

BÀI GIẢNG  
CƠ SỞ LẬP TRÌNH

CHƯƠNG 7.

XỬ LÝ DỮ LIỆU KIỂU CHUỖI

---

NGUYỄN THÀNH THỦY

BỘ MÔN TIN HỌC QUẢN LÝ

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THUYNT@DUE.EDU.VN

# NỘI DUNG

- Làm việc với chuỗi
- Cấu trúc dữ liệu chuỗi
- Toán tử và phương thức xử lý chuỗi

# LÀM VIỆC VỚI CHUỖI

## ❑ Biểu diễn chuỗi ký tự

- Xử dụng cặp dấu nháy đơn `'` hoặc nháy kép `"` để biểu diễn một chuỗi ký tự
- Ví dụ:

```
>>> str1='My name is Đen Vâu'  
>>> str2="My name is Đen Vâu"  
>>> str3="My name's Đen Vâu"  
>>> str4='My name's Đen Vâu'  
File "<stdin>", line 1  
    str4='My name's Đen Vâu'  
            ^  
SyntaxError: invalid syntax
```

Chuỗi chứa dấu `'` thì được bao bởi cặp dấu nháy kép `"`

# LÀM VIỆC VỚI CHUỖI

## ❑ Hiển thị ký tự đặc biệt với dấu \

Escape character	Prints as
\'	Single quote
\"	Double quote
\t	Tab
\n	Newline (line break)
\\	Backslash

# LÀM VIỆC VỚI CHUỖI

## ❑ Hiển thị ký tự đặc biệt với dấu \

### ■ Ví dụ:

```
>>> print("My name's Đen Vâu")
My name's Đen Vâu
>>> print('My name\'s Đen Vâu')
My name's Đen Vâu
>>> print("Hello there!\nHow are you?\nI\'m doing fine.")
Hello there!
How are you?
I'm doing fine.
```

# LÀM VIỆC VỚI CHUỖI

## ❑ Hiển thị ký tự đặc biệt với dấu \

- Ví dụ: sử dụng 1 lệnh print để hiển thị lên màn hình câu thông báo sau:

**"Đen Vâu" is a rap singer  
He's 32 years old  
I want to become a rapper**



# LÀM VIỆC VỚI CHUỖI

## ❑ Định nghĩa chuỗi dữ liệu thô (raw string)

- Đặt ký tự **r** trước dấu “ hoặc ‘ bắt đầu chuỗi, để giữ lại các ký tự đặc biệt trong chuỗi.
- Ví dụ:

```
>>> print('My name\'s Đen Vâu')
My name's Đen Vâu
>>> print(r'My name\'s Đen Vâu')
My name\'s Đen Vâu
>>> print("Hello there!\nHow are you?\nI\'m doing fine.")
Hello there!
How are you?
I'm doing fine.
>>> print(r"Hello there!\nHow are you?\nI\'m doing fine.")
Hello there!\nHow are you?\nI\'m doing fine.
```

# LÀM VIỆC VỚI CHUỖI

## ❑ Biểu diễn chuỗi gồm nhiều dòng văn bản

- Sử dụng 3 dấu nháy đơn ''' bắt đầu và kết thúc khối văn bản để biểu diễn chuỗi gồm nhiều dòng.
- Ví dụ:

```
st="Đường về nhà là vào tim ta  
Dầu nắng mưa gần xa  
Thất bát, vang danh  
Nhà vẫn luôn chờ ta"  
print(st)
```



```
Đường về nhà là vào tim ta  
Dầu nắng mưa gần xa  
Thất bát, vang danh  
Nhà vẫn luôn chờ ta
```



# CẤU TRÚC DỮ LIỆU CHUỖI

## ❑ Số chỉ mục Indexing

- Chuỗi được tổ chức tương tự cấu trúc dữ liệu kiểu danh sách (List);
- Mỗi phần tử của list là một ký tự trong chuỗi, được biểu diễn bằng một số chỉ mục bắt đầu bằng số 0 (zero);
- Ví dụ:

Element	P	y	t	h	o	n
Index	0	1	2	3	4	5

# CẤU TRÚC DỮ LIỆU CHUỖI

## ❑ Truy cập phần tử bằng cửa sổ trượt (*Slicing*)

### ■ Ví dụ:


```
>>> spam = 'Hello world!'
>>> spam[0]
'H'
>>> spam[4]
'o'
>>> spam[-1]
'!'
>>> spam[0:5]
'Hello'
>>> spam[:5]
'Hello'
>>> spam[6:]
'world!'
```

# CẤU TRÚC DỮ LIỆU CHUỖI

## ❑ Truy cập phần tử bằng cửa sổ trượt (*Slicing*)

### ■ Ví dụ:

```
str="Python Programming"
str1=str[:6]
❶ print(str1)
str2=str[7:]
❷ print(str2)
str3=str2[0:3]
❸ print(str3)
str4=str1[-4:]
❹ print(str4)
str5=str[:2] + str[-4:]
❺ print(str5)
```



# CẤU TRÚC DỮ LIỆU CHUỖI

## ❑ Truy cập phần tử bằng cấu trúc lặp


### ■ Cấu trúc while

```
fruit = 'banana'
index = 0
while index < len(fruit):
    letter = fruit[index]
    print(letter)
    index = index + 1
```

### ■ Cấu trúc for

```
fruit = 'banana'
for char in fruit:
    print(char)
```

b  
a  
n  
a  
n  
a



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Toán tử **in** và **not in**

- Kiểm tra một chuỗi nằm trong một chuỗi khác hay không, kết quả trả về **true** hoặc **false**;
- Ví dụ:

```
>>> 'Hello' in 'Hello World'
True
>>> 'Hello' in 'Hello'
True
>>> 'HELLO' in 'Hello World'
False
>>> '' in 'spam'
True
>>> 'cats' not in 'cats and dogs'
False
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Toán tử **in** và **not in**

### ■ Ví dụ:

```
str="Python Programming"
```

```
str1=str[:6]
```

```
str2=str[7:]
```

```
❶ print(str1 in str)
```

```
❷ print(str2 not in str)
```

```
str3="PYTHON"
```

```
❸ print(str3 not in str)
```

```
❹ print(" " in str)
```



❶

❷

❸

❹



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Toán tử **in** và **not in**

■ **Bài tập:** Viết chương trình:

- Nhập vào một chuỗi ký tự st1;
- Nhập vào 1 chuỗi st2, kiểm tra nếu st2 xuất hiện trong st1 thì yêu cầu nhập lại chuỗi khác, còn lại thì dừng.

**Ví dụ:**

Python Programming

Python

Pro

thon

abc

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Toán tử so sánh `==`, `<`, `>`

- Ví dụ: so sánh bằng `==`

```
if word == 'banana':  
    print('All right, bananas.')
```

- Ví dụ: so sánh `<` và `>`

```
if word < 'banana':  
    print('Your word,' + word + ', comes before banana.')
```

```
elif word > 'banana':  
    print('Your word,' + word + ', comes after banana.')
```

```
else:  
    print('All right, bananas.')
```

- **Lưu ý:** chuỗi được so sánh dựa trên thứ tự các ký tự trong chuỗi tính từ trái sang phải. Chữ hoa thứ tự đứng trước chữ thường.



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `upper()`, `lower()`

- Trả về một chuỗi được viết hoa hoặc viết thường;
- Ví dụ:

```
>>> spam = 'Hello world!'
>>> spam = spam.upper()
>>> spam
'HELLO WORLD!'
>>> spam = spam.lower()
>>> spam
'hello world!'
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `upper()`, `lower()`

### ■ Ví dụ:

```
print('How are you?')  
feeling = input()  
if feeling.lower() == 'great':  
    print('I feel great too.')  
else:  
    print('I hope the rest of your day is good.')
```



How are you?

*GREAT*

???



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `isupper()`, `islower()`

- Trả về **true** hoặc **false** nếu một chuỗi (chữ cái) có được viết hoa hay viết thường toàn bộ xâu;
- Ví dụ:

```
>>> spam = 'Hello world!'
>>> spam.islower()
False
>>> spam.isupper()
False
>>> 'HELLO'.isupper()
True
>>> 'abc12345'.islower()
True
>>> '12345'.islower()
False
>>> '12345'.isupper()
False
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `isupper()`, `islower()`

- **Bài tập:** Viết chương trình nhập vào một xâu str và một ký tự ch, in lên màn hình số lượng ký tự ch xuất hiện trong xâu str (không phân biệt chữ hoa và chữ thường).

**Ví dụ:**

```
str=Learn PYTHON Programming
```

```
ch=n
```

```
Number of character n is: 3
```



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `isalpha()`

- `isalpha()`: trả về **True** nếu chuỗi chỉ chứa các **ký tự chữ cái** và **không có ký tự trắng**, còn lại trả về **False**;
- Ví dụ:

```
>>> str='1'
>>> str.isalpha()
False
>>> str='123'
>>> str.isalpha()
False
>>> str='a'
>>> str.isalpha()
True
>>> str='abc'
>>> str.isalpha()
True
>>> str='a b c'
>>> str.isalpha()
False
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `isnumeric()`, `isdecimal()`

- Trả về **True** nếu chuỗi chỉ chứa các **ký tự chữ số** và **không có ký tự trắng**, còn lại trả về **False**;
- Ví dụ:

```
>>> str='1'
>>> str.isnumeric()
True
>>> str='123'
>>> str.isnumeric()
True
>>> str='a'
>>> str.isnumeric()
False
>>> str='abc'
>>> str.isnumeric()
False
>>> str='123abc'
>>> str.isnumeric()
False
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

- **Ví dụ:** Chương trình nhập vào một số nguyên là tuổi một sinh viên, chương trình kiểm tra và **yêu cầu nhập lại nếu dữ liệu không phải là số.**

```
while True:
    print('Enter your age:')
    age = input()
    if age.isdecimal():
        break
    print('Please enter a number for your age.')
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

- ❑ **Bài tập:** Viết chương trình nhập từ bàn phím 2 số nguyên a và b.
  - Yêu cầu kiểm tra tính đúng của dữ liệu khi nhập vào. Nếu dữ liệu không hợp lệ thì yêu cầu nhập lại.
  - In lên màn hình kết quả của biểu thức  $(a+b)$ .





# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `isalnum()`

- `isalnum()`: trả về **True** nếu chuỗi chỉ chứa các **ký tự chữ cái** hoặc **chữ số** và **không có ký tự trắng**, còn lại trả về **False**;

- Ví dụ:

```
>>> str='1'
>>> str.isalnum()
True
>>> str='123'
>>> str.isalnum()
True
>>> str='a'
>>> str.isalnum()
True
>>> str='abc'
>>> str.isalnum()
True
>>> str='a b c'
>>> str.isalnum()
False
>>> str='123abc'
>>> str.isalnum()
True
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

- **Ví dụ:** Chương trình nhập vào mật khẩu là một chuỗi ký tự chỉ bao gồm **chữ cái và chữ số**, chương trình kiểm tra và **yêu cầu nhập lại cho đến khi hợp lệ**.

```
while True:
    print('Select a new password (letters and numbers only):')
    password = input()
    if password.isalnum():
        break
    print('Passwords can only have letters and numbers.')
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `isspace()`

- `isspace()`: trả về **True** nếu chuỗi chỉ chứa các **ký tự trắng**, hoặc **dấu tab**, hoặc **dấu ngắt dòng**, còn lại trả về **False**;
- Ví dụ:

```
>>> str=' '  
>>> str.isspace()  
True  
>>> str='\n'  
>>> str.isspace()  
True  
>>> str='\t'  
>>> str.isspace()  
True  
>>> str='123'  
>>> str.isspace()  
False  
>>> str='abc'  
>>> str.isspace()  
False  
>>> str=''   
>>> str.isspace()  
False
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức **istitle()**

- **istitle()**: trả về **True** nếu chuỗi chỉ chứa các từ, mỗi từ được viết thường và viết hoa chữ cái đầu, còn lại trả về **False**;
- Ví dụ:

```
>>> str='Learn Python'
>>> str.istitle()
True
>>> str='LEARN Python'
>>> str.istitle()
False
>>> str='Learn python'
>>> str.istitle()
False
>>> str='Learn Python 1'
>>> str.istitle()
True
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `istitle()`

**Bài tập:** Viết chương trình nhập vào họ tên của một sinh viên. Kiểm tra tính hợp lệ của chuỗi nhập vào. Biết rằng họ tên hợp lệ nếu tất cả được viết thường, chỉ viết hoa chữ cái đầu của từ.

### Ví dụ 1:

Ho ten: NGUYEN VAN AN

Khong hop le!!!

### Ví dụ 2:

Ho ten: Nguyen Van An

Hop le!!!

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

**Bài tập:** Nhập vào một chuỗi ký tự làm mật khẩu cho chương trình.

**Yêu cầu:**

- Mật khẩu có tối thiểu 8 ký tự;
- Bao gồm: chữ cái, chữ số, chữ viết hoa và viết thường.

Nếu chuỗi mật khẩu không hợp lệ thì yêu cầu nhập lại.

**Ví dụ:**

abc123	---> Không hợp lệ
aaaaa111111	---> Không hợp lệ
aaaaBBBBBB	---> Không hợp lệ
1111111111	---> Không hợp lệ
AAAAA111111	---> Không hợp lệ
aaaAA111111	---> Hợp lệ



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức **startswith(str)**, **endswith(str)**

- **startswith(str)**: Trả về **True** nếu chuỗi bắt đầu bởi chuỗi str;
- **endswith(str)**: Trả về **True** nếu chuỗi kết thúc bởi chuỗi str, còn lại trả về **False**;

Có phân biệt chữ hoa và chữ thường

### ■ Ví dụ:

```
>>> 'Hello world!'.startswith('Hello')
True
>>> 'Hello world!'.endswith('world!')
True
>>> 'abc123'.startswith('abcdef')
False
>>> 'abc123'.endswith('12')
False
>>> 'Hello world!'.startswith('Hello world!')
True
>>> 'Hello world!'.endswith('Hello world!')
True
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức **startswith(str)**, **endswith(str)**

- Ví dụ: Viết hàm **checkstring(st1,st2,st3)** kiểm tra chuỗi **st1** là hợp lệ nếu bắt đầu bằng **st2** hoặc kết thúc bằng **st3**. Hợp lệ thì hàm trả về **True**, còn lại trả về **False**.

### TEST 1:

```
st1="Python Programming"  
st2="Py"  
st3="ming"  
--> True
```

### TEST 2:

```
st1="Python Programming"  
st2="PY"  
st3="MING"  
--> False
```





# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `join(ListOfString)`

- **`join(ListOfString)`**: `ListOfString` là một List gồm các chuỗi ký tự. `join()` dùng để nối các phần tử trong `ListOfString` bằng một chuỗi tương ứng của phương thức;
- Ví dụ:

```
>>> ', '.join(['cats', 'rats', 'bats'])
'cats, rats, bats'
>>> ' '.join(['My', 'name', 'is', 'Simon'])
'My name is Simon'
>>> 'ABC'.join(['My', 'name', 'is', 'Simon'])
'MyABCnameABCisABCSimon'
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `split()`

- `split(str)`: dùng để tách mỗi từ nằm trong chuỗi tương ứng thành một list, mỗi phần tử nằm trong list là một chuỗi con;
- Ví dụ:

```
>>> 'My name is Simon'.split()  
['My', 'name', 'is', 'Simon']
```

Không có tham số str

```
>>> 'MyABCnameABCisABCSimon'.split('ABC')  
['My', 'name', 'is', 'Simon']
```

```
>>> 'My name is Simon'.split('m')  
['My na', 'e is Si', 'on']
```

Có tham số str

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

- **Ví dụ:** sử dụng phương thức **split()** để tách mỗi dòng trong một đoạn văn bản thành các phần tử của một List.

```
>>> spam = '''Dear Alice,  
How have you been? I am fine.  
There is a container in the fridge  
that is labeled "Milk Experiment".
```

```
Please do not drink it.  
Sincerely,  
Bob'''
```

Sử dụng ký tự ngắt dòng

```
>>> spam.split('\n')  
['Dear Alice,', 'How have you been? I am fine.', 'There is a container in the  
fridge', 'that is labeled "Milk Experiment".', '', 'Please do not drink it.',  
'Sincerely,', 'Bob']
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

- **Bài tập:** Cho đoạn văn bản như sau:

*--Người---đâu-gặp---gỡ-làm-chi---  
Trăm----năm-biết-có---duyên---gì--hay--không.  
Ngổn-ngang---trăm-mỗi---bên---lòng----  
----Nên-câu---tuyệt--diệu-ngụ-trong-tính-tình."*

Yêu cầu tiền xử lý để thành đoạn văn bản có ý nghĩa.

*"Người đâu gặp gỡ làm chi  
Trăm năm biết có duyên gì hay không.  
Ngổn ngang trăm mối bên lòng  
Nên câu tuyệt diệu ngụ trong tính tình."*



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `rjust(n,ch)`, `ljust(n,ch)`, `center(n,ch)`

- Trả về một chuỗi mới khi thêm vào chuỗi gốc các ký tự **ch**, sao cho chiều dài của chuỗi đúng bằng n ký tự và chuỗi gốc được đặt nằm bên phải (**`rjust()`**), bên trái (**`ljust()`**) hoặc ở giữa (**`center()`**);
- Ví dụ: trường hợp không có tham số **ch**

```
>>> 'Hello'.rjust(10)
'      Hello'
```

Thêm 5 ký tự trắng vào bên trái chuỗi gốc

```
>>> 'Hello'.rjust(20)
'                Hello'
```

```
>>> 'Hello World'.rjust(20)
'                Hello World'
```

```
>>> 'Hello'.ljust(10)
'Hello      '
```

Thêm 5 ký tự trắng vào bên phải chuỗi gốc

- Lưu ý: không làm thay đổi giá trị trên chuỗi gốc

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `rjust(n,ch)`, `ljust(n,ch)`, `center(n,ch)`

- Trả về một chuỗi mới khi thêm vào chuỗi gốc các ký tự **ch**, sao cho chiều dài của chuỗi đúng bằng **n** ký tự và chuỗi gốc được đặt nằm bên phải (**`rjust()`**), bên trái (**`ljust()`**) hoặc ở giữa (**`center()`**);
- Ví dụ: trường hợp có tham số **ch**

```
>>> 'Hello'.rjust(20, '*')
'*****Hello'
>>> 'Hello'.ljust(20, '-')
'Hello-----'
```

Thêm 15 dấu \* vào bên  
trái chuỗi gốc

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `rjust(n,ch)`, `ljust(n,ch)`, `center(n,ch)`

- Trả về một chuỗi mới khi thêm vào chuỗi gốc các ký tự **ch**, sao cho chiều dài của chuỗi đúng bằng **n** ký tự và chuỗi gốc được đặt nằm bên phải (**`rjust()`**), bên trái (**`ljust()`**) hoặc ở giữa (**`center()`**);
- Ví dụ:

```
>>> 'Hello'.center(20)
'          Hello          '
>>> 'Hello'.center(20, '=')
'=====Hello====='
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `rjust(n,ch)`, `ljust(n,ch)`, `center(n,ch)`

- **Ví dụ:** Viết chương trình nhập vào tên và số lượng của 4 một mặt hàng bất kỳ, in lên màn hình bảng thực đơn theo mẫu sau.

sandwiches.....	4
apples.....	12
cups.....	4
cookies.....	8000





# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `strip(str)`, `rstrip(str)`, and `lstrip(str)`

- Trả về một chuỗi mới trong đó đã xóa các ký tự có trong chuỗi `str` ở 2 đầu (`strip()`), bên phải (`rstrip()`) hoặc bên trái (`lstrip()`) chuỗi gốc.
- **Ví dụ:** trường hợp **không** có tham số `str`

```
>>> spam = '    Hello World    '
```

```
>>> spam.strip()
```

```
'Hello World'
```

Xóa các ký tự trắng ở 2 đầu chuỗi spam

```
>>> spam.lstrip()
```

```
'Hello World    '
```

Xóa các ký tự trắng ở bên trái chuỗi spam

```
>>> spam.rstrip()
```

```
'    Hello World'
```

Xóa các ký tự trắng ở bên phải chuỗi spam

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `strip(str)`, `rstrip(str)`, and `lstrip(str)`

- Trả về một chuỗi mới trong đó đã xóa các ký tự có trong chuỗi `str` ở 2 đầu (`strip()`), bên phải (`rstrip()`) hoặc bên trái (`lstrip()`) chuỗi gốc.
- **Ví dụ:** trường hợp có tham số `str`

```
>>> spam = 'SpamSpamBaconSpamEggsSpamSpam'
>>> spam.strip('ampS')
'BaconSpamEggs'
```

Xóa các ký tự 'a', 'm', 'p', 'S' ở 2 đầu chuỗi spam

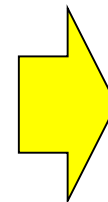
- **Lưu ý:**
  - Có phân biệt chữ hoa và thường
  - Không làm thay đổi giá trị của chuỗi gốc

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `strip(str)`, `rstrip(str)`, and `lstrip(str)`

### ■ Ví dụ:

```
>>> st='AbbaTheAbbaSongsAbab'  
>>> st.rstrip('ab')  
(1) ???  
>>> st.lstrip('ab')  
(2) ???  
>>> st.strip('Aab')  
(3) ???
```



(1) ???  
(2) ???  
(3) ???



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `find(str,n)`

- Thực hiện tìm chuỗi `str` trong chuỗi gốc, bắt đầu từ ký tự có số chỉ mục `n`, `n` để trống thì mặc định bằng 0.
- Trả về vị trí (index) lần đầu được tìm thấy (từ trái sang phải). Nếu không tìm thấy thì trả về -1.
- Ví dụ:

```
>>> word = 'banana'
>>> word.find('a')
1
>>> word.find('na')
2
>>> word.find('na', 3)
4
>>> word.find('A')
-1
```

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `find(str,n)`

- Ví dụ: tách domain xuất hiện trong một chuỗi văn bản.

```
>>> data = 'From stephen.marquard@uct.ac.za Sat Jan 5 09:14:16 2008'
>>> atpos = data.find('@')
>>> print(atpos)
21
>>> spos = data.find(' ',atpos)
>>> print(spos)
31
>>> host = data[atpos+1:spos]
>>> print(host)
uct.ac.za
>>>
```

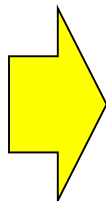
**Đọc thêm:** <https://docs.python.org/3/library/stdtypes.html#string-methods>

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Phương thức `replace(oldvalue, newvalue, n)`

- Tìm và thay thế các chuỗi `oldvalue` xuất hiện trong chuỗi gốc bằng `newvalue`, với `n` lần tìm và thay thế;
- Nếu không có `n`: thì mặc định là tất cả các lần xuất hiện.
- Ví dụ:

```
string = "geeks for geeks geeks geeks geeks"  
print(string.replace("geeks", "Geeks"))  
print(string.replace("geeks", "GeeksforGeeks", 3))
```



Geeks for Geeks Geeks Geeks Geeks  
GeeksforGeeks for GeeksforGeeks GeeksforGeeks geeks geeks

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

❑ Phương thức **replace**(oldvalue, newvalue, n)

**Bài tập:**

- Nhập vào 3 chuỗi st1, st2, st3;
- In lên màn hình vị trí (index) đầu tiên st2 xuất hiện trong st1;
- Thay thế tất cả các chuỗi st2 bằng st3;
- In lên màn hình chuỗi kết quả.

**Ví dụ:**

st1=Toi hoc python, python la NNLT manh

st2=python

st3=Python

Vị trí xuất hiện st2: 8

Xâu kết quả: Toi hoc Python, Python la NNLT manh

# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

## ❑ Định dạng %

- Dùng để nối chuỗi theo phương pháp tham số.

- Các loại định dạng:

- %d: dữ liệu kiểu số nguyên (decimal);
- %g: dữ liệu kiểu số thực;
- %s: dữ liệu kiểu chuỗi (string);

- Ví dụ:

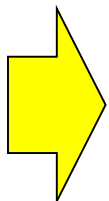
```
n_decimal=3
```

```
n_float=2.5
```

```
st="Python"
```

```
print("Toi da hoc " + st + " lan thu " + str(n_decimal) + ", trong " + str(n_float) + " nam")
```

```
print("Toi da hoc %s lan thu %d, trong %g nam" % (st,n_decimal,n_float))
```



Toi da hoc Python lan thu 3, trong 2.5 nam

Toi da hoc Python lan thu 3, trong 2.5 nam

**Đọc thêm:** <https://docs.python.org/3/library/stdtypes.html#printf-style-string-formatting>



# TOÁN TỬ VÀ PHƯƠNG THỨC XỬ LÝ CHUỖI

Input	Method	Output
'hello WORLD'	.capitalize()	Hello world
'HELLO WORLD'	.lower()	hello world
'hello world'	.upper()	HELLO WORLD
'Python'	.center(10, '*')	**Python**
'HELLO WORLD'	.count('L')	3
'HELLO WORLD'	.index('O')	4
'HELLO WORLD'	.find('OR')	7
'14/09/2022'	.replace('/', '-')	14-09-2022
'14/09/2022'	.split('/')	['14', '09', '2022']
'abc123'	.isalnum()	True
'12345'	.isnumeric()	True
'hello world'	.islower()	True
'HELLO WORLD'	.isupper()	True

# ÔN TẬP

**Bài 7.1** Cho biết kết quả khi thực hiện các đoạn code sau:

- a. `'Hello world!'[1]`
- b. `'Hello world!'[0:5]`
- c. `'Hello world!':5]`
- d. `'Hello world!'[3:]`

**Bài 7.2** Cho biết kết quả khi thực hiện các đoạn code sau:

- a. `'Hello'.upper()`
- b. `'Hello'.upper().isupper()`
- c. `'Hello'.upper().lower()`

**Bài 7.3** Cho biết kết quả khi thực hiện các đoạn code sau:

- a. `'Remember, remember, the fifth of November.'.split()`
- b. `'-'.join('There can be only one.'.split())`

# ÔN TẬP

**Bài 7.4** Xây dựng một hàm có đầu vào là một chuỗi bất kỳ, đầu ra là 2 chuỗi, trong đó chuỗi 1 được viết thường toàn xâu, chuỗi 2 được viết hoa toàn xâu.

**Input:**

Learning Python

**Output:**

learning python

LEARNING PYTHON

**Bài 7.5** Viết chương trình cho phép nhập vào một chuỗi bất kỳ, đếm số chữ cái và chữ số trong câu đó và in kết quả ra màn hình.

**Input:**

Python Programming Class 2021

**Output:**

Chu cai: 22

Chu so: 4