Vòng lặp

Nội dung

Giới thiệu vòng lặp for

Giới thiệu vòng lặp while

Vòng lặp **for**

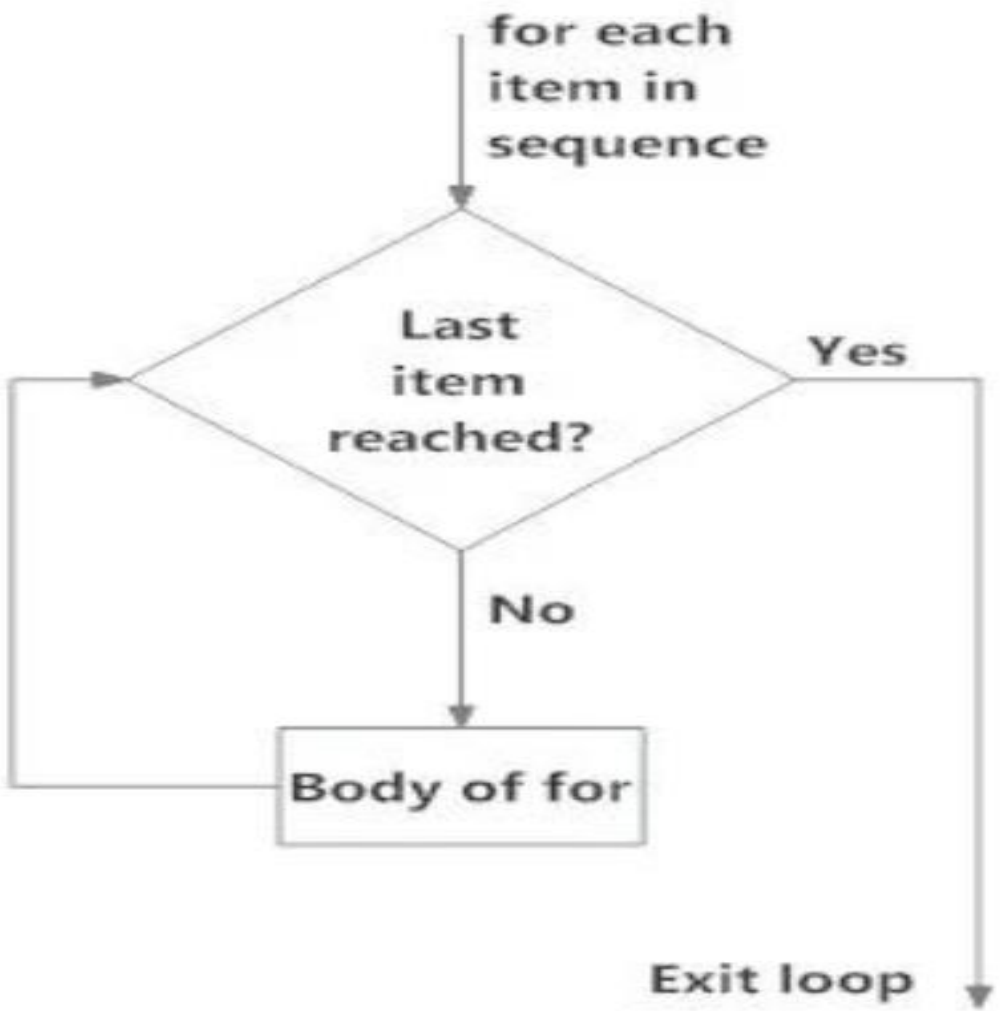

Fig: operation of for loop

Vòng lặp **for**

Ví dụ: Lặp qua 1 chuỗi

```
for val in "apple" :
    print(val)
```

Kết quả: a, p, p ,l ,e

Vòng lặp sẽ trả ra từng phần tử trong chuỗi đã cho

Hàm range() với vòng lặp

Hàm range trả về một mảng trong đó tổng số phần tử sẽ phụ thuộc vào các tham số truyền vào.

Cú pháp:

```
range(start, end, step)
```

Trong đó:

- start: giá trị bắt đầu
- end: giá trị kết thúc
- step: khoảng cách giữa các phần tử hay còn gọi là bước nhảy

Vòng lặp **for**

```
for i in range(5, 10):
    print(i, end=', ')
```

Kết quả sẽ tạo một mảng gồm 5 phần tử có giá trị lần lượt là 5 -> 9

```
5, 6, 7, 8, 9,
```

Vòng lặp **for**

```
for i in range(5):
    print(i, end=', ')
```

Kết quả sẽ tạo một mảng gồm 5 phần tử có giá trị lần lượt từ 0 -> 5.

```
0, 1, 2, 3, 4,
```

Vòng lặp **for**

```
for i in range(1, 10, 2):
    print(i, end=', ')
```

Kết quả trả về một mảng 5 phần tử có giá lần lượt là 1, 3, 5, 7, 9 vì bước nhảy là 2.

```
1, 3, 5, 7, 9,
```

Vòng lặp while

Vòng lặp for lấy một tập hợp các mục và thực thi một khối mã một lần cho mỗi mục trong tập hợp. Ngược lại, vòng lặp while chạy *miễn là*, hoặc *trong khi*, một điều kiện nhất định là đúng.

Vòng lặp với hành động

Có thể sử dụng vòng lặp while để đếm dần một chuỗi số. Ví dụ, vòng lặp while sau đếm từ 1 đến 5

```
current_number = 1
while current_number <= 5:
    print(current_number)
    current_number += 1
```

```
1
2
3
4
5
```

Python lặp lại vòng lặp miễn là điều kiện current_number <= 5 là đúng

Các chương trình ta sử dụng hàng ngày rất có thể chứa các vòng lặp while. Ví dụ, một trò chơi cần một vòng lặp while khi tiếp tục chạy khi ta muốn và nó có thể ngừng chạy ngay sau khi ta yêu cầu nó thoát.

Để người dùng lựa chọn khi nào thoát

Xây dựng chương trình parrot.py để người dùng muốn luôn chạy bằng cách đặt hầu hết chương trình bên trong vòng lặp while

Xác định một giá trị thoát và sau đó giữ chương trình chạy miễn là người dùng chưa nhập giá trị thoát

① prompt = "\nTell me something, and I will repeat it back to you:"

prompt += "\nEnter 'quit' to end the program. ”

② message = ""

③ while message != 'quit':

message = input(prompt)

print(message)

Để người dùng lựa chọn khi nào thoát

Tại message = input(prompt), Python hiển thị lời nhắc và đợi người dùng nhập thông tin đầu vào của họ. Bất cứ thứ gì người dùng nhập đều được gán vào biến message và in ra; sau đó, Python đánh giá lại điều kiện của vòng lặp while.

Miễn là người dùng chưa nhập từ ‘quit’, lời nhắc được hiển thị lại và Python chờ thêm đầu vào. Khi người dùng cuối cùng nhập ‘quit’, Python dừng thực hiện vòng lặp while và chương trình kết thúc:

```
Tell me something, and I will repeat it back to you:
Enter 'quit' to end the program. Hello everyone!
Hello everyone!

Tell me something, and I will repeat it back to you:
Enter 'quit' to end the program. Hello again.
Hello again.

Tell me something, and I will repeat it back to you:
Enter 'quit' to end the program. quit
quit
```

Để người dùng lựa chọn khi nào thoát

Chương trình này hoạt động tốt, ngoại trừ việc nó in từ ‘quit’ như thể nó là một thông điệp thực tế. Một kiểm tra có điều kiện đơn giản nếu giải quyết được điều này

```
prompt = "\nTell me something, and I will repeat it back to you:"
prompt += "\nEnter 'quit' to end the program. "
message = ""
while message != 'quit':
    message = input(prompt)
    if message != 'quit':
        print(message)
```

```
Tell me something, and I will repeat it back to you:
Enter 'quit' to end the program. Hello everyone!
Hello everyone!

Tell me something, and I will repeat it back to you:
Enter 'quit' to end the program. Hello again.
Hello again.

Tell me something, and I will repeat it back to you:
Enter 'quit' to end the program. Quit
```

Sử dụng Flag (cờ)

Đối với một chương trình sẽ chạy miễn là có nhiều điều kiện đúng, ta có thể xác định một biến xác định xem toàn bộ chương trình có đang hoạt động hay không. Biến này, được gọi là flag, hoạt động như một tín hiệu cho chương trình

```
①   active = True
②   while active:
        message = input(prompt)
③       if message == 'quit':
            active = False
④       else:
            print(message)
```

Chương trình này có đầu ra giống như ví dụ trước, nơi chúng ta đã đặt kiểm tra điều kiện trực tiếp trong câu lệnh while Nhưng bây giờ chúng ta có một flag để cho biết liệu chương trình tổng thể có ở trạng thái hoạt động hay không, nó sẽ dễ dàng thêm nhiều kiểm tra hơn cho các sự nên làm cho *active* trở thành False

Sử dụng **break** để thoát khỏi vòng lặp

Để thoát khỏi vòng lặp while ngay lập tức mà không cần chạy bất kỳ mã nào còn lại trong vòng lặp, bất kể kết quả của bất kỳ kiểm tra điều kiện nào, hãy sử dụng câu lệnh break

```
prompt = "\nPlease enter the name of a city you have visited:"
prompt += "\n(Enter 'quit' when you are finished.) "
while True:
    city = input(prompt)
    if city == 'quit':
        break
    else:
        print(f"I'd love to go to {city.title()}!")
```

```
Please enter the name of a city you have visited:
(Enter 'quit' when you are finished.) Hanoi
I'd love to go to Hanoi !

Please enter the name of a city you have visited:
(Enter 'quit' when you are finished.) Saigon
I'd love to go to Saigon!

Please enter the name of a city you have visited:
(Enter 'quit' when you are finished.) quit
```

Sử dụng **continue** trong vòng lặp

Thay vì thoát ra khỏi vòng lặp hoàn toàn mà không thực hiện phần còn lại của nó , ta có thể sử dụng câu lệnh continue để quay lại phần đầu của vòng lặp dựa trên kết quả của một bài kiểm tra có điều kiện

```
current_number = 0
while current_number < 10:
    current_number += 1
    if current_number % 2 == 0:
        continue
    print(current_number)
```

Tránh lặp vô hạn

Mỗi vòng lặp while cần một cách để dừng chạy để nó sẽ không tiếp tục chạy mãi mãi.

```
#normal
x = 1
while x <= 5:
    print(x)
    x += 1
```

```
# This loop runs forever!
x = 1
while x <= 5:
    print(x)
```

```
1
1
1
1
--snip--
```

Nếu chương trình bị mắc kẹt trong một vòng lặp vô hạn, hãy nhấn cTrL-C hoặc chỉ cần đóng cửa sổ đầu cuối hiển thị đầu ra chương trình

Đảm bảo ít nhất một phần của chương trình có thể làm cho điều kiện của vòng lặp True hoặc chương trình tới được một câu lệnh break.

Bài tập

TRY IT YOURSELF

7-4. Pizza Toppings: Write a loop that prompts the user to enter a series of pizza toppings until they enter a `'quit'` value. As they enter each topping, print a message saying you'll add that topping to their pizza.

7-5. Movie Tickets: A movie theater charges different ticket prices depending on a person's age. If a person is under the age of 3, the ticket is free; if they are between 3 and 12, the ticket is $10; and if they are over age 12, the ticket is $15. Write a loop in which you ask users their age, and then tell them the cost of their movie ticket.

Bài tập

7-6. Three Exits: Write different versions of either Exercise 7-4 or Exercise 7-5 that do each of the following at least once:

- Use a conditional test in the `while` statement to stop the loop.
- Use an `active` variable to control how long the loop runs.
- Use a `break` statement to exit the loop when the user enters a `'quit'` value.

7-7. Infinity: Write a loop that never ends, and run it. (To end the loop, press CTRL-C or close the window displaying the output.)

XIN CẢM ƠN!