Python List - Kiểu List trong Python

List trong Python là cấu trúc mảng và các phần tử có index có thứ tự. Không như PHP, key của một mảng có thể vừa
là số, vừa là chuỗi (associated array).

Trong Python, muốn tạo một mảng có key là chuỗi thì sẽ sử dụng cấu trúc Dictionary (phần tiếp). Trong phần
này, chúng ta sẽ nói đến List. Một List được khai báo như mảng trong .JSON. Sử dụng [...] để khai báo một mảng.

Ví dụ:

```
numbers = [1, , , , ]


names = ['Marry', 'Peter']
```

Có thể truy xuất từng phần tử của mảng bằng index, phần tử đầu tiên có thứ tự là `0`

Ví dụ:

```
Print numbers[0]
(Hiển thị 1)


Print numbers[-3]
(Hiển thị 3)


Print names[1]
(Hiển thị 'Peter')
```

Để biết được số lượng phần tử của 1 List, có thể sử dụng hàm `len(array)` để lấy số lượng phần tử của mảng tham
số truyền vào.

1. Kiểm tra sự tồn tại của một phần tử

a. Kiểm tra theo index

Trong nhiều trường hợp bạn muốn truy xuất một phần tử bất kỳ (dựa vào index) của mảng thì nếu truy xuất đến một
phần tử không tồn tại thì ứng dụng sẽ báo lỗi. Do đó, trước khi truy xuất một phần tử, bạn cần kiểm tra xem phần tử này đã tồn tại hay chưa. Hiện tại python không hỗ trợ hàm nào để kiểm tra sự tồn tại của một phần tử trong mảng.

Có 2 cách thường thấy để kiểm tra đó là "Look before you leap" (LBYL) và "Easier to ask forgiveness than
permission" (EAFP).

Ví dụ về "Look before you leap (LBYL)":

```
1£ index < len(array):
□array[index]

else:
□# handle this
```

Ví dụ về "Easier to ask forgiveness than permission" (EAFP):

```
try:
□array[Lindex]

except IndexError:
□# handle this
```

b. Kiểm tra theo giá trị

Để kiểm tra một giá trị có tồn tại / không tồn tại trong mảng hay không thì có thể sử dụng toán tử `in` / `not in`.

Ví dụ:

```
mylist = ['a', 'b', 'c']

Print 'a' in mylist
(Hi□n thị True)

Print 'P' not 1n mylist
(Hi□n thị False)
```

2. Trích xuất mảng con

Tương tự như chuỗi, tó thể tạo các mảng con thông qua toán tử lấy khoản `[start: end]` (range). Mặc định start là
từ vị trí đầu chuỗi `0` và `end` là đến vị trí cuối chuỗi.

Ví dụ:

```
numbers = ['a', 'b', 'c', 'd']

Print numbers[:2]
(Hiển thị ['a', 'b'])

Print numbers[-2:]
(Hiển thị ['c', 'd'])
```

3. Xóa phần tử của mảng

Có thể xóa một phần tử thông qua toán tử `del`. Thứ tự của các phần tử sẽ dịch chuyển tùy vào vị trí của phần tử bị
xóa.

Ví dụ:

```
numbers = [1, 2, 3, 4, 5]
del numbers[0]

Print numbers

(Hiển thị [2, 3, 4, 5)
```

Bạn có thể xóa một khoản dựa vào toán tử lấy khoảng [start:end]

Ví dụ:

```
numbers = |1, 2, 3, 4, 5, 6, 7]
del numbers[2:4]

Print numbers

(Hiển thị [1, 2, 5, 6, 7])
```

4. Nối 2 mảng

Bạn có thể sử dụng toán tử + để nối giá trị của 2 mảng và tạo ra một mảng lớn có số lượng phần tử là tổng số lượng
phần tử của 2 mảng con.

Ví dụ:

```
a = [1,2]
b = [1,3]

Print a + b
(Hiển thị [1, 2, 1, 3])
```

5. Thêm phần tử vào mảng

Nếu bạn muốn thêm phần tử vào một mảng đã tồn tại, hãy dùng phương thức `list.append (newvalue)` để thêm phần
tử có giá trị `newvalue` vào cuối mảng `list`.

Ví dụ:

```
numbers = [1,2,3]
numbers.append(4)

Print numbers

(Hiển thị [1, 2, 3, 4])
```

6. Lấy phần tử cuối mảng

Nếu muốn lấy phần tử cuối cùng của mảng ra khỏi mảng, có thể sử dụng phương thức `list.pop()`, sẽ trả về giá trị
của phần tử cuối cùng và mảng bây giờ sẽ không còn phần tử này.

Ví dụ:

```
numbers = [1,2,3]

mynumber = numbers.pop()

Print mynumber

(Hiển thị 3)
```

```
Print numbers
(Hiển thị [1, 2])
```

7. Tìm một giá trị trong mảng

Nếu bạn muốn tìm vị trí (index) của một giá trị trong một mảng, có thể dùng phương thức `list.index(obj)`. Nếu
tìm thấy sẽ trả về index của phần tử đầu tiên tìm thấy. Nếu không tìm thấy sẽ có Exception.

Ví dụ:

```
aList = [123, 'xyz', 'zara', 'abc'];
Print "Index for xyz : ", aLlist.index('xyz')
Print "Index for zara : ", alist.index('zara')

Khi chạy sẽ hiển thị kết quả:

Index for xyz : 1

Index for zara : 2
```

8. Đảo ngược giá trị của mảng

Để đảo ngược thứ tự các giá trị của một mảng, sử dụng phương thức `list.reverse()`. Phương thức này không trả
về kết quả mà thay đổi trực tiếp mảng `list`.

Ví dụ:

```
numbers = |1, 2, 3, 4]
numbers.reverse()

Print numbers

(Hiển thị [4, 3, 2, 1])
```

9. Sắp xếp giá trị các phần tử

Để sắp xếp thứ tự của giá trị trong mảng, sử dụng phương thức `list.sort([func])` để sắp xếp. Nếu tham số đầu
vào là hàm func không truyên vào thì mặc định là sắp xếp theo giá trị tăng dần. Phương thức này không trả về kết
quả mà thay đổi trực tiếp mảng `list`

Ví dụ:

```
aList = [123, 'xyz', 'zara', 'abc', 'xyz']
aList.sort()

Print "List : ", aList

(Hiển thị List : 123 ; 'abc', 'xyz', 'xyz', 'zara'])
```

Cách triển khai hàm compare func() cũng giống như hàm usort trong PHP. Hàm trả về các giá trị o, -1 và 1.

Revision #2
Created 5 June 2021 14:21:06 by Laptrinh.vn
Updated 5 June 2021 15:01:36 by Laptrinh.vn