Bài 10: KIỂU DỮ LIỆU CHUỖI TRONG PYTHON (Phần 4 – Các phương thức chuỗi)

Xem bài học trên website để ủng hộ Kteam: KIỂU DỮ LIỆU CHUỖI TRONG PYTHON (Phần 4 – Các phương thức chuỗi)

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 thêm cho các bạn về ĐỊNH DẠNG CHUỖI TRONG PYTHON.

Ở bài này, chúng ta sẽ nói đến **KIỂU DỮ LIỆU CHUỖI** trong Python và nội dung chính là các phương thức của kiểu dữ liệu chuỗi.

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.
- KIỂU DỮ LIỆU SỐ và KIỂU DỮ LIỆU CHUỖI trong Python.

Trong bài học này, chúng ta sẽ cùng tìm hiểu các vấn đề:

- Giới thiệu về phương thức của kiểu dữ liệu chuỗi trong Python
- Các phương thức biến đổi
- Các phương thức định dạng
- Các phương thức xử lí

Giới thiệu về phương thức của kiểu dữ liệu chuỗi trong Python

Kiểu dữ liệu của Python có khá nhiều các phương thức chuẩn (chưa tính đến các thư viện) để xử lí chuỗi.

Kteam sẽ giới thiệu với các bạn các phương thức cơ bản thường được sử dụng. Để có thể có được đầy đủ những phương thức chuẩn, hãy ghé thăm tài liệu của Python tại trang

> **Python.org > String methords**

Các phương thức này đều có giá trị trả về và không ảnh hưởng gì tới giá trị ban đầu. Tương tự như một số hàm mà các bạn đã biết: int, float, str

```
>>> k = '12'
>>> int(k)
12
>>> type(k)  # k vẫn thuộc lớp str
<class 'str'>
```

Các phương thức biến đổi

Phương thức capitalize

Cú pháp:

```
<chuỗi>.capitalize()
```

Công dụng: Trả về một chuỗi với kí tự đầu tiên được viết hoa và viết thường tất cả những kí tự còn lại.

```
>>> ‘kteaM’.capitalize()
‘Kteam’
>>> ‘hello, Howkteam!’.capitalize()
‘Hello, howkteam!’
>>> ‘  howKTEAM’.capitalize()
‘  howkteam’
```

Phương thức upper

Cú pháp:

```
<chuỗi>.upper()
```

Công dụng: Trả về một chuỗi với tất cả các kí tự được chuyển thành các kí tự viết hoa

```
>>> ‘kter’.upper()
‘KTER’
>>> 'HOW kteam'.upper()
'HOW KTEAM'
>>> ' python'.upper()
' PYTHON'
```

Phương thức lower

Cú pháp:

```
<chuỗi>.lower()
```

Công dụng: Trả về một chuỗi với tất cả các kí tự được chuyển thành các kí tự viết thường

```
>>> 'FREE education'.lower()
'free education'
>>> 'kteam'.lower()
'kteam'
>>> ' kTer'.lower()
' kter'
```

Phương thức swapcase

Cú pháp:

```
<chuỗi>.swapcase()
```

Công dụng: Trả về một chuỗi với các kí tự viết hoa được chuyển thành viết thường, các kí tự viết thường được chuyển thành viết hoa

```
>>> 'free EDUCATION'.swapcase()
'FREE education'
>>> 'HoW kTeAm'.swapcase()
'hOw KtEaM'
```

Phương thức title

Cú pháp:

```
<chuỗi>.title()
```

Công dụng: Trả về một chuỗi với định dạng tiêu đề, có nghĩa là các từ sẽ được viết hoa chữ cái đầu tiên, còn lại là viết thường

```
>>> 'share to be better'.title()
'Share To Be Better'
>>> 'FREE EDUCATION'.title()
'Free Education'
```

Các phương thức định dạng

Phương thức center

Cú pháp:

```
<chuỗi>.center(width, [fillchar])
```

Công dụng: Trả về một chuỗi được căn giữa với chiều rộng **width**.

- Nếu **fillchar** là None (không được nhập vào) thì sẽ dùng kí tự khoảng trắng để căn, không thì sẽ căn bằng kí tự fillchar.
- Một điều nữa là kí tự fillchar là một chuỗi có độ dài là 1.

```
>>> 'abc'.center(12)
'    abc     '
>>> 'abc'.center(12, '*')
'****abc*****'
>>> 'abc'.center(12, '*a')
Traceback (most recent call last):
```

```
  File "<stdin>", line 1, in <module>
TypeError: The fill character must be exactly one character long
>>> 'abc'.center(12, '')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: The fill character must be exactly one character long
```

Phương thức rjust

Cú pháp:

```
<chuỗi>.rjust(width, [fillchar])
```

Công dụng: Cách hoạt động tương tự như phương thức center, có điều là căn lề phải

```
>>> 'kteam'.rjust(12)
'       kteam'
>>> 'kteam'.rjust(12, '*')
'*******kteam'
```

Phương thức ljust

Cú pháp:

```
<chuỗi>.ljust(width, [fillchar])
```

Công dụng: Cách hoạt động tương tự phương thức center, nhưng căn lề trái.

```
>>> 'Kter'.ljust(12)
'Kter        '
>>> 'kter'.ljust(12, '*')
'kter********'
```

Các phương thức xử lí

Phương thức encode

Cú pháp:

```
<chuỗi>.encode(encoding='utf-8', errors='strict')
```

Công dụng: Đây là phương thức dùng để encode một chuỗi với phương thức mã hóa mặc định là utf-8. Còn về errors mặc định sẽ là strict có nghĩa là sẽ có thông báo lỗi hiện lên nếu có vấn đề xuất hiện trong quá trình encode chuỗi. Một số giá trị ngoài strict là ignore, replace, xmlcharrefreplace. Vì phần này là phần nâng cao, Kteam xin phép không đi sâu.

```
>>> 'ố ồ'.encode()
b'\xe1\xbb\x91 \xe1\xbb\x93'
```

Phương thức join

Cú pháp:

```
<kí tự nối>.join(<iterable>)
```

Công dụng: Trả về một chuỗi bằng cách nối các phần tử trong **iterable** bằng kí tự nối. Một iterable có thể là một tuple, list,... hoặc là một iterator (Kteam sẽ giải thích khái niệm này ở các bài sau).

- Một điểm lưu ý, các phần tử trong iterable buộc phải thuộc lớp str

```
>>> ' '.join(['1', '2', '3']) # iterable ở đây là list ['1', '2', '3']
'1 2 3'
>>> ' '.join(('1', '2', '3')) # iterable ở đây là tuple ('1', '2', '3')
'1 2 3'
>>> ' '.join([1, 2, 3]) # phần tử trong list không phải là chuỗi
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: sequence item 0: expected str instance, int found
```

Phương thức replace

Cú pháp:

```
<chuỗi>.replace(old, new, [count])
```

Công dụng: Trả về một chuỗi với các chuỗi **old** nằm trong chuỗi ban đầu được thay thế bằng chuỗi **new**. Nếu **count** khác None (có nghĩa là ta cho thêm count) thì ta sẽ thay thế old bằng new với số lượng count từ trái qua phải.

- Nếu chuỗi old không nằm trong chuỗi ban đầu hoặc count là 0 thì sẽ trả về một chuỗi giống với chuỗi ban đầu

```
>>> 'abc how abc kteam'.replace('abc', 'aaa')
'aaa how aaa kteam'
>>> 'abc how abc kteam'.replace('a', 'AA')
'AAbc how AAbc kteam'
>>> 'abc how abc kteam'.replace('abcd', 'AA')
'abc how abc kteam'
>>> 'abc how abc kteam'.replace('abc', 'AA', 1)
'AA how abc kteam'
>>> 'abc how abc kteam'.replace('abc', 'BB', 0)
'abc how abc kteam'
```

Phương thức strip

Phương thức này hơi rắc rối một tẹo nếu bạn chưa hiểu rõ cách nó hoạt động.

Cú pháp:

```
<chuỗi>.strip([chars])
```

Công dụng: Trả về một chuỗi với phần đầu và phần đuôi của chuỗi được bỏ đi các kí tự chars. Nếu chars bị bỏ trống thì mặc định các kí tự bị bỏ đi là dấu khoảng trắng và các escape sequence. Một số escape sequence ngoại lệ như **\a** sẽ được encode utf-8. Tuy vậy, không có ảnh hưởng gì tới nội dung.

```
>>> '  Kter   '.strip()
'Kter'
>>> '%%%%Kter%%%'.strip('%')
'Kter'
>>> 'cababHowbaaaca'.strip('abc')
'How'
>>> '\t\n\aKter\a\a\n\v'.strip() # các \a biến thành \x07
'\x07Kter\x07\x07'
>>> print('\x07Kter\x07\x07')  # nhưng khi dùng print vẫn có kết quả tương tự
```

Phương thức rstrip

Cú pháp:

```
<chuỗi>.rstrip()
```

Công dụng: Cách hoạt động hoàn toàn như phương thức strip, nhưng khác là chỉ bỏ đi ở phần đuôi (từ phải sang trái)

```
>>> '   Kter   '.rstrip()
'   Kter'
>>> '%%%%Share%%%'.rstrip('%')
'%%%%Share'
>>> 'cababKterbaaaca'.rstrip('abc')
'cababKter'
```

Phương thức lstrip

Cú pháp:

```
<chuỗi>.lstrip()
```

Công dụng: Cách hoạt động tương tự phương thức rstrip, khác ở chỗ rstrip lo phần đuôi, còn lstrip lo phần đầu (từ trái sang phải)

```
>>> '   Kter   '.lstrip()
'Kter   '
>>> '%%%%Kter%%%'.lstrip('%')
'Kter%%%'
>>> 'cababKterbaaaca'.lstrip('abc')
'Kterbaaaca'
```

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 KIỂU DỮ LIỆU CHUỖI TRONG PYTHON – Phần 3.

Nếu bạn rút gọn được từ 5 dòng trở xuống thì bạn đã giải được câu hỏi trên. Còn đây là cách rút gọn ngắn nhất

```
print('+ {:-<6} + {:-^15} + {:->10} +\n'.format('', '', '') + '| {:<6} | {:^15} | {:>10} |\n'.format('ID', 'Ho va ten', 'Noi sinh') + '| {:<6} | {:^15} | {:>10} |\n'.format('123', 'Yui Hatano', 'Japanese') + '| {:<6} | {:^15} | {:>10} |\n'.format('6969', 'Sunny Leone', 'Canada') + '+ {:-<6} + {:-^15} + {:->10} +'.format('', '', ''))
```

Kết luận

Qua bài viết này, bạn đã biết được một vài PHƯƠNG THỨC CHUỖI.

Ở bài viết sau, Kteam sẽ tiếp tục giới thiệu thêm một số phương thức của KIỂU DỮ LIỆU CHUỖI TRONG PYTHON (Phần 5)

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**".