Bài 7: DỮ LIỆU KIỂU DANH SÁCH (LIST)

1. Khái niệm danh sách (list)

Danh sách (list) trong Python là một cấu trúc dữ liệu dùng để lưu trữ nhiều giá trị trong một biến duy nhất, mỗi giá trị đó được gọi là một phần tử trong danh sách.

- Các phần tử trong danh sách có thể khác kiểu dữ liệu (số, chuỗi, boolean...).
- Danh sách là có thứ tự (ordered) và thay đổi được (mutable).

➢ **Cú pháp:**

ten_danh_sach = [pt1, pt2, pt3, ...]

➢ **Ví dụ:**

so = [1, 2, 3, 4, 5]
chuoi = ["apple", "banana", "cherry"]
hon_hop = [10, "python", True, 3.14]

➢ **Lưu ý:**

- Các phần tử trong danh sách được đánh chỉ số bắt đầu từ chỉ số 0, 1, … (từ trái sang phải) và -1, -2, … (từ phải sang trái).
- Cách truy cập phần tử của danh sách a với chỉ số k là a[k]

2. Các hàm và phép toán cơ bản trong danh sách

2.1 Hàm len()

➢ Cú pháp: len(danh_sach) → Trả về số lượng phần tử trong danh sách.

➢ Ví dụ:

a = [1, 2, 3, 4]
print(len(a)) =>> 4

2.2 Phép nối +

Dùng để nối 2 danh sách thành 1 danh sách mới.

Ví dụ:

a = [1, 2, 3]
b = [4, 5]
c = a + b
print(c) =>> [1, 2, 3, 4, 5]

2.3 Phép lặp *

Lặp lại danh sách nhiều lần.

Ví dụ:

*a = [0] * 5*
print(a) =>> [0, 0, 0, 0, 0]

2.4 Toán tử in và not in

Dùng để kiểm tra phần tử có trong danh sách hay không.

<phần tử> in <danh sách>

Kết quả trả về: True (nếu phần tử thuộc danh sách) hoặc False (nếu phần tử không thuộc danh sách).

<phần tử> not in <danh sách>

Kết quả trả về: True (nếu phần tử không thuộc danh sách) hoặc False (nếu phần tử thuộc danh sách).
Ví dụ:
a = [1, 2, 3, 4]
print(2 in a) =>> True
print(5 not in a) =>> True

2.5 Hàm min(), max(), sum()
min(ds) → phần tử nhỏ nhất
max(ds) → phần tử lớn nhất
sum(ds) → tổng các phần tử số trong danh sách.
Ví dụ:
a = [4, 7, 2, 9]
print(min(a)) =>> 2
print(max(a)) =>> 9
print(sum(a)) =>> 22

2.6 Lệnh del
Cú pháp: del danh_sach[chỉ_số] → Xóa phần tử theo chỉ số.
Ví dụ:
a = [1, 2, 3, 4]
del a[1]
print(a) =>> [1, 3, 4]

3. Các phương thức của danh sách

Hàm/Phương thức	Chức năng
len(ds)	Độ dài danh sách
sum(ds)	Tổng các phần tử (số)
max(ds)	Phần tử lớn nhất
min(ds)	Phần tử nhỏ nhất
sorted(ds) hoặc ds.sort()	Trả về danh sách mới đã sắp xếp tăng dần
sorted(ds,reverse=True) hoặc ds.sort(reverse=True)	Trả về danh sách mới đã sắp xếp giảm dần
ds.reverse()	Trả về danh sách có các phần tử sắp xếp ngược lại so với ds ban đầu.
ds.append(x)	Thêm x vào cuối danh sách như 1 phần tử
ds.extend(x)	Thêm x vào cuối danh sách như một mở rộng tự nhiên
ds.insert(i, x)	Chèn x vào vị trí i
ds.remove(x)	Xóa phần tử đầu tiên có giá trị x
ds.pop()/ ds.pop(i)	Xóa và lấy ra phần tử cuối cùng của ds (thao tác pop của ngăn xếp)/Xóa phần tử tại vị trí i.
ds.clear()	Xóa dữ liệu đưa ds thành rỗng
ds.copy()	Trả về một danh sách mới là bản sao của ds

ds.count(x)	Đếm số lần xuất hiện của giá trị x trong danh sách
ds.index(x)	Trả về vị trí đầu tiên của x trong danh sách

4. Các thao tác trên danh sách

4.1 Khởi tạo danh sách nhiều phần tử

Có nhiều cách khởi tạo danh sách:

1. ds = [1, 2, 3]
2. ds = list((1, 2, 3))
3. ds = [] hoặc ds = list() # tạo danh sách rỗng
for i in range(100):
ds.append(i)
4. ds = list(range(5)) =>> [0, 1, 2, 3, 4]

Ngoài các cách trên, Python còn cung cấp nhiều cách linh hoạt để tạo danh sách:

- **Cú pháp 1**: <danh sách> = [<biểu thức> for <phần tử> in <tập giá trị>]
  Cú pháp trên dùng để tạo một danh sách gồm các phần tử là giá trị của <biểu thức> khi <phần tử> lần lượt nhận giá trị của <tập giá trị>
  Ví dụ:
  ds = [i for i in range(6)] =>> [0, 1, 2, 3, 4, 5]
  ds = [i *i for i in range(6)] =>> [0, 1, 4, 9, 16, 25]
  ds = [2*i for i in (1, 2, 3)] =>> [2, 4, 6]
- **Cú pháp 2**: <danh sách> = [<biểu thức> for <phần tử> in <tập giá trị> if <điều kiện>]
  Cú pháp trên dùng để tạo một danh sách gồm các phần tử là giá trị của <biểu thức> khi <phần tử> lần lượt nhận giá trị của <tập giá trị> thoả <điều kiện>
  Ví dụ:
  ds = [i for i in range(6) if i % 2 = = 0] =>> [0, 2, 4]
- Sử dụng nhiều vòng for để tạo danh sách: Mỗi phần tử của danh sách có thể là các danh sách, khi đó để tạo danh sách ta có thể sử dụng các vòng for lồng nhau.
  Ví dụ:
  ds = [[i, j] for i in (1, 2, 3) for j in range(2)]
  =>>> [[1, 0], [1, 1], [2, 0], [2, 1], [3, 0], [3, 1]]
  ds = [[i, j] for i in (1, 2, 3) for j in range(2) if i + j == 2]
  =>>> [[1, 1], [2, 0]]

4.2 Lấy các phần tử trong danh sách

- Truy cập phần tử theo chỉ số: a[i] hoặc a[-1] cho phần tử cuối.
  Ví dụ:

a = [10, 20, 30, 40]
print(a[0]) # 10
print(a[-1]) # 40

- Lấy nhiều phần tử liên tiếp trong danh sách (slicing).
  <biến danh sách>[a:b]: trả về một danh sách gồm các phần tử từ chỉ số a đến chỉ số b-1 trong <biến danh sách>.

<biến danh sách>[:b]: trả về một danh sách gồm các phần tử từ chỉ số 0 đến chỉ số b-1 trong <biến danh sách>.
<biến danh sách>[a:]: trả về một danh sách gồm các phần tử từ chỉ số a đến hết danh sách trong <biến danh sách>.
<biến danh sách>[a:b:c]: trả về một danh sách gồm các phần tử từ chỉ số [a], [a + c], [a + 2c], …, [a + tc] với a + tc < b trong <biến danh sách>.

*** Ví dụ:**

```
a = [10, 20, 30, 40, 50, 60]
print(a[1:4])   =>>  [20, 30, 40]
print(a[:3])    =>> [10, 20, 30]
print(a[2:])    =>> [30, 40, 50]
print(a[1:5:2])   =>> [20, 40, 60]
```

5. Lưu ý khi sử dụng danh sách

- Chỉ số bắt đầu từ 0. Nếu tính theo chiều ngược lại thì chỉ số đầu tiên là -1
- Khi dùng del hoặc a[i], nếu chỉ số vượt phạm vi sẽ báo lỗi.
- sum(), min(), max() chỉ hoạt động với danh sách số.
- Dùng in với danh sách lớn có thể chậm vì Python phải quét toàn bộ.

6. Cú pháp đảo ngược danh sách

- ***Cách 1: Dùng cú pháp slicing***

```
ds = [1, 2, 3, 4, 5]
ds_dao = ds[::-1]
print(ds_dao)
```

* Giải thích:
ds[::-1] nghĩa là lấy toàn bộ danh sách nhưng bước nhảy -1 → duyệt ngược.
Kết quả: [5, 4, 3, 2, 1].

- ***Cách 2: Dùng phương thức reverse()***

```
ds = [1, 2, 3, 4, 5]
ds.reverse()
print(ds)
```

* Giải thích:
reverse() đảo ngược tại chỗ (không tạo danh sách mới, mà thay đổi danh sách gốc).
Kết quả: [5, 4, 3, 2, 1].

*** *Cách 3: Dùng hàm reversed()***

```
ds = [1, 2, 3, 4, 5]
ds_dao = list(reversed(ds))
print(ds_dao)
```

* Giải thích:
reversed(ds) trả về một đối tượng iterator sinh ra phần tử theo thứ tự ngược.
Dùng list() để biến thành danh sách mới.
Kết quả: [5, 4, 3, 2, 1].

*** Lưu ý:**

- Nếu muốn tạo bản sao ngược mà không ảnh hưởng đến danh sách gốc → dùng slicing hoặc reversed()
- Nếu chỉ cần đảo ngược danh sách ngay trong biến hiện tại → dùng reverse().
- Nếu chỉ cần duyệt ngược (không cần list mới), nên dùng trực tiếp:

```
for x in reversed(ds):
    print(x)
```

7. Nhập/xuất danh sách theo dữ liệu chuẩn. Đọc/ghi danh sách với dữ liệu kiểu file

7.1. Nhập/Đọc danh sách

- **TH1: Các phần tử cùng nằm trên 1 dòng (muốn lưu các phần tử thành một danh sách)**

Cú pháp

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
Cách 1: Dùng input() *ds = input().split()* # các phần tử là chuỗi cách nhau bởi khoảng trắng (nếu cách nhau bởi dấu phẩy ta thay .split(,) *ds = list(map(int,input().split())* # các phần tử là số nguyên cách nhau bởi khoảng trắng. ***Cách 2: Dùng sys.stdin*** *import sys* *ds = sys.stdin.readline().split())* # các phần tử là chuỗi cách nhau bởi khoảng trắng (nếu cách nhau bởi dấu phẩy ta thay .split(,) *ds=list(map(int,sys.stdin.readline().split())* # các phần tử là số nguyên cách nhau bởi khoảng trắng.	**Dùng open() đọc trực tiếp file** with open(<tên tệp>, 'r') as <biến tệp>: #Nếu các phần tử là chuỗi *ds = list(<biến tệp>.readline().strip().split())* # Nếu các phần tử là số nguyên *ds = list(map(int, <biến tệp>.readline().strip().split()))*

- **TH2: Mỗi phần tử nằm trên một dòng với số phần tử biết trước (muốn lưu các phần tử thành một danh sách)**

Cú pháp:

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
+ *Dùng for* **Cú pháp 1:** *n = int(input())* *ds = [ ]* *for _ in range(n):* *x = input()* *ds.append(x)* **Cú pháp 2:**	**Dùng open() đọc trực tiếp file** *with open(<tên tệp>, 'r') as <biến tệp>:* *n = int(<biến tệp>.readline().strip())* *for _ in range(n):* *ds = list(map(int, <biến tệp>.readline().strip().split()))*

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
n = int(input()) *ds = [input() for _ in range(n)]* **Cú pháp 3:** *n = int(input())* *for _ in range(n):* *a = list(map(int,input().split()))* **+ *Dùng while*** (ít dùng hơn) *n = int(input())* *ds = [ ]* *i = 0* *while i < n:* *x = input()* *ds.append(x)* *i += 1*	

- **TH3: Mỗi phần tử nằm trên một dòng với số phần tử chưa biết trước (muốn lưu các phần tử thành một danh sách)**

Cú pháp:

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
+ *Cách 1: Nhập đến khi gặp dòng rỗng ("")* *ds = []* *while True:* *dong = input("Nhập phần tử (Enter để dừng): ").strip()* *if dong == "": # nếu người dùng chỉ nhấn Enter* *break* *ds.append(dong)* **+ *Cách 2: Nhập đến khi gặp ký tự đặc biệt (ví dụ "STOP")*** *ds = []* *while True:* *dong = input("Nhập phần tử (gõ STOP để dừng): ").strip()* *if dong == "STOP":* *break* *ds.append(dong)* **+ *Cách 3: Dùng try/except đến khi hết dữ liệu (EOF)*** *ds = []*	**Dùng open() đọc trực tiếp file** *ds = []* *with open(<tên tệp>, "r") as <biến tệp>:* *for dong in <biến tệp>:* *dong = dong.strip()* *if dong:* *ds.append(int(dong))*

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
try: * while True:* * dong = input().strip()* * ds.append(dong)* *except EOFError:* * pass*	

- **TH4: Nhập/đọc t dòng , mỗi dòng gồm n phần tử cách nhau bởi dấu cách.**

Cú pháp:

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
Cách 1: Dùng input() *t = int(input("Nhập số dòng t: "))* *A = [] # danh sách lưu ma trận* *for _ in range(t):* * dong = input().strip().split() # tách theo dấu cách* * dong = [int(x) for x in dong] # chuyển sang số nguyên* * A.append(dong) # thêm dòng vào ma trận* **Cách 2: Dùng sys.stdin() (nhập nhanh với dữ liệu lớn)** *import sys* *t = int(sys.stdin.readline().strip())* *A = []* *for _ in range(t):* * dong = sys.stdin.readline().strip().split()* * dong = [int(x) for x in dong]* * A.append(dong)*	**Dùng open() đọc trực tiếp file** *A = []* *with open("<tên tệp>", "r") as f:* * t = int(f.readline().strip()) # dòng đầu là số dòng t* * for _ in range(t):* * dong = f.readline().strip().split()* * dong = [int(x) for x in dong]* * A.append(dong)*

- **TH5: Nhập/đọc nhiều dòng (không giới hạn), mỗi dòng gồm n số nguyên cách nhau bởi dấu cách.**

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
ls = [] *while True:* * try:* * dong = input()* * if not dong.strip():*	*with open("<tên tệp>", "r") as f:* *()]#cho mỗi dòng là 1 phần tử của danh sách a(đã loại bỏ kí tự \n)* *a = [dong.strip()for dong in f.readlines]*

Nhập dữ liệu chuẩn	Đọc dữ liệu từ file
break *ls.append(dong)* *except EOFError:* *break* # Cách xử lý dòng: *for dong in ls:* *l, r = map(int, dong.strip().split()) # phương thức strip() là loại bỏ khoảng trắng ở đầu và cuối dòng.* *…….*	**Cách khác:** (Đọc toàn bộ dữ liệu) *with open("<tên tệp>", "r") as f:* *a = f.read().splitlines()* (Lưu ý: *fi.readlines()* # giữ lại dấu xuống dòng '\n' *fi.read().splitlines()* # loại bỏ '\n') # Cách xử lý dòng: *for dong in a:* *if not dong.strip():* *continue* *l, r = map(int, dong.strip().split())* *…..*

7.2. Xuất danh sách

Xuất ra màn hình chuẩn	Ghi dữ liệu ra file
1) In ra danh sách sau khi nhập từ bàn phím: *print(a)* ***2) In ra các phần tử của danh sách a trên cùng 1 dòng cách nhau bởi dấu cách hoặc dấu phẩy hoặc dấu chấm:*** *- Cách 1: for i in a:* *print(i, end=" ") # hoặc print(i, end=",") hoặc print(i, end=";")* **Hoặc in theo vị trí của các phần tử** *for in range(len(a)):* *print(a[i], end = " ")* (cách này khi in ra thì cuối danh sách sẽ dư 1 khoảng trắng hoặc 1 dấu ",” hoặc một dấu ";") *- Cách 2:* *+ print(" ". join(map(str,a)))* *+ print(" ". join(str(i) for i in a))* *- Cách 3:* print(*a, sep = ",") nếu các phần tử cách nhau bằng khoảng trắng thì có thể dùng lệnh print(*a) nhưng hai lệnh này có dư kí hiệu xuống dòng do mặc định. (print(*a, sep=' ') thường an toàn, nhưng không tuyệt đối an toàn nếu hệ thống	***1) Ghi ra danh sách sau khi đọc từ file output:*** *with open("<tên tệp>", "w") as f:* *f.write(str(a))* ***2) Ghi ra các phần tử của danh sách a trên cùng 1 dòng cách nhau bởi dấu cách hoặc dấu phẩy hoặc dấu chấm:*** *with open("<tên tệp>", "w") as f:* *f.write(" ".join(str(x) for x in a))* # *hoặc f.write* (" ". join(map(str,a))) Cách khác: Ghi danh sách bằng print(..., file=f) (RẤT NHANH) *with open("<tên tệp>", "w") as f:* *for x in a:* *print(x, file=f)* # *hoặc print(*a, file=f)* ***3) Ghi mỗi phần tử của danh sách trên 1 dòng:*** **Cách 1:** *with open("<tên tệp>", "w") as f:* *for x in a:* *f.write(str(x) + "\n")* **Cách 2:** (nhanh hơn): s = "\n".join(str(x) for x in a)

Xuất ra màn hình chuẩn	Ghi dữ liệu ra file
chấm kiểm tra cả ký tự xuống dòng cuối cùng hoặc dấu cách sau cùng) * **Lưu ý:** Dùng " ". join(map(str,a))) là cách chuẩn, an toàn và được khuyến nghị nhất (kể cả ghi ra file out) ***3) In mỗi phần tử của danh sách trên 1 dòng*** *print('\n'.join(map(str, a))) # không có kí tự xuống dòng* *hoặc print ('\n'.join(map(str, a)) + '\n') # có thêm 1 dòng trống cuối danh .* ***4) In danh sách hai chiều (ma trận)*** ***Cách 1:*** *for row in A:* *print(*row)* ***Cách 2:*** *for row in A:* *print(" ".join(str(x) for x in row))* ***Cách 3:** In thủ công (ít dùng hơn)* *for row in A:* *for x in row:* *print(x, end=" ")* *print()*	with open("<tên tệp>", "w") as f: f.write(s) 4***) Ghi danh sách hai chiều (ma trận)*** *with open("<tên tệp>", "w") as f:* *for row in A:* *f.write(" ".join(str(x) for x in row) + "\n")*

BÀI TẬP ÁP DỤNG

Bài 1: Cho một danh sách A gồm các số nguyên:

a, Tìm độ dài hay số lượng phần tử của danh sách A

b, Tính tổng các số trong danh sách A

c, Tính tổng các số dương trong danh sách A.

d, Tìm giá trị lớn nhất, nhỏ nhất và vị trí của chúng trong danh sách A

e, Sắp xếp danh sách A tăng dần/ giảm dần.

f, Kiểm tra phần tử x có trong danh sách A không, nếu có hãy đếm số lần xuất hiện của nó và chỉ ra vị trí nhỏ nhất nó xuất hiện.

g, Đếm số lần xuất hiện của mỗi phần tử trong danh sách

h, In ra các phần tử xuất hiện đúng một lần

i, In ra tất cả các phần tử không trùng lặp trong danh sách.

Hướng dẫn code:

a, Tìm độ dài hay số lượng phần tử của danh sách A

Cách tối ưu - dùng hàm len(A)

```
A = list(map(int,input().split()))
n = len(A)
print(n)
```

b, Tính tổng các số trong danh sách A

- **Cách 1: Dùng vòng lặp for**

```
A = list(map(int,input().split()))
tong = 0
for x in A:
    tong += x
print(tong)
```

- **Cách 2: Dùng vòng lặp for với chỉ số**

```
A = list(map(int,input().split()))
tong = 0
for i in range(len(A)):
    tong += A[i]
print(tong)
```

- **Cách 3: Dùng vòng lặp while với chỉ số**

```
A = list(map(int,input().split()))
tong = 0
i = 0
while i < len(A):
    tong += A[i]
    i += 1
print(tong)
```

- **Cách 4: Dùng hàm sum() - cách tối ưu khuyên dùng**

```
A = list(map(int,input().split()))
tong = sum(A)
print(tong)
```

c, Tính tổng các số dương trong danh sách A.

- **Cách 1:**

```
A = list(map(int,input().split()))
tong = 0
for x in A:
        if x > 0:
        tong += x
print("Tổng số dương =", tong)
```

- **Cách 2**:

```
A = list(map(int,input().split()))
print(sum(x for x in A if x > 0))
```

d, Tìm giá trị lớn nhất, nhỏ nhất và vị trí của nó trong danh sách A

- **Cách 1:**

```
A = list(map(int,input().split()))
# Khởi tạo biến
max_val = A[0]
min_val = A[0]
pos_max = 0  # vị trí đầu tiên của giá trị lớn nhất
pos_min = 0  # vị trí đầu tiên của giá trị nhỏ nhất
# Duyệt danh sách
for i in range(len(A)):
    if A[i] > max_val:
        max_val = A[i]
        pos_max = i
    if A[i] < min_val:
        min_val = A[i]
        pos_min = i
# In kết quả
print(f"Giá trị lớn nhất: {max_val}, vị trí: {pos_max}")
print(f"Giá trị nhỏ nhất: {min_val}, vị trí: {pos_min}")
```

(***Lưu ý***: Vị trí ở đây là index bắt đầu từ 0.
Nếu muốn index bắt đầu từ 1, chỉ cần pos_max + 1, pos_min + 1)

- **Cách 2:**

```
A = list(map(int,input().split()))
LN = max(A)
NN = min(A)
VT_max = A.index(LN)
VT_min = A.index(NN)
 print(f"{LN} vị trí {VT_max })
print(f"{NN} vị trí {VT_min })
```

e, Sắp xếp danh sách A tăng dần/ giảm dần.

- **Cách 1: Dùng sort() (sắp xếp trực tiếp trên danh sách) – tối ưu nhất**

```
# Sắp xếp tăng
A = list(map(int,input().split()))
A.sort()
print(A)
# Sắp xếp giảm
A = list(map(int,input().split()))
A.sort(reverse=True)
print(A)
```

- **Cách 2: Dùng sorted() (tạo ra danh sách mới - bảo toàn danh sách ban đầu)**

 # Sắp xếp tăng:

```
A = list(map(int,input().split()))
B = sorted(A)
print(B)
```

 # Sắp xếp giảm:

```
A = list(map(int,input().split()))
B = sorted(A, reverse=True)
print(B)
```

f, Kiểm tra phần tử x có trong danh sách A không, nếu có hãy đếm số lần xuất hiện của nó và chỉ ra vị trí nhỏ nhất nó xuất hiện.

Cách 1:

```
A = list(map(int,input().split()))
x = int(input("Nhập phần tử cần tìm: "))
if x in A:
    dem = A.count(x)            # Đếm số lần xuất hiện
    first_pos = A.index(x) + 1      # Vị trí đầu tiên (tính từ 1)
    print(f"Phần tử {x} có trong danh sách.")
    print(f"Số lần xuất hiện: {dem}")
    print(f"Vị trí nhỏ nhất nó xuất hiện: {first_pos}")
else:
    print(f"Phần tử {x} KHÔNG có trong danh sách.")
```

Cách 2: Dùng Counter - tối ưu hơn

```
from collections import Counter
A = list(map(int,input().split()))
x = int(input("Nhập phần tử cần tìm: "))
dem = Counter(A)   # Đếm số lần xuất hiện của tất cả phần tử
if x in dem:       # Kiểm tra x có xuất hiện không
    count = dem[x]                          # Số lần xuất hiện
    first_pos = next(i for i, v in enumerate(A) if v == x) + 1  # Vị trí nhỏ nhất
# enumerate(A) tạo cặp (index, value)
# next(...) trả về phần tử đầu tiên thỏa điều kiện
    print(f"Phần tử {x} có trong danh sách.")
    print(f"Số lần xuất hiện: {count}")
    print(f"Vị trí nhỏ nhất nó xuất hiện: {first_pos}")
else:
    print(f"Phần tử {x} KHÔNG có trong danh sách.")
```

g, Đếm số lần xuất hiện của mỗi phần tử trong danh sách

Cách tối ưu

```
from collections import Counter
A = list(map(int,input().split()))
dem = Counter(A)
for phantu, solanxh in dem.items():
    print(f"{phantu}: {solanxh}")
```

(hoặc print(dem)

h, In ra các phần tử xuất hiện đúng một lần

- **Cách 1:**

```
A = list(map(int,input().split()))
for x in A:
    if A.count(x) == 1:
        print(x, end=" ")
```

- **Cách 2:**

```
A = list(map(int,input().split()))
print([x for x in A if A.count(x) == 1])
```

- **Cách 3**: Counter O(n), tránh A.count O(n^2) - Tối ưu hơn

```
from collections import Counter
A = list(map(int,input().split()))
dem = Counter(A)
for x in A:
    if dem[x] == 1:
        print(x, end=" ")
```

i, In ra tất cả các phần tử không trùng lặp trong danh sách.

Cách tối ưu

```
from collections import Counter
A = list(map(int,input().split()))
dem = Counter(A)
for x in dem.keys():
    print(x, end=" ")
```