Bài 25: VÒNG LẶP WHILE TRONG PYTHON

Xem bài học trên website để ủng hộ Kteam: Vòng lặp While trong Python

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage How Kteam nhé!

Dẫn nhập

Trong bài trước, Kteam đã giới thiệu đến bạn CÂU ĐIỀU KIỆN IF - một dạng cấu trúc rẽ nhánh rất quan trọng trong mọi ngôn ngữ lập trình không chỉ riêng Python

Ở bài này Kteam sẽ giới thiệu với các bạn **Vòng lặp While trong Python**.

Nội dung

Để đọc hiểu bài này tốt nhất bạn cần:

- Cài đặt sẵn MÔI TRƯỜNG PHÁT TRIỂN CỦA PYTHON.
- Xem qua bài CÁCH CHẠY CHƯƠNG TRÌNH PYTHON.
- Nắm CÁCH GHI CHÚ và BIẾN TRONG PYTHON.
- CÁC KIỂU DỮ LIỆU ĐƯỢC GIỚI THIỆU TRONG PYTHON
- CÂU ĐIỀU KIỆN IF TRONG PYTHON

Bạn và Kteam sẽ cùng tìm hiểu những nội dung sau đây

- Đặt vấn đề
- Cấu trúc vòng lặp while và cách hoạt động
- Sử dụng vòng lặp để xử lí chuỗi, list, tuple

- Câu lệnh break và continue
- Cấu trúc vòng lặp while-else và cách hoạt động

Đặt vấn đề

Lại là câu chuyện về Tèo – Kter "bờ rào" của Kteam. Sắp tới là sinh nhật Tèo, Tèo tham vọng mời tất cả thành viên trong group lập trình của Kteam. Thế nên, Tèo mua một xấp giấy về ghi thiệp mời các bạn tham dự buổi tiệc.

Một bạn, hai bạn, rồi ba bạn và tới bạn thứ năm thì Tèo đã thấm mệt. Dòng chữ cũng không được nắn nót như ban đầu. Nhớ lại là còn hơn 9999 người cần phải mời nữa. Nên Tèo mệt quá, không muốn mời ai nữa và ăn sinh nhật một mình luôn.

Nếu bạn là Tèo, bạn sẽ viết được bao nhiêu tấm thiệp với dòng chữ nắn nót và đẹp như tấm thiệp ban đầu? Liệu bạn có đủ kiên nhẫn viết hết 1000 tấm thiệp thậm chí là 100000?

Hiển nhiên là "Không!". Mà trường hợp của Tèo cũng chả phải hiếm. Vì vậy, con người đã tạo ra máy tính để giúp họ làm những việc tương tự. Máy tính có khả năng lặp đi lặp lại một tiến trình với số lần rất lớn. Hiệu suất của lần cuối cùng cũng như lần đầu tiên. Thêm một điều nữa là công việc đó được làm với một tốc độ chóng mặt

Làm sao chúng làm được như vậy? Đó là nhờ tuyệt kĩ vòng lặp. Và chúng ta sẽ bắt đầu đi tìm hiểu chiêu thức vòng lặp đầu tiên trong Python chính là **While**.

Cấu trúc vòng lặp while và cách hoạt động

Nào! Cùng ngó sơ cấu trúc, sau đó Kteam sẽ giải thích cho bạn cách mà nó hoạt động

```
while expression:

    # while-block
```

Lưu ý: Việc chia block như thế này cũng giống như khi bạn sử dụng **câu lệnh if** và đã được Kteam giới thiệu ở bài trước CẤU TRÚC RẼ NHÁNH.

Nó sẽ hoạt động ra sao?

Rất đơn giản! Việc đầu tiên, Python sẽ kiểm tra giá trị boolean của expression. Nếu là **False**, thì bỏ qua while-block và đến với câu lệnh tiếp theo. Ngược lại, sẽ thực hiện toàn bộ câu lệnh trong while-block. Sau khi thực hiện xong, quay ngược lại kiểm tra giá trị boolean của expression một lần nữa. Nếu **False** thì bỏ qua while-block, còn **True** thì tiếp tục thực hiện while-block. Và sau khi thực hiện xong while-block lại quay về kiểm tra giá trị boolean expression như những lần trước.

Ví dụ:

```
>>> k = 5
>>>
>>> while k > 0:
...     print('k =', k)
...     k -= 1
...
k = 5
k = 4
k = 3
k = 2
k = 1
>>> k # k bằng 0 nên > 0 là một boolean False, do đó vòng lặp đã kết thúc
0
```

Sử dụng vòng lặp để xử lí chuỗi, list, tuple

Đây là những iterable cho phép ta truy xuất một giá trị bất kí trong nó bằng phương pháp indexing. Thế nên, ta có thể nhờ điều này kết hợp với vòng lặp để xử lí chúng.

```
>>> s = 'How Kteam'
>>> idx = 0 # vị trí bắt đầu bạn muốn xử lí của chuỗi
>>> length = len(s) # lấy độ dài chuỗi làm mốc kết thúc
>>>
>>> while idx < length:
...     print(idx, 'stands for', s[idx])
...     idx += 1 # di chuyển index tới vị trí tiếp theo
...
0 stands for H
1 stands for o
2 stands for w
3 stands for
4 stands for K
5 stands for t
6 stands for e
7 stands for a
8 stands for m
```

Đơn giản phải không nào. **List** và **Tuple** hoàn toàn tương tự.

Câu lệnh break và continue

Lưu ý: Hai câu lệnh này chỉ có thể dùng trong các vòng lặp

Câu lệnh break

Câu lệnh break dùng để kết thúc vòng lặp. Cứ nó nằm trong block của vòng lặp nào thì vòng lặp đó sẽ kết thúc khi chạy câu lệnh này.

Trong trường hợp vòng lặp a chứa vòng lặp b. Trong **vòng lặp b** chạy câu lệnh break thì chỉ **vòng lặp b** kết thúc, còn **vòng lặp a** thì không.

Ví dụ *:

```
>>> five_even_numbers = []
>>> k_number = 1
>>>
>>> while True: # vòng lặp vô hạn vì giá trị này là hằng nên ta không thể tác động được
...     if k_number % 2 == 0: # nếu k_number là một số chẵn
...         five_even_numbers.append(k_number) # thêm giá trị của k_number vào list
...     if len(five_even_numbers) == 5: # nếu list này đủ 5 phần tử
...         break # thì kết thúc vòng lặp
...     k_number += 1
...
>>> five_even_numbers
[2, 4, 6, 8, 10]
>>> k_number
10
```

Câu lệnh continue

Câu lệnh này dùng để chạy tiếp vòng lặp. Giả sử một vòng lặp có cấu trúc như sau:

```
while expression:
    #while-block-1
    continue
```

```
    #while-block-2
```

Khi thực hiện xong while-block-1, câu lệnh **continue** sẽ tiếp tục vòng lặp, không quan tâm những câu lệnh ở dưới continue và như vậy nó đã bỏ qua while-block-2.

Ví dụ:

```
>>> k_number = 1
>>> while k_number < 10:
...     if k_number % 2 == 0: # nếu k_number là số chẵn
...         k_number += 1  # thì tăng một đơn vị cho k_number và tiếp tục vòng lặp
...         continue
...     print(k_number, 'is odd number')
...     k_number += 1
...
1 is odd number
3 is odd number
5 is odd number
7 is odd number
9 is odd number
```

Cấu trúc vòng lặp while-else và cách hoạt động

Ta sẽ xem cấu trúc trước:

```
while expression:

    # while-block

else:
```

```
        # else-block
```

Cấu trúc này gần tương tự như **while** bình thường. Thêm một điều, khi vòng vòng lặp **while** kết thúc thì khối lệnh else-block sẽ được thực hiện.

Ví dụ:

```
>>> while k < 3:
...     print('value of k is', k)
...     k += 1
... else:
...     print('k is not less than 3 anymore')
...
value of k is 0
value of k is 1
value of k is 2
k is not less than 3 anymore
```

Trong trường hợp trong while-block chạy câu lệnh break thì vòng lặp **while** sẽ kết thúc và phần else-block cũng sẽ không được thực hiện.

```
>>> k = 0
>>> while k < 5:
...     print('value of k is', k)
...     k += 1
...     if k > 3:
...         print('k is greater than 3')
...         break
... else:
...     print('k is not less than 5 anymore')
...
value of k is 0
value of k is 1
value of k is 2
value of k is 3
k is greater than 3
```

Củng cố bài học

Đáp án bài trước

Bạn có thể tìm thấy câu hỏi của phần này tại CÂU HỎI CỦNG CỐ trong bài CẤU TRÚC RẼ NHÁNH TRONG PYTHON

Cách 1:

```
k1 = int(input('Nhap so thu nhat\n=> '))
k2 = int(input('Nhap so thu hai\n=> '))
k3 = int(input('Nhap so thu ba\n=> '))

if k1 > k2 and k1 > k3:
    print('so lon nhat la', k1)
elif k2 > k1 and k2 > k3:
    print('so lon nhat la', k2)
else:
    print('so lon nhat la', k3)
```

Cách 2:

```
k1 = int(input('Nhap so thu nhat\n=> '))
k2 = int(input('Nhap so thu hai\n=> '))
k3 = int(input('Nhap so thu ba\n=> '))

if k1 > k2 and k1 > k3: print('so lon nhat la', k1)
elif k2 > k1 and k2 > k3: print('so lon nhat la', k2)
else: print('so lon nhat la', k3)
```

Câu hỏi củng cố

1. Viết lại một vòng lặp có chức năng tương tự **ví dụ *** nhưng không dùng câu lệnh break
2. Cho một file text tên draft.txt như sau:

```
an so dfn Kteam odsa in fasfna Kteam mlfjier
as dfasod nf ofn asdfer fsan dfoans ldnfad Kteam asdfna
asdofn sdf pzcvqp Kteam dfaojf kteam dfna Kteam dfaodf
afdna Kteam adfoasdf ncxvo aern Kteam dfad
```

Trong file này có một số chữ **Kteam** (**Kteam** sẽ không xuất hiện ở đầu dòng), và trước nó là một chữ ngẫu nhiên nào đó và nhiệm vụ của bạn là đổi chữ đó thành **How**. Nhớ là sử dụng vòng lặp.

Sau khi đổi thành công, bạn lưu nội dung đó vào file tên kteam.txt.

Đây là mẫu của kteam.txt:

```
an so How Kteam odsa in How Kteam mlfjier
as dfasod nf ofn asdfer fsan dfoans How Kteam asdfna
asdofn sdf How Kteam dfaojf kteam How Kteam dfaodf
How Kteam adfoasdf ncxvo How Kteam dfad
```

3. Sắp xếp một mảng số nguyên có dạng như sau:

```
[56, 14, 11, 756, 34, 90, 11, 11, 65, 0, 11, 35]
```

Lưu ý: là các số 11 là những số cố định không được thay đổi vị trí của nó.

Sau khi sắp xếp lại mảng trên sẽ là:

```
[0, 14, 11, 34, 35, 56, 11, 11, 65, 90, 11, 756]
```

Đáp án của phần này sẽ được trình bày ở bài tiếp theo. Tuy nhiên, Kteam khuyến khích bạn tự trả lời các câu hỏi để củng cố kiến thức cũng như thực hành một cách tốt nhất!

Kết luận

Qua bài viết này, Bạn đã biết về VÒNG LẶP WHILE TRONG PYTHON.

Ở bài viết sau, Kteam sẽ nói đến một vòng lặp nữa đó là VÒNG LẶP FOR TRONG PYTHON.

Cảm ơn bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên "**Luyện tập – Thử thách – Không ngại khó**".