**Giới thiệu về Set trong Python**

**Set** là một container, tuy nhiên không được sử dụng nhiều bằng [LIST](http://www.howkteam.vn/course/kieu-du-lieu-chuoi-trong-python--phan-1/kieu-du-lieu-list-trong-python--phan-1-1548)hay [TUPLE](http://www.howkteam.vn/course/kieu-du-lieu-list-trong-python--phan-1/kieu-du-lieu-tuple-trong-python-1550).

Một Set gồm các yếu tố sau:

* Được giới hạn bởi cặp ngoặc **{}**, tất cả những gì nằm trong đó là những phần tử của Set.
* Các phần tử của Set được phân cách nhau ra bởi dấu phẩy (**,**).
* Set không chứa nhiều hơn 1 phần tử trùng lặp

Set chỉ có thể chứa các **hashable object**nhưng chính nó **không phải** là một **hashable object**. Do đó, bạn không thể chứa một set trong một set.

**Ví dụ:**

>>> set\_1 = {69, 96}

>>> set\_1

{96, 69}

>>> type(set\_1) # kiểu set thuộc lớp set

<**class** '**set**'>

>>> **set\_2** = {'**How** **Kteam**'}

>>> **set\_2**

{'**How** **Kteam**'}

>>> **set\_3** = {(69, 'Free Education'), (1, 2, 3)}

>>> **set\_3**

{(69, 'Free Education'), (1, 2, 3)}

>>> **set\_4** = {[1, 2], [3, 4]}

**Traceback** (most recent call last):

File "<stdin>", line 1, **in** <module>

TypeError: unhashable type: 'list'

>>> set\_5 = {(1, 2, ['How Kteam'])}

Traceback (most recent call last):

File "<stdin>", line 1, **in** <module>

TypeError: unhashable type: 'list'

>>> set\_6 = {1, 2, {'HowKteam'}}

Traceback (most recent call last):

File "<stdin>", line 1, **in** <module>

TypeError: unhashable type: 'set'

**Cách khởi tạo Set**

**Sử dụng cặp  dấu ngoặc {} và đặt giá  trị bên trong**

**Cú pháp:**

**{<giá trị thứ nhất>, <giá trị thứ hai>, .., <giá trị thứ n – 1>, <giá trị thứ n>}**

|  |
| --- |
|  |

**Lưu ý:** Khi khởi tạo bằng cách này, ít nhất phải có một giá trị.

**Ví dụ:**

>>> set\_ = {1, 2, 3, 4}

>>> set\_

{1, 2, 3, 4}

>>> set\_1 = {1, 1, 1} # các giá trị trùng lặp bị loại bỏ

>>> set\_1

{1}

>>> empty\_set = {} # thử khởi tạo set rỗng

>>> empty\_set

{}

>>> type(empty\_set) # không phải là set

<**class** '**dict**'>

12

**Sử dụng Set Comprehension**

>>> set\_1 = {value **for** value **in** range(3)}

>>> set\_1

{0, 1, 2}

**Sử dụng constructor Set**

**Cú pháp:**

**set(iterable)**

**Công dụng:** Giống hoàn toàn với việc bạn sử dụng constructor **List**. Khác biệt duy nhất là constructor Set sẽ tạo ra một Set.

**Ví dụ:**

>>> set\_1 = set((1, 2, 3))

>>> set\_1

{1, 2, 3}

>>> set\_2 = set('How Kteam')

>>> set\_2 # set không quan tâm đến vị trí của các phần tử

{'o', ' ', 'a', 'm', 'H', 'K', 't', 'w', 'e'}

>>> set\_3 = set('aaaaaaaaa')

>>> set\_3

{'a'}

>>> set\_4 = set([1, 6, 8, 3, 1, 1, 3, 6])

{8, 1, 3, 6}

>>> empty\_set = set() # cách bạn tạo được empty set

>>> empty\_set

set()

|  |
| --- |
|  |

**Một số toán tử với Set trong Python**

Nhằm giúp các bạn dễ hiểu hơn về các toán tử với Set trong Python, Kteam minh họa các set dưới dạng biểu đồ Venn, với S1, S2 tương ứng các Set1, Set2 chứa các phần tử.

**Toán tử in**

**Cú pháp:**

value **in**<Set>

**Công dụng:** Kết quả trả về là True nếu value xuất hiện trong Set. Ngược lại sẽ là False

**Ví dụ:**

>>> 1 **in** {1, 2, 3}

**True**

>>> 4 **in** {'a', 'How Kteam', 5}

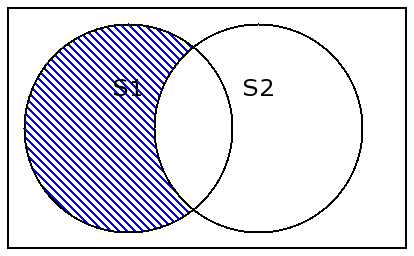
**False**

**Toán tử -**

**Cú pháp:**

<Set1> **-**<Set2>

**Công dụng:** Kết quả trả về là một Set gồm các phần tử chỉ tồn tại trong Set1 mà không tồn tại trong Set2



**Ví dụ:**

>>> {1, 2, 3} – {2, 3}

{1}

>>> {1, 2, 3} - {4}

{1, 2, 3}

>>> {1, 2, 3} - {1, 2, 3}

set()

>>> {1, 2, 3} - {1, 2, 3, 4}

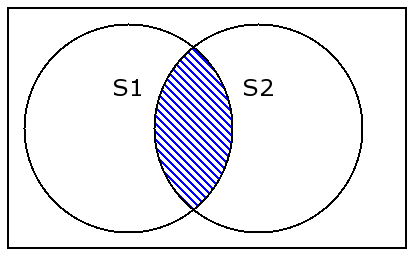
set()

**Toán tử &**

**Cú pháp:**

<Set1> **&**<Set2>

**Công dụng:** Kết quả trả về là một Set chứa các phần tử vừa tồn tại trong Set1 vừa tồn tại trong Set2



**Ví dụ:**

>>> {1, 2, 3} & {4, 5}

set()

>>> {1, 2, 3} & {1, 4, 5}

{1}

>>> {1, 2, 3} & {1, 2, 3}

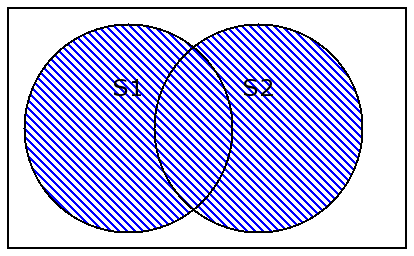
{1, 2, 3}

**Toán tử |**

**Cú pháp:**

<Set1> **|** <Set2>

**Công dụng:**Kết quả trả về là một Set chứa tất cả các phần tử tồn tại trong hai Set



**Ví dụ:**

>>> {1, 2, 3} | {1, 2, 3}

{1, 2, 3}

>>> {1, 2, 3} | {4, 5}

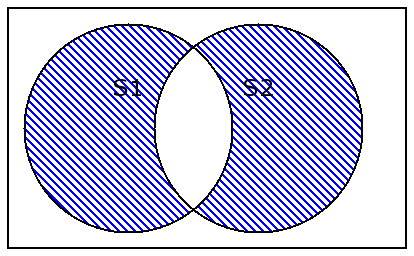
{1, 2, 3, 4, 5}

**Toán tử ^**

**Cú pháp:**

<Set1> **^** <Set2>

**Công dụng:**Kết quả trả về là một Set chứa tất cả các phần tử chỉ tồn tại ở một trong hai Set



**Ví dụ:**

>>> {1, 2, 3} ^ {4, 5}

{1, 2, 3, 4, 5}

>>> {1, 2, 3} ^ {1, 2, 3}

set()

>>> {1, 2, 3} ^ {1, 4}

{2, 3, 4}

**Indexing và cắt Set trong Python**

Ở trên Kteam đã đề cập về việc set không quan tâm đến vị trí của phần tử nằm trong set. Nên, việc indexing và cắt set trong Python không được hỗ trợ.

**Set không phải là một hash object**

Đúng như vậy! Điều đó có thể chứng minh theo hai cách:

Ở ví dụ dưới, bạn cũng thấy, ta đã thay đổi nội dung của set nhưng id của set vẫn là id ban đầu

**Ví dụ:**

>>> a = {1, 2}

>>> id(a)

52255360

>>> a.add(3)

>>> id(a)

52255360

Thêm nữa, set không thể chứa một set khác

>>> a = {1, 2}

>>> b = {a}

Traceback (most recent call last):

File "<stdin>", line 1, **in** <module>

TypeError: unhashable type: 'set'

**Củng cố bài học**

**Câu hỏi củng cố**

Giải thích lí do tại sao lại có sự thay đổi ở set a? Cho giải pháp khắc phục?

>>> a = {1, 2}

>>> b = a

>>> b.clear()

>>> a # tại sao lại trở thành set rỗng?

set()

Đáp án của phần này sẽ được trình bày ở bài tiếp theo. Tuy nhiên, Kteam khuyến khích bạn tự trả lời các câu hỏi để củng cố kiến thức cũng như thực hành một cách tốt nhất!