Python cho Khoa học dữ liệu

Bài 8: Chuỗi ký tự - Strings

Hà Minh Tuấn
hmtuan@hcmus.edu.vn

Khoa Toán - Tin học
Trường Đại học Khoa học Tự nhiên, ĐHQG-HCM

Ngày 31 tháng 8 năm 2025

Nội dung bài học

Python String là gì?

- String = chuỗi ký tự Unicode, immutable.
- Được bao trong '', "", hoặc ''' '''.
- Ví dụ:

```
1234      # integer
'1234'    # string
```

Tạo String trong Python

```
'Welcome To TutorialsPoint'
"Welcome To TutorialsPoint"
'''Welcome To TutorialsPoint'''
"""Welcome To TutorialsPoint"""
```

```
'Welcome To TutorialsPoint'
```

- Python 3: mặc định là Unicode.
- Không cần prefix u như Python 2.

Truy cập và Cập nhật String

```
var1 = 'Hello World!'
var2 = "Python Programming"

print(var1[0])       # H
print(var2[1:5])     # ytho

print(var1[:6] + 'Python')
```

```
H
ytho
Hello Python
```

Escape Sequences in C

Ký hiệu	Giá trị	Mô tả
\a	0x07	Bell or alert
\b	0x08	Backspace
\cx		Control-x
\C-x		Control-x
\e	0x1b	Escape
\f	0x0c	Formfeed
\M-\C-x		Meta-Control-x
\n	0x0a	Newline
\nnn		Octal notation (n = 0–7)
\r	0x0d	Carriage return
\s	0x20	Space
\t	0x09	Tab
\v	0x0b	Vertical tab
\x		Character x
\xnn		Hexadecimal notation (n = 0–9, a–f, A–F)

String Operators

- `+`: nối chuỗi (`'Hello' + 'Python'`)
- `*`: lặp chuỗi (`'Hi' * 3`)
- `[]`, `[:]`: indexing & slicing
- `in / not in`: kiểm tra membership
- `r''`: raw string
- %: định dạng chuỗi

String Formatting với %

```
print("My name is %s and weight is %d kg!" % ('Zara', 21))
```

```
My name is Zara and weight is 21 kg!
```

- %s: string
- %d: integer
- %f: float
- %x / %X: hex

String Indexing trong Python

- Chuỗi trong Python là một dãy ký tự Unicode có thứ tự.
- Mỗi ký tự có một chỉ số (index) duy nhất.
- Chỉ số bắt đầu từ 0 ở bên trái và tăng dần.
- Nếu vượt quá chỉ số cuối, Python sẽ báo lỗi `IndexError`.

Ví dụ: String Indexing

```
var = "HELLO PYTHON"
print(var[0])
print(var[7])
print(var[11])
print(var[12])
```

```
H
Y
N
ERROR!
Traceback (most recent call last):
  ...
IndexError: string index out of range
```

Positive và Negative Indexing

- Python cho phép sử dụng chỉ số âm.
- Với chuỗi `"HELLO PYTHON"`:
  - Positive index: H=0, E=1, ..., N=11
  - Negative index: H=-12, E=-11, ..., N=-1
- Điều này giúp truy cập nhanh từ cuối chuỗi.

Ví dụ: Negative Indexing

```
var = "HELLO PYTHON"
print(var[-1])
print(var[-5])
print(var[-12])
```

```
N
Y
H
```

Chuỗi là kiểu dữ liệu bất biến (Immutable)

- Chuỗi trong Python là **immutable**.
- Nghĩa là không thể thay đổi trực tiếp ký tự trong chuỗi.

Ví dụ: String Immutability

```
var = "HELLO PYTHON"
var[7] = "y"
print(var)
```

```
Traceback (most recent call last):
  ...
TypeError: 'str' object does not support item assignment
```

String Slicing

- Toán tử : dùng để cắt chuỗi con.
- Cú pháp: `substr = var[x:y]`
- Trả về chuỗi từ chỉ số `x` đến `y-1`.

Ví dụ: String Slicing

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

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

Slicing với Negative Index

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

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

Giá trị mặc định trong Slicing

- Nếu bỏ qua x, mặc định là 0.
- Nếu bỏ qua y, mặc định đến hết chuỗi.
- Nếu bỏ qua cả hai, sẽ ra toàn bộ chuỗi gốc.

Ví dụ: Default Index trong Slicing

```
var = "HELLO PYTHON"
print(var[0:5])
print(var[:5])
print(var[6:12])
print(var[6:])
print(var[:])
```

```
HELLO
HELLO
PYTHON
PYTHON
HELLO PYTHON
```

Ví dụ: Trường hợp đặc biệt

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

```
(chuỗi rỗng)
(chuỗi rỗng)
```

Return Type của Slicing

```
var = "HELLO PYTHON"
print("var[:6][:2]:", var[:6][:2])

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

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

String Modification trong Python

- Chuỗi trong Python là **immutable** (bất biến).
- Không thể thay đổi trực tiếp ký tự trong chuỗi.
- Để "chỉnh sửa"chuỗi, cần tạo ra một chuỗi mới dựa trên chuỗi gốc.
- Một số cách phổ biến:
  1. Chuyển thành `list`, chỉnh sửa rồi nối lại.
  2. Sử dụng `array` module.
  3. Dùng lớp `StringIO` trong `io`.

Cách 1: Chuyển chuỗi thành List

```
s1 = "WORD"
print("original string:", s1)

l1 = list(s1)           # chuyển sang list
l1.insert(3, "L")       # chèn ký tự
print(l1)

s1 = ''.join(l1)        # ghép lại thành chuỗi
print("Modified string:", s1)
```

```
original string: WORD
['W', 'O', 'R', 'L', 'D']
Modified string: WORLD
```

Cách 2: Dùng array module

```
import array as ar

s1 = "WORD"
print("original string:", s1)

sar = ar.array('u', s1)    # tạo array kiểu Unicode
sar.insert(3, "L")         # chèn ký tự

s1 = sar.tounicode()       # chuyển về chuỗi
print("Modified string:", s1)
```

```
original string: WORD
Modified string: WORLD
```

Cách 3: Dùng StringIO

```
import io

s1 = "WORD"
print("original string:", s1)

sio = io.StringIO(s1)  # tạo buffer như file
sio.seek(3)            # dịch con trỏ đến vị trí 3
sio.write("LD")        # ghi đè
s1 = sio.getvalue()    # lấy chuỗi kết quả

print("Modified string:", s1)
```

```
original string: WORD
Modified string: WORLD
```

String Concatenation trong Python

- **Concatenation** = nối nhiều chuỗi thành một chuỗi mới.
- Các cách phổ biến:
  1. Dùng + để nối 2 chuỗi.
  2. Nối chuỗi với khoảng trắng.
  3. Dùng $*$ để lặp lại chuỗi.
  4. Kết hợp + và $*$.
- Ngoài + và $*$, không có toán tử số học nào khác dùng cho chuỗi.

Cách 1: Toán tử '+'

```
str1 = "Hello"
str2 = "World"

str3 = str1 + str2
print("String 3:", str3)
```

```
String 3: HelloWorld
```

Cách 2: Nối với khoảng trắng

```
str1 = "Hello"
str2 = "World"
blank = " "

str3 = str1 + blank + str2
print("String 3:", str3)
```

```
String 3: Hello World
```

Cách 3: Toán tử '*' (repetition)

```
newString = "Hello" * 3
print(newString)
```

```
HelloHelloHello
```

Cách 4: Kết hợp '+' và '*'

```
str1 = "Hello"
str2 = "World"

str3 = str1 + str2 * 3
str4 = (str1 + str2) * 3

print("String 3:", str3)
print("String 4:", str4)
```

```
String 3: HelloWorldWorldWorld
String 4: HelloWorldHelloWorldHelloWorld
```

String Formatting trong Python

- **String Formatting** = xây dựng chuỗi động bằng cách chèn giá trị biến/biểu thức.
- Nối chuỗi (+) không cho phép toán hạng không phải chuỗi.
- Python cung cấp 4 cách định dạng chuỗi:
  1. Toán tử %
  2. Phương thức `format()`
  3. f-string (formatted string literals)
  4. Lớp `Template` trong module `string`

Cách 1: Dùng % operator

```
name = "Hà Minh Tuấn"
print("Welcome to %s!" % name)
```

```
Welcome to Hà Minh Tuấn!
```

Cách 2: Dùng format() method

```
msg = "Welcome to {}"
print(msg.format("Hà Minh Tuấn"))
```

```
Welcome to Hà Minh Tuấn
```

Cách 3: Dùng f-string

```
item1_price = 2500
item2_price = 300
total = f'Total: {item1_price + item2_price}'
print(total)
```

```
Total: 2800
```

Cách 4: Dùng Template class

```python
from string import Template

# Định nghĩa template
msg = "Hello and Welcome to $ name !"

# Tạo đối tượng Template
templateObj = Template(msg)

# Thay thế giá trị
new_msg = templateObj.substitute(name="Tutorialspoint")
print(new_msg)
```

```
Hello and Welcome to Tutorialspoint !
```

Escape Character trong Python

- Escape character là một ký tự đặc biệt đi kèm dấu gạch chéo ngược (\).
- Nó cho trình thông dịch biết rằng ký tự này có ý nghĩa đặc biệt.
- Ví dụ: `\n` đại diện cho ký tự xuống dòng (newline).
- Nếu thêm tiền tố `r` hoặc `R` trước chuỗi thì Python hiểu đó là raw string, tức là không xử lý escape sequence.
- Ví dụ: `'Hello'` là chuỗi bình thường, còn `r'Hello'` là raw string.

Ví dụ: Chuỗi thường và Raw String

```python
# normal string
normal = "Hello"
print(normal)

# raw string
raw = r"Hello"
print(raw)
```

```
Hello
Hello
```

Ví dụ: Xử lý Escape Sequence

```
normal = "Hello\nWorld"
print(normal)

raw = r"Hello\nWorld"
print(raw)
```

```
Hello
World
Hello\nWorld
```

Case Conversion Methods

- `capitalize()` : Viết hoa chữ cái đầu tiên.
- `casefold()` : Chuyển toàn bộ về thường (Unicode-aware).
- `lower()` : Chuyển toàn bộ về thường.
- `swapcase()` : Đảo ngược hoa thường.
- `title()` : Viết hoa chữ cái đầu mỗi từ.
- `upper()` : Viết hoa toàn bộ.

Ví dụ: Case Conversion

```
s = "hello World"
print(s.capitalize())
print(s.upper())
print(s.lower())
print(s.swapcase())
print(s.title())
```

```
Hello world
HELLO WORLD
hello world
HELLO wORLD
Hello World
```

Alignment Methods

- `center(width, fillchar)` : Căn giữa.
- `ljust(width[, fillchar])` : Căn trái.
- `rjust(width[, fillchar])` : Căn phải.
- `expandtabs(tabsize)` : Chuyển `\t` thành nhiều khoảng trắng.
- `zfill(width)` : Thêm số 0 phía trước.

Ví dụ: Alignment

```
s = "42"
print(s.center(6, "*"))
print(s.ljust(6, "-"))
print(s.rjust(6, "."))
print("A\tB".expandtabs(4))
print(s.zfill(5))
```

```
**42**
42----
....42
A   B
00042
```

Split and Join Methods

- `strip()`, `lstrip()`, `rstrip()` : Xoá khoảng trắng.
- `split()`, `rsplit()`, `splitlines()` : Tách chuỗi.
- `partition()`, `rpartition()` : Tách thành bộ 3.
- `join()` : Nối chuỗi từ danh sách.
- `removeprefix()`, `removesuffix()` : Xoá tiền tố/hậu tố.

Ví dụ: Split và Join

```
s = "  hello world  "
print(s.strip())
print("a,b,c".split(","))
print("line1\nline2".splitlines())
print("-".join(["2025","08","28"]))
print("unittest".removeprefix("unit"))
```

```
hello world
['a', 'b', 'c']
['line1', 'line2']
2025-08-28
test
```

Boolean String Methods

- `isalnum()`, `isalpha()`, `isdigit()`, `isnumeric()`, `isdecimal()`
- `islower()`, `isupper()`, `istitle()`
- `isspace()`, `isascii()`, `isidentifier()`, `isprintable()`

Ví dụ: Boolean Methods

```
print("abc123".isalnum())
print("abc".isalpha())
print("123".isdigit())
print("hello".islower())
print("HELLO".isupper())
```

```
True
True
True
True
True
```

Find and Replace Methods

- `count()`, `find()`, `rfind()`
- `index()`, `rindex()`
- `replace(old,new[,max])`
- `startswith()`, `endswith()`

Ví dụ: Find và Replace

```
s = "banana"
print(s.count("a"))
print(s.find("na"))
print(s.replace("na","NA",1))
print(s.startswith("ba"))
print(s.endswith("na"))
```

```
3
2
baNAna
True
True
```

Translation Methods

- `maketrans()` : Tạo bảng dịch.
- `translate()` : Thay thế ký tự theo bảng dịch.

Ví dụ: Translation

```
table = str.maketrans("aeiou", "12345")
s = "hello world"
print(s.translate(table))
```

```
h2ll4 w4rld
```