TÁC GIẢ: 8 SYNC

CỘNG ĐỒNG

- Kevin Nguyễn
- Nhóm Chia Sẻ Công Nghệ
- Nhóm BlockChain
- Tiktok: 8 Sync
- Youtube: 8 Sync Dev
- Zalo

KHÓA HỌC:

- Fullstack Python
- Fullstack Nextjs
- Fullstack Android-IOS

Tài liệu sẽ được cập nhật định kì và thông báo trong group nên các bạn chú ý nhen .

SLICING CHUỖI TRONG PYTHON

Trong Python, một chuỗi là một chuỗi các ký tự Unicode được sắp xếp. Mỗi ký tự trong chuỗi có một chỉ mục duy nhất trong chuỗi. Chỉ mục bắt đầu từ 0. Ký tự đầu tiên trong chuỗi có chỉ mục vị trí của nó là 0. Chỉ mục tiếp tục tăng dần về phía cuối chuỗi.

Nếu một biến chuỗi được khai báo như sau: `var = "HELLO PYTHON"`, chỉ mục của mỗi ký tự trong chuỗi sẽ như sau:

< 8 Sync Dev />

```
H   E   L   L   O       P   Y   T   H   O   N
0   1   2   3   4   5   6   7   8   9   10  11
```

TRUY CẬP KÝ TỰ TRONG CHUỖI PYTHON

Python cho phép bạn truy cập bất kỳ ký tự nào từ chuỗi bằng cách sử dụng chỉ mục của nó. Trong trường hợp này, 0 là giới hạn dưới và 11 là giới hạn trên của chuỗi. Vì vậy, `var[0]` trả về 'H', `var[6]` trả về 'P'. Nếu chỉ mục trong dấu ngoặc vuông vượt quá giới hạn trên, Python sẽ ném ra IndexError.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
print(var[0])   # 'H'
print(var[7])   # 'Y'
print(var[11])  # 'N'
print(var[12])  # IndexError: string index out of range
```

CHỈ MỤC ÂM VÀ DƯƠNG TRONG CHUỖI PYTHON

Một trong những tính năng đặc biệt của các loại chuỗi Python (và do đó là một đối tượng chuỗi) là nó có một hệ thống chỉ mục âm. Trong ví dụ trên, một hệ thống chỉ mục dương được sử dụng, trong đó chỉ mục tăng dần từ trái sang phải. Trong trường hợp của chỉ mục âm, ký tự ở cuối có chỉ mục -1 và chỉ mục giảm dần từ phải sang trái, kết quả là ký tự đầu tiên H có chỉ mục -12.

Ví dụ

< 8 Sync Dev />

```
print(var[-1])   # 'N'
print(var[-5])   # 'Y'
print(var[-12])  # 'H'
print(var[-13])  # IndexError: string index out of range
```

Một lần nữa, nếu chỉ mục vượt ra khỏi phạm vi, IndexError sẽ được gặp phải.

SỬ DỤNG CHỈ MỤC ÂM VÀ DƯƠNG

Chúng ta có thể sử dụng chỉ mục dương hoặc âm để truy xuất một ký tự từ chuỗi.

Ví dụ

< 8 Sync Dev />

```
print(var[0], var[-12])  # ('H', 'H')
print(var[7], var[-5])   # ('Y', 'Y')
print(var[11], var[-1])  # ('N', 'N')
```

Trong Python, chuỗi là một đối tượng không thay đổi. Đối tượng là không thay đổi nếu nó không thể được sửa đổi tại chỗ, sau khi được lưu trữ trong một vị trí bộ nhớ cụ thể. Bạn có thể lấy bất kỳ ký tự nào từ chuỗi bằng cách sử dụng chỉ mục của nó, nhưng bạn không thể thay thế nó bằng một ký tự khác.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
var[7] = "y"  # TypeError: 'str' object does not support item assignment
```

Lỗi TypeError là do chuỗi là không thay đổi.

SỬ DỤNG PHÉP CẮT CHUỖI TRONG PYTHON

Python định nghĩa ":" như là toán tử cắt chuỗi. Nó trả về một chuỗi con từ chuỗi gốc. Sử dụng như sau:

< 8 Sync Dev />

```
substr = var[x:y]
```

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"

print("var:", var)
print("var[3:8]:", var[3:8])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[3:8]: LO PY
```

SỬ DỤNG CHỈ MỤC ÂM VÀ DƯƠNG TRONG PHÉP CẮT CHUỖI PYTHON

Chỉ mục âm cũng có thể được sử dụng cho việc cắt.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
print("var:", var)
print("var[3:8]:", var[3:8])
print("var[-9:-4]:", var[-9:-4])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[3:8]: LO PY
var[-9:-4]: LO PY
```

GIÁ TRỊ MẶC ĐỊNH CỦA CHỈ MỤC TRONG PHÉP CẮT CHUỖI

Cả hai toán hạng cho toán tử Cắt của Python là tùy chọn. Toán hạng đầu tiên mặc định là không, điều này có nghĩa là nếu chúng ta không cung cấp toán hạng đầu tiên, cắt bắt đầu từ ký tự ở chỉ mục 0, tức là ký tự đầu tiên. Nó cắt chuỗi con trái nhất lên đến "y-1" ký tự.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
print("var:", var)
```

```
print("var[0:5]:", var[0:5])
print("var[:5]:", var[:5])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[0:5]: HELLO
var[:5]: HELLO
```

Tương tự, toán hạng y cũng là tùy chọn. Mặc định là "-1", có nghĩa là chuỗi sẽ được cắt từ vị trí x đến cuối chuỗi.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
print("var:", var)
print("var[6:12]:", var[6:12])
print("var[6:]:", var[6:])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[6:12]: PYTHON
var[6:]: PYTHON
```

Tự nhiên, nếu cả hai toán hạng đều không được sử dụng, phép cắt sẽ bằng với chuỗi gốc. Điều này bởi vì "x" là 0, và "y" mặc định là chỉ mục cuối cùng+1 (hoặc -1).

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
print("var:", var)
print("var[0:12]:", var[0:12])
print("var[:]:", var[:])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[0:12]: HELLO PYTHON
var[:]: HELLO PYTHON
```

Toán hạng bên trái phải nhỏ hơn toán hạng bên phải, để lấy một chuỗi con của chuỗi gốc. Python không ném ra bất kỳ lỗi nào, nếu toán hạng bên trái lớn hơn, nhưng trả về một chuỗi rỗng.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"
print("var:", var)
print("var[-1:7]:", var[-1:7])
print("var[7:0]:", var[7:0])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[-1:7]:
var[7:0]:
```

KIỂU TRẢ VỀ CỦA PHÉP CẮT CHUỖI

Cắt trả về một chuỗi mới. Bạn hoàn toàn có thể thực hiện các thao tác chuỗi như ghép chuỗi, hoặc cắt trên chuỗi đã được cắt.

Ví dụ

< 8 Sync Dev />

```
var = "HELLO PYTHON"

print("var:", var)
print("var[:6][:2]:", var[:6][:2])

var1 = var[:6]
print("slice:", var1)
print("var1[:2]:", var1[:2])
```

Kết quả sẽ là:

< 8 Sync Dev />

```
var: HELLO PYTHON
var[:6][:2]: HE
slice: HELLO
var1[:2]: HE
```