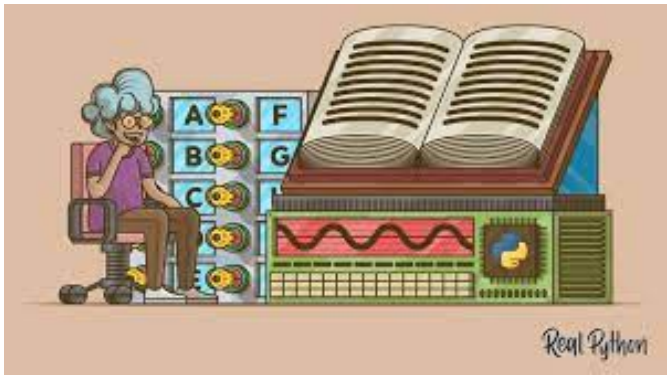
Lenguaje de Programación Python

Conjuntos

Dr. Mario Marcelo Berón
Argentina Programa
Universidad Nacional de San Luis

Tipos Conjunto

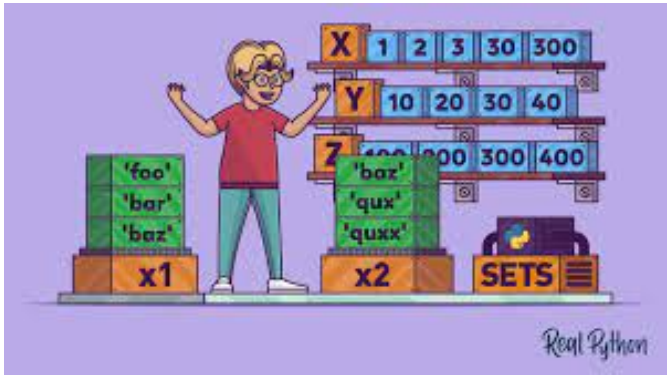
- Un tipo de dato conjunto es una colección que soporta los operadores de membresía *in*, *not in*, la función de tamaño *len*.



- Se proveen dos tipos de datos conjunto: conjunto mutable *set* y conjunto inmutable *frozenset*.
- Se pueden agregar a un conjunto objetos de tipo *float*, *frozenset*, *int*, *str* y *tuple*.

Tipos Conjunto

- Cuando se itera sobre conjuntos los elementos se proveen en orden aleatorio.



- Los tipos conjuntos se pueden comparar usando los operadores de comparación estándar (<,>,==,!=,<=,>=).
- == y != tiene el significado usual. Las comparaciones se aplican ítem por ítem.

Set

- Un conjunto es una colección desordenada de cero o más referencias a objetos.
- Son mutables es decir se pueden agregar y eliminar objetos.

- No tiene la noción de índice.
- No se pueden hacer rebanadas ni zancadas.

Set

set

set()

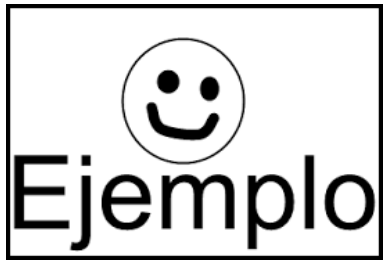
Tarea

Crea un conjunto.

Tarea

- Sin argumentos retorna un conjunto vacío.
- Con argumentos puede suceder:
  - Si es un conjunto crea una copia superficial del argumento.
  - Con cualquier otro argumento intenta convertir el objeto dado a un conjunto.
  - Acepta solo un argumento.
- Los conjuntos pueden ser creados sin usar la *set()* pero los conjuntos vacíos tienen que ser creados usando *set()* no usando llaves.

Operaciones

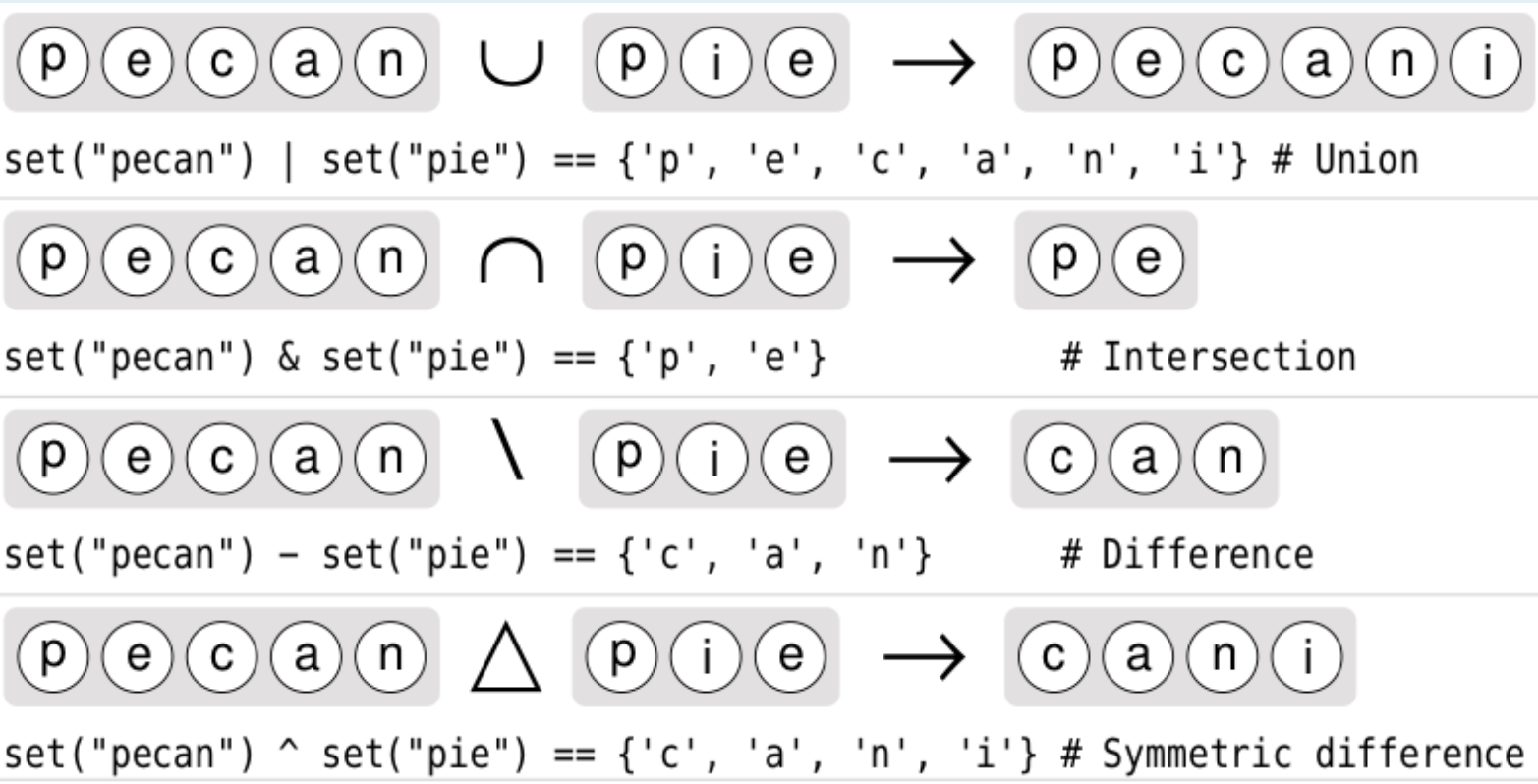


dict

```
set("Manzana")
set("Naranja")
{'P', 'E', 'R', 'A'}
```

Operaciones

Unión, Intersección,Diferencia, Diferencia Simétrica



Métodos

add

s.add(x)

Tarea

Incorpora el ítem x al conjunto s.

Ejemplo

```
>>> s={10,20,30}
>>> s.add(40)
>>> s
{40, 10, 20, 30}
```

Métodos

clear

s.clear()

Tarea

Elimina todos los ítems del conjunto s.

Ejemplo

```
>>> s={10,20,30}
>>> s.add(40)
>>> s
{40, 10, 20, 30}
```

Métodos

copy

s.copy()

Tarea

Retorna una copia superficial del conjunto s. Este método puede ser utilizado con *frozensets*.

Ejemplo

```
>>> import copy
>>> s #{10, 20, 30}
>>> t=s.copy() #t {10, 20, 30}
```

Métodos

difference - s-t

s.difference(t)

Tarea

Retorna un nuevo conjunto que tiene los ítems que están en s y que no están en t.

Ejemplo

```
>>> s #{10, 20, 30}
>>> t #{10, 20, 30}
>>> s.difference(t) #set()
```

Métodos

difference - s-=t

s.difference_update(t)

Tarea

Elimina de s todo ítem que está en t.

Ejemplo

```
>>> s
{10, 20,s 30}
>>> s.difference_update(t)
>>> s #set()
```

Métodos

difference - s-=t

s.discard(x)

Tarea

Si x está en s lo elimina.

Ejemplo

```
>>> s={10,20,30}
>>> s.discard(10)
>>> s
{20, 30}
```

Métodos

difference - s-=t

s.intersection(t)

Tarea

Retorna un nuevo conjunto con la intersección de s y t. Este método se puede utilizar con *frozenset*.

Ejemplo

```
>>> s={10,20,30,-1}
>>> t={10,20,30}
>>> s.intersection(t) #s{10, 20, 30}
```

Métodos

difference - s-=t

s.pop()

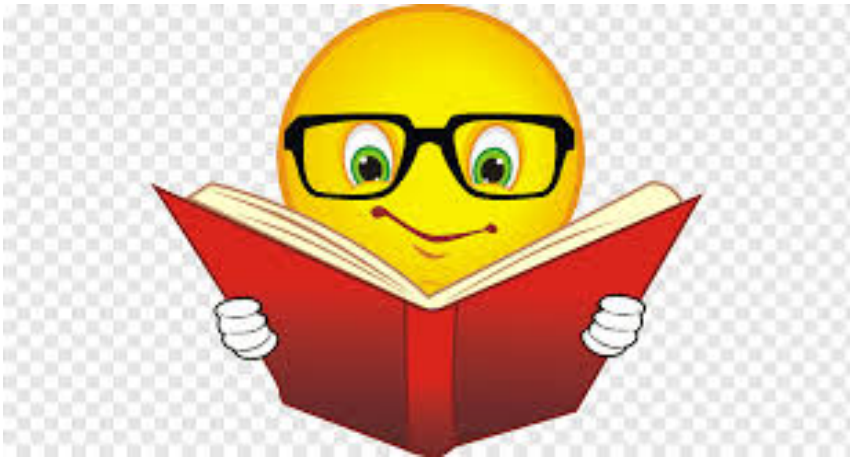
Tarea

Retorna y elimina un elemento aleatorio de s o dispara una excepción KeyError si s está vacío.

Ejemplo

```
>>> s={10,20,30,-1}
>>> s.pop()
10
```

Métodos



.... y... muchos métodos más

Frozenset

- Es un conjunto que una vez creado no se puede cambiar.

- Son inmutables, en consecuencia solo permite el uso de métodos y operadores que producen un resultado sin afectar el frozenset o conjunto a los cuales se aplica.

Frozenset

set

frozenset()

Tarea

Crea un *frozenset*.

Tarea

- Sin argumentos retorna un frozenset vacío.
- Con argumentos puede suceder:
  - Si es un frozenset retorna una copia superficial del argumento.
  - Con cualquier otro argumento intenta convertir el objeto dado a un frozenset.
  - Acepta solo un argumento.

Frozenset

Las operaciones soportadas por este tipo son:*frozenset.copy()*, *frozenset.difference()* (-), *frozenset.intersection()* (&), *frozenset.isdisjoint()*, *frozenset.issubset()* (<=; also < for proper subsets), *frozenset.issuperset()* (>=; also > for proper supersets), *frozenset.union()* (|), y *frozenset.symmetric_difference()* (ˆ)

Frozenset

- Si un operador binario se usa con un set y con un frozenset el tipo de dato del resultado es el mismo tipo de dato del operando que está a la izquierda.
- En el caso de == y != el orden no importa y el resultado producirá True o False.