# BÀI 28 PHẠM VI CỦA BIẾN

SAU BÀI NÀY EM SẼ:

- Biết và trình bày được ý nghĩa của phạm vi hoạt động của biến trong chương trình và hàm.

1. Một biến được định nghĩa trong chương trình chính (bên ngoài các hàm) thì sẽ được sử dụng như thế nào bên trong các hàm?
2. Một biến được khai báo bên trong một hàm thì có sử dụng được ở bên ngoài hàm đó hay không?

Bài này sẽ giúp em tìm câu trả lời cho các câu hỏi trên.

## 1. PHẠM VI CỦA BIẾN KHAI BÁO TRONG HÀM

**Hoạt động 1** **Phạm vi của biến khi khai báo trong hàm**

Quan sát các lệnh sau để tìm hiểu phạm vi có hiệu lực của biến khi khai báo bên trong một hàm.

Các biến được khai báo bên trong một hàm chỉ được sử dụng bên trong hàm đó. Chương trình chính không sử dụng được.

```
>>> def func(a,b):
        n = 10
        a = a*2
        b = a+b
        return a+b+n
>>> a = 1
>>> b = 2
>>> func(a,b)
16
>>> a,b
(1, 2)
>>> n
Traceback (most recent call last):
  File "<pyshell#11>", line 1, in <module>
    n
NameError: name 'n' is not defined
```

Bên trong hàm này có các biến n, a, b đang hoạt động. n = 10, a và b được thay đổi.

Đây là các biến bên ngoài hàm: a, b. Các biến này được gán: a = 1, b = 2.

Sau khi chạy hàm, các biến a, b vẫn không thay đổi.

Biến n chỉ có tác dụng bên trong hàm func, gọi bên ngoài hàm này sẽ bị báo lỗi.

Trong Python tất cả các biến khai báo bên trong hàm đều có tính địa phương (cục bộ), không có hiệu lực ở bên ngoài hàm.

**1.** Giả sử có các lệnh sau:

```
>>> a,b = 1,2
>>> def f(a,b):
        a = a + b
        b = b*a
        return a + b
```

Giá trị của a, b bằng bao nhiêu sau khi thực hiện lệnh sau?

a) `f(1, 2)` b) `f(10, 20)`

**2.** Ta có thể khai báo một biến bên trong hàm trùng tên với biến đã khai báo trước đó bên ngoài hàm không?

## 2. PHẠM VI CỦA BIẾN KHAI BÁO NGOÀI HÀM

**Hoạt động 2** Phạm vi của biến khi khai báo bên ngoài hàm

Quan sát các lệnh sau, tìm hiểu phạm vi có hiệu lực của biến khi khai báo bên ngoài một hàm.

**Ví dụ 1.** Biến khai báo bên ngoài hàm không có tác dụng bên trong hàm.

```
>>> def f(n):
        t = n + 1
        return t

>>> t = 10
>>> f(5)
6
>>> t
10
```

Trong chương trình chính biến t được khai báo bên ngoài hàm f() và gán giá trị 10. Khi gọi f(5), t sẽ được gán 6. Hàm trả lại giá trị 6. Nhưng khi thoát khỏi f(), t vẫn có giá trị 10. Do vậy biến t không có tác dụng bên trong hàm f().

**Ví dụ 2.** Bên trong hàm có thể truy cập để sử dụng giá trị của biến đã khai báo trước đó ở bên ngoài hàm.

```
>>> def f(a,b):
    return a + b + N

>>> N = 10
>>> f(1,2)
13
```

Trong chương trình chính, biến N được khai báo và gán giá trị 10. Khi gọi hàm f(1,2), giá trị trả lại là biểu thức có N tham gia. Vậy trong hàm f() được phép truy cập giá trị của biến N.

*Lưu ý*: Nếu muốn biến bên ngoài vẫn có tác dụng bên trong hàm thì cần khai báo lại biến này bên trong hàm với từ khoá **global**.

```
>>> def f(n):
    global t
    t = 2*n + 1
    return t
>>> t = 10
>>> f(1)
3
>>> t
3
```

Biến t được khai báo với từ khoá **global**.

Biến t khai báo trong chương trình chính và được gán giá trị 10.
Sau khi thực hiện f(1), giá trị t đã thay đổi.

Biến đã khai báo bên ngoài sẽ không có tác dụng bên trong hàm như một biến. Nếu muốn có tác dụng thì cần khai báo lại biến này trong hàm với từ khoá **global**.

Giả sử hàm f(x,y) được định nghĩa như sau:

```
>>> def f(x,y):
        a = 2*(x + y)
        print(a + n)
```

Kết quả nào được in ra khi thực hiện các lệnh sau?

```
n = 10
f(1,2)
```

## THỰC HÀNH

### Phạm vi của biến

**Nhiệm vụ 1.** Viết hàm với đầu vào là danh sách A chứa các số và số thực x. Hàm trả lại một danh sách kết quả B từ danh sách A bằng cách chỉ giữ lại các phần tử lớn hơn hoặc bằng x.

Hướng dẫn. Biến B kiểu danh sách cần được định nghĩa trong hàm và được bổ sung thêm các phần tử từ A nếu thoả mãn điều kiện lớn hơn hoặc bằng x.

```
def Select(A,x):
    B = []
    for k in range(len(A)):
        if A[k] >= x:
            B.append(A[k])
    return B
```

**Nhiệm vụ 2.** Viết hàm với đầu vào là xâu kí tự Str và số c, đầu ra là danh sách các từ được tách ra từ xâu Str nhưng đã được chuyển thành chữ in hoa hoặc chữ in thường, hoặc chỉ chuyển kí tự đầu các từ thành chữ in hoa tuỳ thuộc vào tham số đầu vào c như sau:

– Nếu c = 0, danh sách B là các từ được chuyển thành chữ in hoa.

– Nếu c = 1, danh sách B là các từ được chuyển thành chữ in thường.

– Nếu c = 2, danh sách B là các từ được chuyển viết chữ hoa kí tự đầu của mỗi từ.

Hướng dẫn. Chúng ta cần sử dụng các lệnh sau:

`Str.upper()` – chuyển kí tự của xâu thành chữ in hoa.

`Str.lower()` – chuyển kí tự của xâu thành chữ in thường.

`Str.title()` – chuyển kí tự đầu mỗi từ của xâu thành chữ in hoa, các kí tự khác chuyển về chữ thường.

Hàm được định nghĩa có dạng **`Tach_tu(Str,c)`**. Đầu tiên xâu `Str` cần được tách từ bằng lệnh `split()`. Sau đó danh sách kết quả sẽ được chuyển đổi chữ in hoa, in thường sử dụng một trong các lệnh trên tuỳ thuộc vào giá trị của đối số c.

```
def Tach_tu(Str, c):
    A = Str.split()
    for k in range(len(A)):
        if c == 0:
            A[k] = A[k].upper()
        if c == 1:
            A[k] = A[k].lower()
        if c == 2:
            A[k] = A[k].title()
    return A
```

**Nhiệm vụ 3.** Viết chương trình yêu cầu thực hiện lần lượt các việc sau, mỗi việc cần được thực hiện bởi một hàm:

1. Nhập từ bàn phím một dãy các số nguyên, mỗi số cách nhau bởi dấu cách. Chuyển các số này vào danh sách A và in danh sách A ra màn hình.
2. Trích từ danh sách A ra một danh sách B gồm các phần tử lớn hơn 0. In danh sách B ra màn hình.
3. Trích từ danh sách A ra một danh sách C gồm các phần tử nhỏ hơn 0. In danh sách C ra màn hình.

Hướng dẫn. Với mỗi việc trên được viết thành một hàm. Toàn bộ chương trình có thể như sau:

```
def Nhap_Dulieu():
    s = input("Nhập các số nguyên cách nhau bởi dấu cách: ")
    A = s.split()
    for k in range(len(A)):
        A[k] = int(A[k])
    return A
def getB(A):
    B = []
```

```
    for x in A:
        if x > 0:
            B.append(x)
    return B
def getC(A):
    C = []
    for x in A:
        if x < 0:
            C.append(x)
    return C
# Chương trình chính
A = Nhap_Dulieu()
print("Danh sách A:",A)
B = getB(A)
C = getC(A)
print("Danh sách B:",B)
print("Danh sách C:",C)
```

## LUYỆN TẬP

**1.** Viết hàm với đầu vào, đầu ra như sau:

– Đầu vào là danh sách `sList`, các phần tử là xâu kí tự.

– Đầu ra là danh sách `cList`, các phần tử là kí tự đầu tiên của các xâu kí tự tương ứng trong danh sách `sList`.

**2.** Viết hàm `Tach_day()` với đầu vào là danh sách `A`, đầu ra là hai danh sách B, C được mô tả như sau:

– Danh sách B thu được từ A bằng cách lấy ra các phần tử có chỉ số chẵn.

– Danh sách B thu được từ A bằng cách lấy ra các phần tử có chỉ số lẻ.



## VẬN DỤNG

**1.** Viết hàm có hai tham số đầu vào là m, n. Đầu ra trả lại hai giá trị là:

– ƯCLN của m, n.

– Bội chung nhỏ nhất (BCNN) của m, n.

*Gợi ý*: Sử dụng công thức ƯCLN(m, n) × BCNN(m, n) = m × n.

**2.** Viết chương trình nhập ba số tự nhiên từ bàn phím day, month, year, các số cách nhau bởi dấu cách. Các số này biểu diễn giá trị của ngày, tháng, năm nào đó. Chương trình cần kiểm tra và in ra thông báo số liệu đã nhập vào đó có hợp lệ hay không.