```
# coding=utf-8
__author__ = "dvminhtan@gmail.com"
__copyright__ = "Nguyễn Minh Tân"

""" List
    - Introduction
    - Accessing list
    - Operations
    - Working with lists
    - Functions and Methods
"""

""" Introduction
    - Python cung cấp cho chúng ta một loạt các kiểu dữ liệu hỗn hợp, dạng là một chuỗi liên tiếp gì đó.
    - Trong đó, kiểu dữ liệu List - rất linh hoạt và được dùng thường xuyên nhất.
"""

# Cách tạo ra một list: Đặt các phần tử vào trong cặp ngoặc vuông và cách nhau bởi dấu phẩy (,)
empty_list = []  # Một list rỗng = list không có phần tử nào
my_list = [26, 10, 1992, 28, -2000]  # Một list chứa các số nguyên
your_list = [1, 'hi', 'greeting', 3.14]  # Một list hỗn hợp các kiểu dữ liệu
our_list = ['123-string', [0, 1], ['str1', 'str-2'], [empty_list, my_list, your_list]]  # Nested list

""" Truy cập vào list (=> Giống với chuỗi. Ta có thể coi chuỗi là 1 list, mỗi phần tử là một ký tự)
    - Bằng chỉ số dương: Truy cập bằng chỉ số với toán tử []. Chỉ số được đánh từ 0, đến (Số lượng phần tử - 1).
```

Cố tính truy cập vào một gia trị chỉ số khác sẽ báo lỗi IndexError

- Bằng chỉ số âm: Chỉ số âm được đánh như sau: -1 là phần tử cuối cùng, ..., đến -(số lượng phần tử) là phần tử đầu tiên

- Bằng đoạn cắt: Sử dụng toán tử cắt dấu hai chấm

"""

```
my_list = ["first", 1, 2, 3, 'l', [4, 5, 6, 7]]
print(len(my_list))  # Hàm len() trả lại số lượng phần tử trong list
print(my_list[0])
print(my_list[3])
print(my_list[5])

# print(my_list[10])  # Lỗi IndexError: list index out of range

# Trong trường hợp trên là có list lồng nhau
print(my_list[5][2])  # Truy cập đến list con bằng [5], sau đó truy cập vào phần tử của list con bằng [2]
print(my_list[0][1])  # Truy cập vào phần tử đầu tiên của my_list => được 1 chuỗi, rồi truy cập vào phần tử thứ 2 của chuỗi

print(my_list[-2])  # Truy cập phần tử thứ 2 từ cuối lên
print(my_list[-1][-1])  # Truy cập vào phần tử cuối cùng trong phần tử cuối cùng của my_list
print(my_list[0][-3])  # Truy cập vào phần tử thứ 3 từ cuối lên trong phần tử đầu tiên của my_list

print(my_list[2:5])
print(my_list[:-3])
print(my_list[3:])
print(my_list[::3])
```

```
print(my_list[::-1])



""" Các phép toán với kiểu dữ liệu list. Khác với chuỗi thì list là kiểu có thể thay đổi được.
    Sau đây là minh họa cho các phép toán trên list
"""
# Phép toán gán (=): Dùng phép gán (=) để thay đổi 1 hoặc 1 đoạn phần tử tại chỉ số nào đó của list
my_list = [0, 1, 2, 4, 4, 5, 10]
my_list[3] = 1992
my_list[-1] = 2020
print(my_list)
my_list[1:4] = [-1, -2, -3]
print(my_list)

# Thêm một phần tử vào list bằng phương thức append() hoặc thêm một list vào 1 list bằng phương thức extend().
# => Luôn luôn là thêm vào cuỗi list gốc trở đi
your_list = [0, 1, 2]
your_list.append(3)
print(your_list)
your_list.extend([-3, -2, -1])
print(your_list)

# Phép toán cộng (+) 2 list để ra được list mới - chính là nối 2 list.
# Phép toán nhân list với số nguyên dương n, để ra được 1 list mới lặp n lần
our_list = [0, 1]
```

```
print(our_list + [-3, -2, -1])

print(our_list * 3)


# Chèn thêm 1 phần tử mới vào list bằng phương thức insert(index, value)

# Chèn thêm nhiều phần tử mới vào list bằng cách ép các phần tử đó vào 1 đoạn cắt rỗng [vị_trí_chèn:vị_trí_chèn]

my_list = [0, 2, 6, 7]

my_list.insert(2, 4)

print(my_list)

my_list[3:3] = [5, -5, -6, -7]  # thử cả với my_list[-3:-3] = [5, -5, -6, -7]

print(my_list)


# Xóa một hoặc nhiều phần tử trong list xử dụng toán tử del

my_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

del my_list[3]

print(my_list)

del my_list[-2]

print(my_list)

del my_list[2:4]

print(my_list)

del my_list[-3:-1]

print(my_list)

del my_list  # Xóa cả list đi

print(my_list)  # Lỗi: NameError: name 'my_list' is not defined


# Ngoài ra còn có thể dùng remove(phần_tử_muốn_xóa) hoặc pop(vị_trí_phần_tử_muốn_xóa)
```

```
# hoặc clear() để xóa hết các phần tử, tạo ra List rỗng
my_list = [0, 1, 2, 3, 8, 5, 6, 7, 8, 9]
my_list.remove(8)  # Xóa đi phần tử đầu tiên từ trái sang phải = phần tử muốn xóa
print(my_list)
my_list.remove(1992)  # Lỗi: ValueError: list.remove(x): x not in list
print(my_list.pop(3))  # Xóa đi và trả lại phần tử tại chỉ số 3
print(my_list)
print(my_list.pop())  # Xóa đi và trả lại phần tử tại vị trí cuối cùng
print(my_list)

my_list.clear()
print(my_list)

# Thêm nữa, có thể xóa các phần tử bằng cách dùng gán mảng rỗng vào 1 đoạn cắt
my_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
my_list[2:4] = []
print(my_list)

""" Danh sách các method của kiểu dữ liệu list
    - list.append(element): Thêm một phần tử element vào cuối list
    - list.extend(list2): Thêm nhiều phần tử trong list2 vào cuối của list
    - list.insert(index, element): Thêm phần tử element vào chỉ số index trong list
    - list.remove(element): Xóa phần tử element đầu tiên từ trái sang phải trong list nếu có, ko có sẽ lỗi
    - list.count(element): Đếm số lượng element trong list
    - list.pop(index=-1): Xóa và trả lại giá trị phần tử tại chỉ số index
    - list.reverse(): Đảo ngược list
```

```
    - list.sort(key=..., reverse=...): Sắp xếp list theo key nào đó, theo chiều tăng/giảm (reverse=True => giảm)
    <=> sorted(list, key=..., reverse=...)
    - list.copy(): Trả lại bản sao của list
    - list.clear(): Xóa bỏ toàn bộ các phần tử => mảng rỗng
"""
my_list = [0, 1, 2, 0, 3, 2, 5, 2]
print(f"Số lượng phần tử giá trị 2: {my_list.count(2)}")
my_list.reverse()
print(my_list)

print(sorted(my_list))  # sorted() ko làm thay đổi list gốc
print(my_list)
my_list.sort()
print(my_list)  # sort() ko trả kết quả mà thay đổi ngay trong list gốc

your_list = ["abc", "1234", []]
your_list.sort(key=len)
print(your_list)

ran_list = ["abc", "1234", []]
ran_list.sort(key=len, reverse=True)
print(ran_list)


""" Kiểm tra tồn tại:
    - in: kiểm tra xem có trong list không
```

```
    - not in: kiểm tra xem không có trong list
"""
languages = ["C/C++", "Python", "Java", "JavaScript"]
print("python" in languages)
print("Python" in languages)
print("C#" not in languages)

""" Duyệt list:
    - Qua chỉ số của list
    - Qua phép toán in
"""
languages = ["C/C++", "Python", "Java", "JavaScript"]
for i in range(len(languages)):
    print(f"I can use {languages[i]}")
print("===> Duyệt theo toán tử in:")
for lg in languages:
    print(f"I can use {lg}")
```