Métodos de los conjuntos

Métodos básicos

add()

Añade un ítem a un conjunto, si ya existe no lo añade:

```
c = set()
c.add(1)
c.add(2)
c.add(3)
c
```

```
{1, 2, 3}
```

discard()

Borra un ítem de un conjunto:

```
c.discard(1)
c
```

```
{2, 3}
```

copy()

Devuelva una copia de un conjunto, ya que éstos como la mayoría de colecciones se almacenan por referencia:

```
c1 = {1,2,3,4}
c2 = c.copy()
print(c1, c2)
c2.discard(4)
print(c1, c2)
```

```
{1, 2, 3, 4} {1, 2, 3, 4}
{1, 2, 3, 4} {1, 2, 3}
```

clear()

Borra todos los ítems de un conjunto:

```

```

```
c2.clear()
c2
```

```
set()
```

Comparación de conjuntos

```
c1 = {1,2,3}
c2 = {3,4,5}
c3 = {-1,99}
c4 = {1,2,3,4,5}
```

isdisjoint()

Comprueba si el conjunto es disjunto de otro conjunto, es decir, si no hay ningún elemento en común entre ellos:

```
c1.isdisjoint(c2)
```

```
False
```

issubset()

Comprueba si el conjunto es subconjunto de otro conjunto, es decir, si sus ítems se encuentran todos dentro de otro:

```
c3.issubset(c4)
```

```
False
```

issuperset()

Comprueba si el conjunto es contenedor de otro subconjunto, es decir, si contiene todos los ítems de otro:

```
c3.issuperset(c1)
```

```
False
```

Métodos avanzados

Se utilizan para realizar uniones, diferencias y otras operaciones avanzadas entre conjuntos.

Suelen tener dos formas, la normal que **devuelve** el resultado, y otra que hace lo mismo pero **actualiza** el propio resultado.

union()

Une un conjunto a otro y devuelve el resultado en un nuevo conjunto:

```
c1 = {1,2,3}
c2 = {3,4,5}
c3 = c1.union(c2)
print(c1, "+", c2, "=", c3)
```

```
{1, 2, 3} + {3, 4, 5} = {1, 2, 3, 4, 5}
```

update()

Une un conjunto a otro en el propio conjunto:

```
c1.update(c2)
c1
```

```
{1, 2, 3, 4, 5}
```

difference()

Encuentra los elementos no comunes entre dos conjuntos:

```
c1 = {1,2,3}
c2 = {3,4,5}

c1.difference(c2)
```

```
{1, 2}
```

difference_update()

Guarda en el conjunto los elementos no comunes entre dos conjuntos:

```
c1.difference_update(c2)
c1
```

```
{1, 2}
```

intersection()

Devuelve un conjunto con los elementos comunes en dos conjuntos:

```
c1 = {1,2,3}
c2 = {3,4,5}

c1.intersection(c2)
```

```
{3}
```

intersection_update()

Guarda en el conjunto los elementos comunes entre dos conjuntos:

```
c1.intersection_update(c2)
c1
```

```
{3}
```

symmetric_difference()

Devuelve los elementos simétricamente diferentes entre dos conjuntos, es decir, todos los elementos que no concuerdan entre los dos conjuntos:

```
c1 = {1,2,3}
c2 = {3,4,5}

c1.symmetric_difference(c2)
```

```
{1, 2, 4, 5}
```

Última edición: 1 de Octubre de 2018