

PROGRAMACION I



CLASE 11 – TUPLAS Y CONJUNTOS

PROF. ING. VERÓNICA GALATI

TUPLAS

Son similares a las listas
Sus elementos **son inmutables**

Intentar modificar una tupla provoca un error:
TypeError



TUPLAS

Para crear una tupla, se utiliza () y ,
() son opcionales

```
semana = ("Lunes", "Martes", "Miercoles", "Jueves", "Viernes")
```

```
semana = "Lunes", "Martes", "Miercoles", "Jueves", "Viernes"
```

TUPLAS

`dias=()` Para crear una tupla vacía

`dias=("Lunes",)` Para crear una tupla con un elemento se debe agregar **coma**

TUPLAS

Ejemplos crear tuplas con un elemento

```
>>> var1="Lunes"  
>>> type(var1)  
<class 'str'>
```

```
>>> var3=("Lunes")  
>>> type(var3)  
<class 'str'>
```

```
>>> var2="Lunes",  
>>> type(var2)  
<class 'tuple'>
```

```
>>> var4=("Lunes",)  
>>> type(var4)  
<class 'tuple'>
```

TUPLAS

Ejemplos crear tuplas con un elemento

```
>>> var1=10  
>>> type(var1)  
<class 'int'>
```

```
>>> var2=(10)  
>>> type(var2)  
<class 'int'>
```

```
>>> var3=10,  
>>> type(var3)  
<class 'tuple'>
```

```
>>> var4=(10,)  
>>> type(var4)  
<class 'tuple'>
```

TUPLAS

Podemos utilizar el operador *

```
>>> binario=(0,1)*3  
>>> print(binario)  
(0, 1, 0, 1, 0, 1)
```

TUPLAS

En una tupla se pueden combinar distintos tipos de datos

```
>>> primavera=(21,"Septiembre")  
>>>  
>>> invierno=(21,"Junio")
```

Cada elemento puede contener una tupla:

```
>>> estaciones=(primavera,invierno)  
>>> print(estaciones)  
  
((21, 'Septiembre'), (21, 'Junio'))
```


TUPLAS

Podemos:

utilizar el operador +

acceder a cada elemento con subíndice

```
>>> fecha=()
>>> fecha=fecha + (25,)
>>> fecha += ("Enero",)
>>> fecha += (2009,)
>>> print(fecha)
(25, 'Enero', 2009)
```

```
>>> fecha[0]
25
>>> fecha[1]
'Enero'
>>> fecha[2]
2009
```

TUPLAS DE TUPLAS

Cada elemento **puede** contener una tupla

```
>>> fecha  
(25, 'Enero', 2009)
```

```
>>> alumno="Lautaro",  
>>> alumno=alumno +(fecha,)   
>>> print(alumno)  
('Lautaro', (25, 'Enero', 2009))
```

```
>>> print(alumno[0])  
Lautaro  
>>> print(alumno[1])  
(25, 'Enero', 2009)  
>>> print(alumno[1][0])  
25  
>>> print(alumno[1][1])  
Enero  
>>> print(alumno[1][2])  
2009
```

TUPLAS

Se puede iterar con un ciclo:

```
semana = ("Lunes", "Martes", "Miercoles", "Jueves", "Viernes")  
  
for dia in semana:  
    print(dia)
```

```
>>> %Run tuplas_1.py
```

```
Lunes  
Martes  
Miercoles  
Jueves  
Viernes
```

TUPLAS

Se puede utilizar rebanadas:

```
semana = ("Lunes", "Martes", "Miercoles", "Jueves", "Viernes")  
  
print(semana[1:3])  
  
>>> %Run tuplas_1.py  
('Martes', 'Miercoles')  
  
findeLargo=semana[len(semana)-1:] + ("Sabado", "Domingo")  
  
print(findeLargo)  
('Viernes', 'Sabado', 'Domingo')
```

TUPLAS

Funciones, Operadores y Métodos

Funciones: `len()`, `max()`, `min()`, `sum()`

Operador: `*` `+` `in`

Métodos: `index()`, `count()`

TUPLAS

Empaquetado

Es el proceso por el cual, una serie de valores simples se convierten en una tupla

```
dia=22  
mes=10  
año=2019  
fecha=(dia,mes,año)  
print(fecha)
```

```
>>> %Run tuplas_1.py  
(22, 10, 2019)
```

TUPLAS

Desempaquetado

Es el proceso por el cual, una tupla se asigna a una serie de variables simples

```
fecha=(25, 'Enero', 2009)  
dia,mes,año=fecha
```

TUPLAS

Empaquetado - Desempaquetado

```
def DiaSiguiente(dia, mes, anio):  
    """ Determina el dia siguiente a una fecha ingresada  
    Devuelve el dia, mes y año correspondientes al dia siguiente de 1  
    Parámetros:  
    dia -- numero entero positivo correspondiente al día del mes  
    mes -- numero entero positivo correspondiente al mes del año  
    anio -- numero entero positivo correspondiente al año calendario  
    """  
    ...  
    return dia, mes, anio  
  
def __main__():  
    ...  
    dia, mes, anio = DiaSiguiente(dia, mes, anio)
```


CONSULTAS???



Práctica 8
Ejercicios 1 al 6

CONJUNTOS (SET)

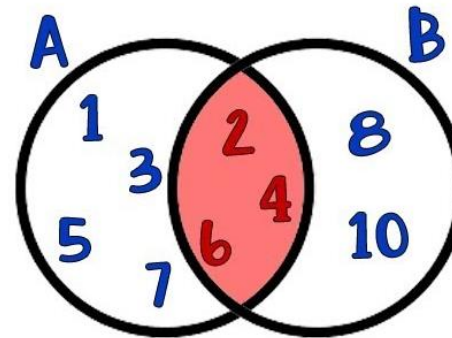
Es una colección de elementos sin orden y sin duplicados

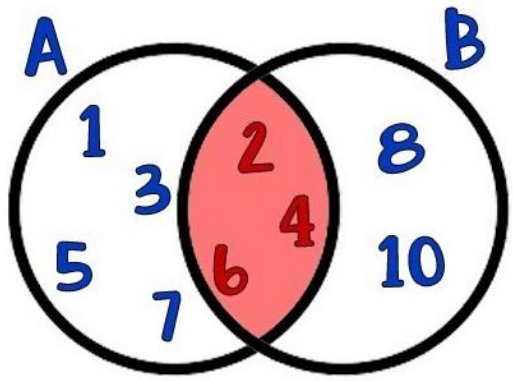
Los mismos conjuntos de matemática: Python permite hacer operaciones sobre conjuntos en forma simple

$A = \{1, 2, 3, 4, 5, 6, 7\}$

$B = \{2, 4, 6, 8, 10\}$

$A \cap B$

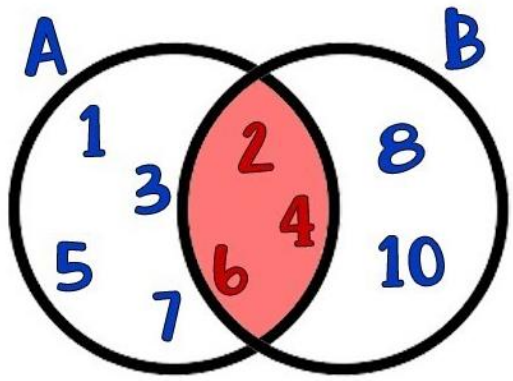




CONJUNTOS (SET)

Es una colección de elementos sin orden y sin duplicados
No tienen orden interno : **NO es una secuencia**

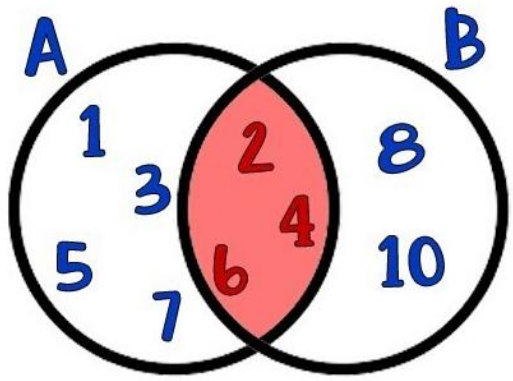
Pueden contener elementos de distinto tipo



CONJUNTOS

Para crear un conjunto, se utiliza $\{$ y $\}$,

```
países={"Argentina", "Brasil", "Chile", "Uruguay"}
```



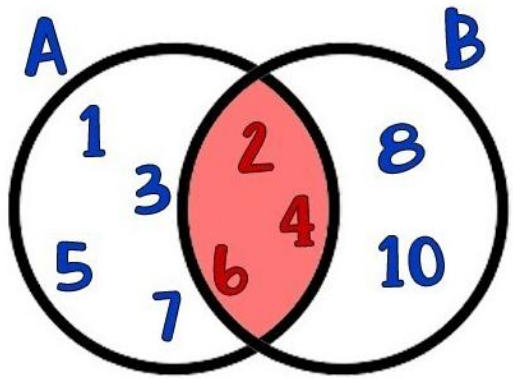
CONJUNTOS

Funciones, Operadores y Métodos

Funciones: `len()`, `max()`, `min()`, `sum()`

Operador: `in`, `not in`, matemáticos

Métodos: `add`, `remove`, `discard`, `clear`, `issubset`

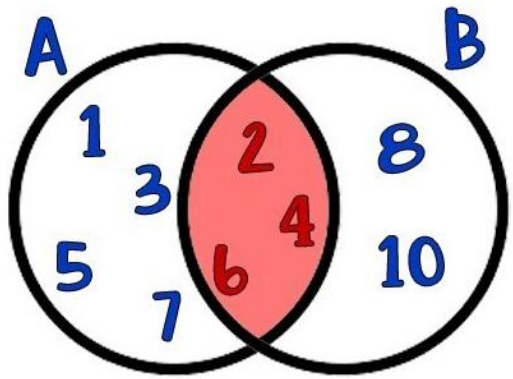


CONJUNTOS

Operadores matemáticos & intersección

```
A={1,2,3,4,5,6,7}  
B={2,4,6,8,10}  
C=A&B  
print(C)
```

```
>>> %Run conjuntos_1.py  
{2, 4, 6}
```

CONJUNTOS

Operadores matemáticos

| unión

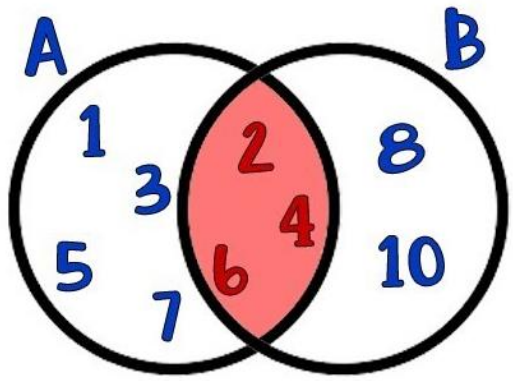
```
A={1,2,3,4,5,6,7}
```

```
B={2,4,6,8,10}
```

```
C=A|B
```

```
print(C)
```

```
>>> %Run conjuntos_1.py  
{1, 2, 3, 4, 5, 6, 7, 8, 10}
```



CONJUNTOS

Operadores matemáticos

- resta

```
A={1,2,3,4,5,6,7}
```

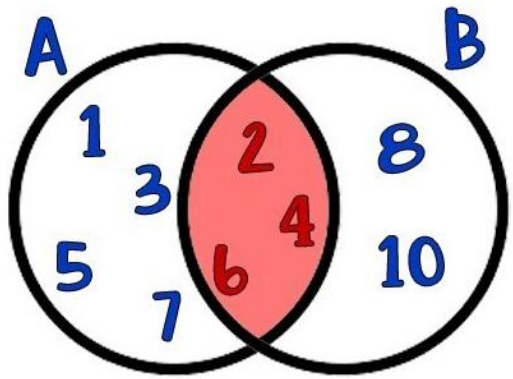
```
B={2,4,6,8,10}
```

```
C=A-B
```

```
print(C)
```

```
>>> %Run conjuntos_1.py
```

```
{1, 3, 5, 7}
```

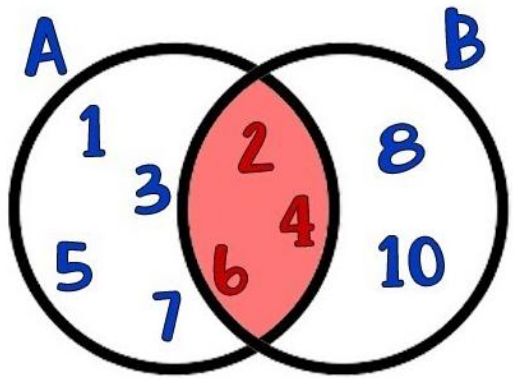
CONJUNTOS

Operadores matemáticos

\wedge diferencia simétrica

```
A={1,2,3,4,5,6,7}
B={2,4,6,8,10}
C=A^B
print(C)
```

```
>>> %Run conjuntos_1.py
{1, 3, 5, 7, 8, 10}
```



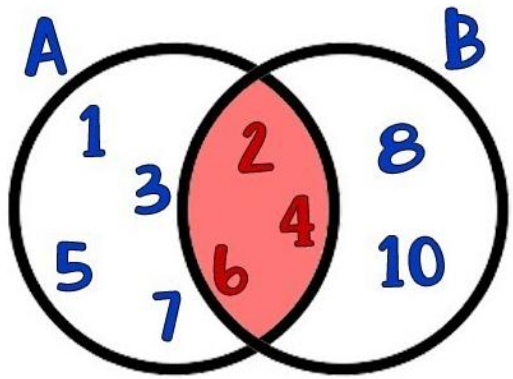
MÉTODOS CONJUNTOS

add(<elemento>)

Agrega un elemento al conjunto

```
países={"Argentina", "Brasil", "Chile", "Uruguay"}  
países.add("Paraguay")  
print(países)
```

```
>>> %Run tuplas_2.py  
{'Brasil', 'Uruguay', 'Chile', 'Paraguay', 'Argentina'}
```

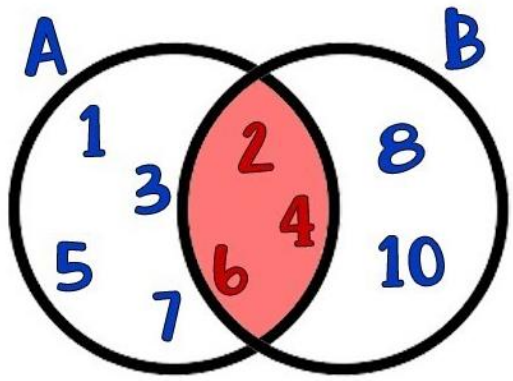


MÉTODOS CONJUNTOS

remove(<elemento>)

Elimina el elemento del conjunto, Provoca una excepción (**KeyError**) si no está presente.

```
países={"Argentina", "Brasil", "Chile", "Uruguay"}  
países.remove("Brasil")
```



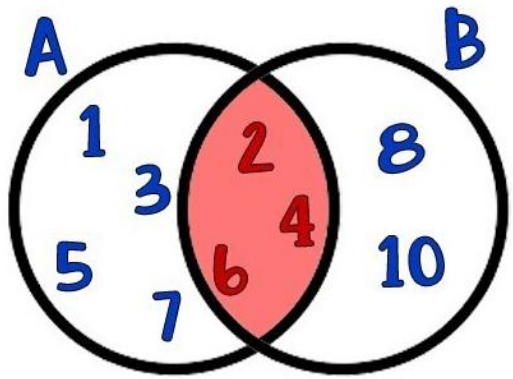
MÉTODOS CONJUNTOS

discard(<elemento>)

Elimina el elemento del conjunto.

No provoca error si no se encuentra en el conjunto.

```
países={"Argentina", "Brasil", "Chile", "Uruguay"}  
países.discard("Brasil")
```

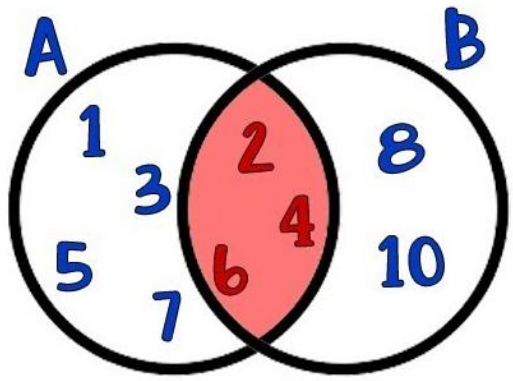


MÉTODOS CONJUNTOS

clear()

Elimina TODOS los elementos del conjunto.

```
países={"Argentina", "Brasil", "Chile", "Uruguay"}  
países.clear()
```



MÉTODOS CONJUNTOS

issubset(<conjunto>)

Retorna True si <conjunto> esta incluido en el conjunto

```
conjunto = {3, 4, 5}
```

```
if conjunto.issubset({2, 3, 4, 5, 6}):
```

```
    print("Incluido")
```


CONVERTIR TIPO DE DATOS

set(<iterable>) :convierte a conjunto

list(<iterable>) :convierte a lista

tuple(<iterable>):convierte a tupla

```
#Tupla a conjunto  
a=set((2,3,2))  
print(a)
```

```
#Conjunto a Tupla  
b=tuple({2,3,4})  
print(b)
```

```
#Tupla a Lista  
fecha=list((2,10,2019))  
print(fecha)
```

```
#Lista a Tupla  
fecha=tuple([2,10,2019])  
print(fecha)
```

```
#Lista a Conjunto  
conjunto=set([2,2,4,5,6,4])  
print(conjunto)
```

```
#Conjunto a Lista  
lista=list({2,5,7})  
print(lista)
```

CONSULTAS???



Práctica 8
Ejercicios 7 y 8