa hàm

Vd: Hàm in lời chào

```
def sayWelcome():
    print("Welcome To UEL")
```

Vd: Hàm kiểm tra một số có phải là số chẵn

```
def isEven(x):
    if x % 2 == 0:
        return True
    return False
```

```
def phepChia(sobichia, sochia):
    return sobichia / sochia
```

1. Khái quát về hàm trong Python

Cách gọi hàm

Khi gọi hàm cần khai báo đầy đủ các đối số nếu có.

Gọi hàm **không** có kết quả trả về:

FunctionName([parameter]) → ***sayWelcome****()*

Gọi hàm **có** kết quả trả về:

result = FunctionName([parameter])

→ *b =* ***isEven****(x)*

→ *x =* ***phepChia****(9, 2)*

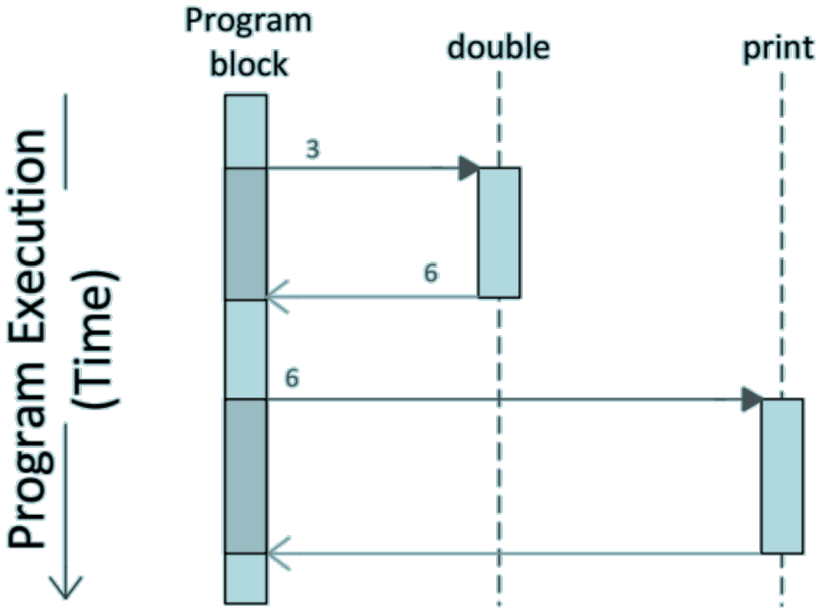
→ *hoặc* ***phepChia****(sochia=2, sobichia=9)*

1. Khái quát về hàm trong Python

Cơ chế hoạt động của hàm

Hàm trong Python cũng như trong các ngôn ngữ lập trình khác, đều hoạt động theo cơ chế LIFO (LAST IN FIRST OUT).

```
def double(n):
    return 2 * n
x = double(3)
print(x)
```



1. Khái quát về hàm trong Python

Tạo tài liệu cho hàm

func_example.py

```
def USLN(x, y):
    """
    Hàm tìm ước số chung lớn nhất của 2 số x, y
    """
    us = 1
    sobe = x if x < y else y
    for i in range(1, sobe + 1):
        if x % i == 0 and y % i == 0:
            us = i
    return us
```

1. Khái quát về hàm trong Python

Tạo tài liệu cho hàm

func_example.py

```
>>> from func_example import *
>>> help(USLN)
Help on function USLN in module func_example:

USLN(x, y)
    Hàm tìm ước số chung lớn nhất của 2 số x, y
```

1. Khái quát về hàm trong Python

Biến toàn cục (global variable)

Tất cả các biến khai báo bên trong hàm chỉ có phạm vi ảnh hưởng trong hàm, các biến này gọi là biến ***local***. Khi thoát khỏi hàm thì các biến này không thể truy xuất được.

```
x = 8
def increment():
    x = 6
    x = x + 1
increment()
print(x)
```

?

```
x = 8
def increment():
    global x
    x = 6
    x = x + 1
increment()
print(x)
```

?

```
x = 8
def increment():
    x = x + 1
increment()
print(x)
```

?

1. Khái quát về hàm trong Python

Tham số (parameter) mặc định

Ví dụ:

```
def print(self, *args, sep=' ', end='\n', file=None): # known special case of print
    """
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
    """
    pass
```

1. Khái quát về hàm trong Python

Tham số (parameter) mặc định

Ví dụ:

```
def lastItem(m, n=0):
    last = 0
    for i in range(1, m+n, 1):
        last = i
    return last

x = lastItem(8)
print("x = {0}".format(x)) ?    x = 7
y = lastItem(8, 2)
print(f"y = {y}") ?      y = 9
```

1. Khái quát về hàm trong Python

Lambda expression

Biểu thức xử lý

lambda *parameter list* : *expression*

```
def handle(f, x):
    return f(x)
a = handle(lambda x: x % 2 == 0, 9)
b = handle(lambda x: x % 2 == 0, 8)
print(f"a = {a}, b = {b}")
```

f là một function
x là một tham số

? False, True

```
def handle(f, x, y):
    return f(x, y)
tong = handle(lambda x, y: x + y, 3, 5)
print(f"Tong = {tong}")
```

?

1. Khái quát về hàm trong Python

Lambda expression

lambda *parameter list* : *expression*

```
def handle(f, x):
    return f(x)
def isEven(x):
    return x % 2 == 0
def isOdd(x):
    return x % 2 != 0

r1 = handle(isOdd, 5)
r2 = handle(lambda x: isOdd(x), 5)
r3 = handle(lambda y: isEven(y), 9)
print(f"R = {r1}, {r2}, {r3}")   ?
```

2. Hàm đệ quy

Khái niệm

Đệ quy là cách thức định nghĩa hàm gọi lại chính nó. Định nghĩa hàm đệ quy cần lưu ý:

1. *Quy luật giải quyết bài toán*
2. *Điều kiện dừng*

Phân tích kỹ thuật đệ quy và vòng lặp?

2. Hàm đệ quy

Ví dụ:

- Tính n giai thừa (với n là số nguyên): **n! = n * (n-1)!**

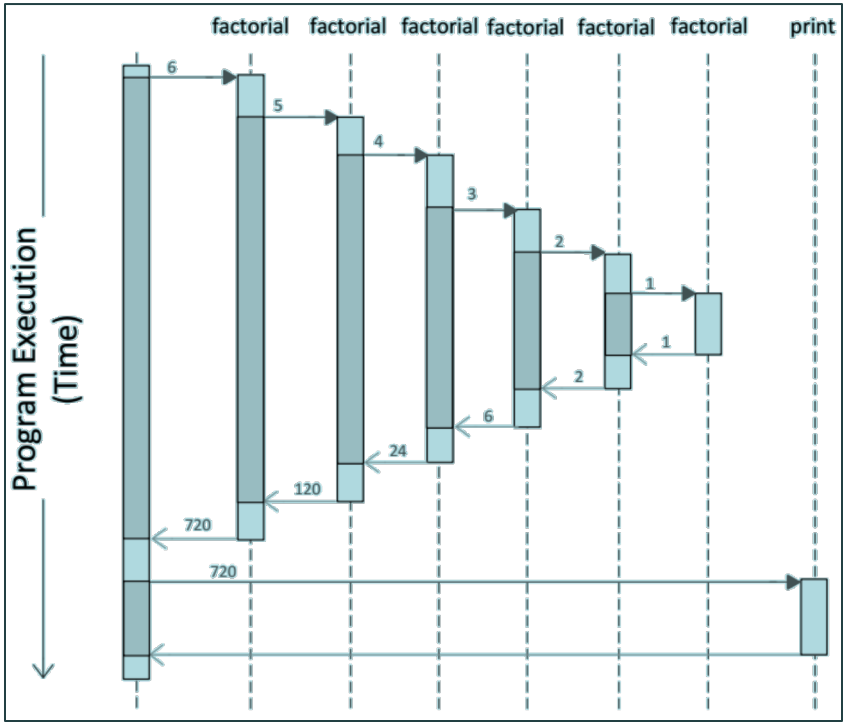
$$n! = \begin{cases} 1, \text{if } n = 0 \\ n * (n-1)! \end{cases}$$

Điều kiện dừng

Quy luật

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)

result = factorial(6)
print(result)
```




```
def factorial(n):
    return 1 if n == 0 else n * factorial(n - 1)
```

2. Hàm đệ quy

Bài tập:

- Tính số hạng thứ n của dãy Fibonacci:

$$F(1) = 1, F(2) = 1, F(n) = F(n-1) + F(n-2)$$

```
def fibonacci(n):
    if n == 1 or n == 2:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```

```
def fibonacci(n):
    return 1 if n == 1 or n == 2 else fibonacci(n-1) + fibonacci(n-2)
```

3. Một số hàm toán học thông dụng

Các hàm cơ bản

- sqrt → căn bậc 2
- pow → lũy thừa
- log → log(x) = $\log_e x$ = lnx
- log10 → $\log_{10} x$
- exp → e^x
- degrees → đổi radian sang độ
- radians → tính radian 180/pi*x
- fabs → trị tuyệt đối
- round → làm tròn số
- randrange(start, stop, [step]) → sinh số ngẫu nhiên ∈ [start, stop)
- randint(a, b) → sinh số ngẫu nhiên ∈ [a, b]

3. Một số hàm toán học thông dụng

Các hàm cơ bản

```
from math import *
print('sqrt(9) = ', sqrt(9))
print('pow(2,3) = ', pow(2, 3))
print('log(4) = ', log(4))
print('log10(100) = ', log10(100))
print('exp(3) = ', exp(3))
print('degrees(0.5235) = ', degrees(0.5235))
print('radians(60) = ', radians(60))
print('fabs(-6) = ', fabs(-6))
```

```
sqrt(9) =  3.0
pow(2,3) =  8.0
log(4) =  1.3862943611198906
log10(100) =  2.0
exp(3) =  20.085536923187668
degrees(0.5235) =  29.994340575098594
radians(60) =  1.0471975511965976
fabs(-6) =  6.0
```

```
print(round(10/3, 2)) → 3.33
```

```
from random import *
print(randrange(1, 10, 2)) → 1 or 3 or 5 or 7 or 9
```

3. Một số hàm toán học thông dụng

Hàm eval

```
from math import *
x = eval('3*sqrt(16)+pow(2,3)')
print(f'x = {x}')
```

```
def func(f, a, b):
    return f(a, b)
x = eval('func(lambda a, b: a if a > b else b, 6, 8)')
print('x = {0}'.format(x))
```

Q & A