Python cho Khoa học dữ liệu

Bài 5: Luồng và cấu trúc điều khiển

Hà Minh Tuấn

Khoa Toán - Tin học
Trường Đại học Khoa học Tự nhiên, ĐHQG-HCM

Ngày 28 tháng 8 năm 2025

Nội dung bài học

Điều khiển luồng trong Python

- Mặc định chương trình Python chạy tuần tự từ trên xuống.
- Để có khả năng **ra quyết định** và **lặp lại**, Python cung cấp:
  - Câu lệnh điều kiện (if, elif, else, match-case).
  - Câu lệnh lặp (for, while).
  - Câu lệnh nhảy (break, continue).

Câu lệnh if..elif..else

- Dùng để chọn nhánh lệnh tuỳ thuộc vào biểu thức Boolean.
- Cấu trúc:

```
if điều_kiện:
    khối lệnh
elif điều_kiện_khác:
    khối lệnh
else:
    khối lệnh
```

Ví dụ if..elif..else

```
marks = 80
result = ""

if marks < 30:
    result = "Failed"
elif marks > 75:
    result = "Passed with distinction"
else:
    result = "Passed"

print(result)
```

```
Passed with distinction
```

Câu lệnh match-case

- Hoạt động như switch-case trong các ngôn ngữ khác.
- So khớp mẫu (pattern) và thực thi lệnh tương ứng.
- Có thể dùng ký hiệu _ cho mặc định (default).

Ví dụ match-case

```
def checkVowel(n):
    match n:
        case 'a' | 'e' | 'i' | 'o' | 'u':
            return "Vowel alphabet"
        case _:
            return "Simple alphabet"

print(checkVowel('a'))
print(checkVowel('m'))
print(checkVowel('o'))
```

```
Vowel alphabet
Simple alphabet
Vowel alphabet
```

Vòng lặp trong Python

- **for loop**: lặp qua các phần tử của một chuỗi (list, tuple, string...).
- **while loop**: lặp lại khối lệnh khi điều kiện còn đúng.
- Tránh lặp vô hạn (infinite loop).

Ví dụ for loop

```
words = ["one", "two", "three"]
for x in words:
    print(x)
```

```
one
two
three
```

Ví dụ while loop

```
i = 1
while i < 6:
    print(i)
    i += 1
```

```
1
2
3
4
5
```

Câu lệnh nhảy (Jump statements)

- **break**: thoát khỏi vòng lặp ngay lập tức.
- **continue**: bỏ qua phần còn lại của vòng lặp hiện tại, sang vòng kế tiếp.

Ví dụ break

```
x = 0
while x < 10:
    print("x:", x)
    if x == 5:
        print("Breaking...")
        break
    x += 1
print("End")
```

```
x: 0
x: 1
x: 2
x: 3
x: 4
x: 5
Breaking...
End
```

Ví dụ continue

```
for letter in "Python":
    if letter == "h":
        continue
    print("Current Letter :", letter)
```

```
Current Letter : P
Current Letter : y
Current Letter : t
Current Letter : o
Current Letter : n
```

Câu lệnh rẽ nhánh trong Python

- Cú pháp chính: **if..elif..else**.
- `if` yêu cầu một biểu thức Boolean, sau dấu `:` là một khối lệnh thụt vào.
- Nếu biểu thức là `True` $\rightarrow$ thực thi khối lệnh.
- Nếu `False` $\rightarrow$ bỏ qua khối lệnh, thực thi lệnh ở mức thụt dòng trước đó.
- Python coi mọi giá trị khác 0/khác `None` là **True**, còn 0 hoặc `None` là **False**.

Các loại câu lệnh rẽ nhánh

1. **if**: chỉ chạy khi điều kiện đúng.
2. **if..else**: chọn giữa 2 nhánh.
3. **nested if**: lồng nhiều điều kiện trong nhau.

Single Statement Suites

- Nếu khối lệnh chỉ có một dòng, có thể viết cùng dòng với `if`.

```
var = 100
if (var == 100): print("Value of expression is 100")
print("Good bye!")
```

```
Value of expression is 100
Good bye!
```

Ví dụ if..else

```
var = 100
if (var == 100):
    print("Value of var is equal to 100")
else:
    print("Value of var is not equal to 100")
```

```
Value of var is equal to 100
```

Ví dụ Nested if

```
var = 100
if (var == 100):
    print("The number is equal to 100")
    if var % 2 == 0:
        print("The number is even")
    else:
        print("The number is odd")
elif var == 0:
    print("The given number is zero")
else:
    print("The given number is negative")
```

```
The number is equal to 100
The number is even
```

Câu lệnh if trong Python

- Câu lệnh `if` kiểm tra một điều kiện **True/False**.
- Nếu điều kiện **True** $\rightarrow$ khối lệnh bên trong `if` được thực thi.
- Nếu điều kiện **False** $\rightarrow$ khối lệnh `if` bị bỏ qua.

Cú pháp:

```
if expression:
    # các câu lệnh cần thực thi
```

Ví dụ: Giảm giá 10% nếu mua trên 1000

```
discount = 0
amount = 1200

# Kiểm tra giá trị amount
if amount > 1000:
    discount = amount * 10 / 100

print("amount =", amount - discount)
```

```
amount = 1080.0
```

Thay đổi giá trị amount

```
discount = 0
amount = 800

if amount > 1000:
    discount = amount * 10 / 100

print("amount =", amount - discount)
```

```
amount = 800
```

Câu lệnh if-else trong Python

- `if-else` dùng để chọn giữa 2 khối lệnh.
- Nếu điều kiện trong `if` là **True** $\rightarrow$ thực hiện khối `if`.
- Nếu điều kiện là **False** $\rightarrow$ thực hiện khối `else`.

Cú pháp:

```
if condition:
    # khối lệnh khi True
else:
    # khối lệnh khi False
```

Ví dụ: Kiểm tra độ tuổi đi bầu cử

```
age = 25
print("age:", age)
if age >= 18:
    print("eligible to vote")
else:
    print("not eligible to vote")
```

```
age: 25
eligible to vote
```

Thử giá trị khác cho age

```
age = 12
print("age:", age)
if age >= 18:
    print("eligible to vote")
else:
    print("not eligible to vote")
```

```
age: 12
not eligible to vote
```

Câu lệnh if-elif-else

- Cho phép kiểm tra nhiều điều kiện tuần tự.
- Nếu một điều kiện **True** $\rightarrow$ thực hiện khối lệnh tương ứng, bỏ qua các khối còn lại.
- Chỉ có **một khối else** cuối cùng (tùy chọn).

Cú pháp:

```
if condition1:
    # khối lệnh 1
elif condition2:
    # khối lệnh 2
elif condition3:
    # khối lệnh 3
else:
    # khối lệnh mặc định
```

Ví dụ: Tính giảm giá với if-elif-else

```
amount = 2500
print("Amount =", amount)

if amount > 10000:
    discount = amount * 20 / 100
elif amount > 5000:
    discount = amount * 10 / 100
elif amount > 1000:
    discount = amount * 5 / 100
else:
    discount = 0

print("Payable amount =", amount - discount)
```

```
Amount = 2500
Payable amount = 2375.0
```

Thử nhiều giá trị amount

```
for amount in [800, 2500, 7500, 15000]:
    if amount > 10000:
        discount = amount * 20 / 100
    elif amount > 5000:
        discount = amount * 10 / 100
    elif amount > 1000:
        discount = amount * 5 / 100
    else:
        discount = 0
    print("Amount:", amount,
          "Payable amount =", amount - discount)
```

```
Amount: 800 Payable amount = 800
Amount: 2500 Payable amount = 2375.0
Amount: 7500 Payable amount = 6750.0
Amount: 15000 Payable amount = 12000.0
```

Câu lệnh if lồng nhau (Nested if)

- Python hỗ trợ **nested if**, tức là đặt một câu lệnh `if` bên trong một `if` khác.
- Dùng khi cần kiểm tra thêm điều kiện sau khi điều kiện đầu tiên đúng.
- Có thể lồng `if`, `if...else` hoặc `if...elif...else` bên trong nhau.

Cú pháp:

```
if condition1:
    # khối lệnh 1
    if condition2:
        # khối lệnh 2
```

Ví dụ Nested if đơn giản

```
num = 36
print("num =", num)

if num % 2 == 0:
    if num % 3 == 0:
        print("Divisible by 3 and 2")

print("....execution ends....")
```

```
num = 36
Divisible by 3 and 2
....execution ends....
```

Nested if với else

- Có thể thêm `else` trong cấu trúc nested if.
- Nếu điều kiện `if` đầu tiên sai → chuyển sang `else`.
- Bên trong mỗi khối `if` hoặc `else` lại có thể chứa thêm if-else khác.

Cú pháp:

```
if expr1:
    if expr2:
        ...
    else:
        ...
else:
    if expr3:
        ...
    else:
        ...
```

Ví dụ Nested if...else

```
num = 8
print("num =", num)

if num % 2 == 0:
    if num % 3 == 0:
        print("Divisible by 3 and 2")
    else:
        print("Divisible by 2 not divisible by 3")
else:
    if num % 3 == 0:
        print("Divisible by 3 not divisible by 2")
    else:
        print("Not divisible by 2 not divisible by 3")
```

Kết quả chạy với nhiều giá trị num

```
num = 8
Divisible by 2 not divisible by 3

num = 15
Divisible by 3 not divisible by 2

num = 12
Divisible by 3 and 2

num = 5
Not divisible by 2 not divisible by 3
```

Câu lệnh `match-case` trong Python

- Xuất hiện từ Python 3.10.
- Tương tự `switch-case` trong C/Java nhưng mạnh mẽ hơn nhờ **pattern matching**.
- So khớp một biến với nhiều mẫu (pattern).
- Chỉ **case đầu tiên** khớp sẽ được thực thi.
- Có thể dùng _ như "wildcard" để bao quát các trường hợp còn lại.

Cú pháp:

```
match variable:
    case pattern1: statements
    case pattern2: statements
    ...
    case _: default_statements
```

Ví dụ cơ bản với match-case

```
def weekday(n):
    match n:
        case 0: return "Monday"
        case 1: return "Tuesday"
        case 2: return "Wednesday"
        case 3: return "Thursday"
        case 4: return "Friday"
        case 5: return "Saturday"
        case 6: return "Sunday"
        case _: return "Invalid day number"

print(weekday(3))
print(weekday(6))
print(weekday(7))
```

```
Thursday
Sunday
Invalid day number
```

Kết hợp nhiều case (OR)

```
def access(user):
    match user:
        case "admin" | "manager":
            return "Full access"
        case "Guest":
            return "Limited access"
        case _:
            return "No access"

print(access("manager"))
print(access("Guest"))
print(access("Ravi"))
```

```
Full access
Limited access
No access
```

Danh sách làm đối số trong match-case

```
def greeting(details):
    match details:
        case [time, name]:
            return f'Good {time} {name}!'
        case [time, *names]:
            msg = ''
            for name in names:
                msg += f'Good {time} {name}!\n'
            return msg

print(greeting(["Morning", "Ravi"]))
print(greeting(["Evening","Kajal","Praveen","Lata"]))
```

```
Good Morning Ravi!
Good Evening Kajal!
Good Evening Praveen!
Good Evening Lata!
```

Dùng if trong case

```
def intr(details):
    match details:
        case [amt, duration] if amt < 10000:
            return amt*10*duration/100
        case [amt, duration] if amt >= 10000:
            return amt*15*duration/100

print("Interest =", intr([5000,5]))
print("Interest =", intr([15000,3]))
```

```
Interest = 2500.0
Interest = 6750.0
```