Chapter 5: Loops and iteration (Cấu Trúc Vòng Lặp)

1) Bước lặp lại nhiều lần

Ý nghĩa:
- *Dùng để lặp lại cùng một đoạn mã nhiều lần.*
- *Mỗi lần lặp, biến lặp (iteration variable) sẽ thay đổi.*

```
n = 5
while n > 0 :
    print(n)
    n = n - 1
print('Blastoff!')
print(n)
```

```
5
4
3
2
1
Blastoff!
0
```

2) Vòng lặp vô hạn

Khái niệm:
- *Là vòng lặp không bao giờ kết thúc vì điều kiện luôn đúng (True) hoặc biến điều kiện không thay đổi.*

```
n = 5
while n > 0 :
    print('Lather')
    print('Rinse')
print('Dry off!')
```

3) Thoát khỏi vòng lặp

Khái niệm:
- *break dùng để kết thúc vòng lặp ngay lập tức,*
- *bỏ qua phần còn lại và nhảy ra khỏi vòng lặp.*

```
while True:
    line = input('> ')
    if line == 'done' :
        break
    print(line)
print('Done!')
```

```
> hello there
hello there
> finished
finished
> done
Done!
```

4) Kết thúc vòng lặp với continue

Khái niệm:
- *Câu lệnh continue dùng để kết thúc sớm vòng lặp hiện tại,*
- *rồi quay lại đầu vòng lặp để kiểm tra điều kiện lần kế tiếp.*
- *Các lệnh sau continue trong vòng lặp sẽ không được thực hiện ở lần đó.*

```
while True:
    line = input('> ')
    if line[0] == '#' :
        continue
    if line == 'done' :
        break
    print(line)
print('Done!')
```

```
hello there

# don't print this
print this!

done
```

5) Một vòng lặp đơn giản

Ý nghĩa:
Mỗi lần lặp, biến (i) sẽ lấy từng giá trị trong danh sách.
Khi hết phần tử, vòng lặp kết thúc tự động (không cần điều kiện như while).

```
for i in [5, 4, 3, 2, 1] :
    print(i)
print('Blastoff!')
```

6) Vòng lặp xác định với chuỗi

- *Ý nghĩa:*
- *Biến letter lần lượt nhận từng ký tự trong chuỗi 'banana'.*
- *Khi hết ký tự, vòng lặp tự dừng.*

```
friends = ['Joseph', 'Glenn', 'Sally']
for friend in friends :
    print('Happy New Year:', friend)
print('Done!')
```

7) Tạo vòng lặp "thông minh"

Phần này nói về cách làm cho vòng lặp "thông minh" hơn bằng cách thêm logic xử lý bên trong (if, tính toán, điều kiện dừng...).

8) Lặp qua một tập hợp

Khái niệm:
- *Set là một tập hợp các phần tử duy nhất (không trùng lặp) và không có thứ tự.*
- *Dùng vòng lặp for để duyệt qua từng phần tử trong set.*

```
friends = {'Joseph', 'Glenn', 'Sally'}
for friend in friends:
    print('Happy New Year:', friend)
```

➡️ In lời chúc cho từng người trong set.

9) Tìm giá trị lớn nhất

```
largest_so_far = -1
print 'Before', largest_so_far
for the_num in [9, 41, 12, 3, 74, 15] :
    if the_num > largest_so_far :
        largest_so_far = the_num
    print largest_so_far, the_num

print 'After', largest_so_far
```

```
Before -1
9 9
41 41
41 12
41 3
74 74
74 15
After 74
```

Giải thích nhanh
- *: Biến lưu giá trị lớn nhất đã gặp.*
- *: Biến tạm đại diện cho từng số trong danh sách.*
- *Vòng lặp : Duyệt qua từng phần tử.*
- *Câu lệnh : So sánh và cập nhật giá trị lớn nhất nếu cần.*

10) Đếm trong một vòng lặp

Nguyên lý hoạt động :
1. *zork = 0: Đặt biến đếm bắt đầu từ 0.*
2. *for thing in list: Vòng lặp chạy một lần cho mỗi phần tử (tổng cộng 6 lần).*
3. *zork = zork + 1: Trong mỗi lần chạy, biến đếm tăng thêm 1.*
4. *Kết quả: zork lưu giữ tổng số phần tử (6) sau khi vòng lặp kết thúc.*

```
zork = 0
print 'Before', zork
for thing in [9, 41, 12, 3, 74, 15] :
    zork = zork + 1
    print zork, thing
print 'After', zork
```

```
Before 0
1 9
2 41
3 12
4 3
5 74
6 15
After 6
```

11) Tính tổng trong một vòng lặp

```
zork = 0
print 'Before', zork
for thing in
    zork = zork + thing
    print zork  thing
print 'After', zork
```

```
Before 0
9 9
50 41
62 12
65 3
139 74
154 15
After 154
```

Quá trình Cộng Dồn:

Lần lặp	thing (Giá trị hiện tại)	zork cũ	zork = zork + thing	zork mới	Output
1	9	0	0 + 9	9	9 9
2	41	9	9 + 41	50	50 41
3	12	50	50 + 12	62	62 12
4	3	62	62 + 3	65	65 3
5	74	65	65 + 74	139	139 74
6	15	139	139 + 15	154	154 15

12) Tìm giá trị trung bình trong vòng lặp

```
count = 0
sum = 0
print('Before', count, sum)
for value in [9, 41, 12, 3, 74, 15] :
    count = count + 1
    sum = sum + value
    print(count, sum, value)
print('After', count, sum, sum / count)
```

```
Before 0 0
1 9 9
2 50 41
3 62 12
4 65 3
5 139 74
6 154 15
After 6 154 25.666
```

13) Lọc trong vòng lặp

```
print('Before')
for value in [9, 41, 12, 3, 74, 15] :
    if value > 20:
        print('Large number',value)
print('After')
```

```
Before
Large number 41
Large number 74
After
```

14) Tìm kiếm bằng biến Boolean

```
found = False
print 'Before', found
for value in [9, 41, 12, 3, 74, 15] :
    if value == 3 :
        found = True
    print found, value
print 'After', found
```

```
Before False
False 9
False 41
False 12
True 3
True 74
True 15
After True
```

15) Cách tìm giá trị lớn nhất

```
largest_so_far = -1
print('Before', largest_so_far)
for the_num in [9, 41, 12, 3, 74, 15] :
    if the_num > largest_so_far :
        largest_so_far = the_num
    print(largest_so_far, the_num)

print('After', largest_so_far)
```

```
Before -1
9 9
41 41
41 12
41 3
74 74
74 15
After 74
```

16) Tìm giá trị nhỏ nhất

```
smallest = None
print('Before')
for value in [9, 41, 12, 3, 74, 15] :
    if smallest is None :
        smallest = value
    elif value < smallest :
        smallest = value
    print(smallest, value)
print('After', smallest)
```

```
Before
9 9
9 41
9 12
3 3
3 74
3 15
After 3
```