
# S.E.

# Programar
con Python

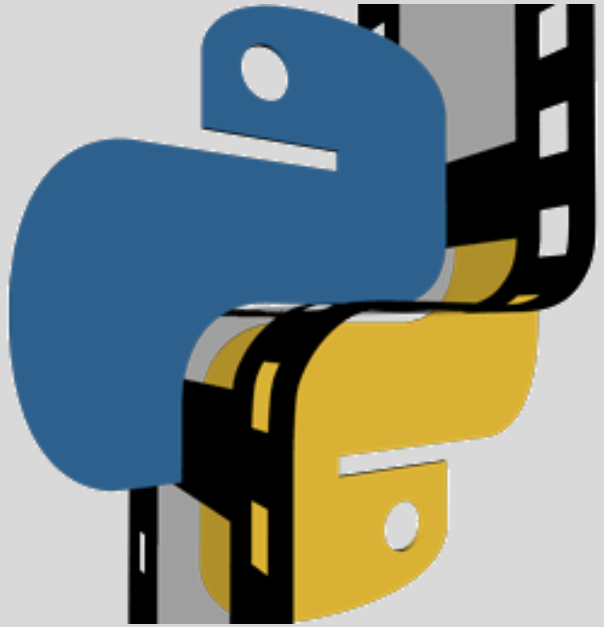
---

## Nivel inicial

## Clase 7

# Contenido

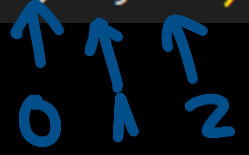
- ESTRUCTURAS DE DATOS
  - T-UPLAS
  - DICCIONARIOS
  - LISTAS (“arreglos”, “arrays”)
  - CONJUNTOS



# ESTRUCTURAS DE DATOS: T-UPLAS

**Las TUPLAS son tipos de datos en forma de vector.**

```
#datos_de_tupla = ( día, mes, año)
fecha = ( 4, 8, 25)
```




**Conocemos la cantidad de argumentos.**

**Son inmutables (no se pueden cambiar una vez definidas).**

**Se puede acceder a la información interna por separado.**

```
dia=fecha[0]
```

```
anio=fecha[2]
```

```
anio=fecha[-1]
```

# ESTRUCTURAS DE DATOS: T-UPLAS

**Las TUPLAS son tipos de datos en forma de vector.**

```
#datos_de_tupla = ( nombre, dirección, localidad, edad)
persona = ( 'Julieta', 'calle falsa 123', 'Springfield', 18)
```

```
nombre = persona[0]
direccion = persona[1]
edad = persona[-1]
```

**¿¿la edad al año siguiente??**

# ESTRUCTURAS DE DATOS: DICCIONARIO

**Los DICCIONARIOS son colecciones de pares de valores (clave, valor).**

**Las CLAVES son ÚNICAS. El VALOR es el que está asociado a una clave**

```
persona =dict(nombre = 'Laura' , edad = 25, dirección = 'calle Falsa 123')
```

**Son mutables (pueden cambiar una vez definidas).**

**Se puede acceder a la información interna por separado, utilizando get()**

```
print(persona.get('nombre'))
```

```
Laura
```

# ESTRUCTURAS DE DATOS: DICCIONARIO

**Los DICCIONARIOS son colecciones de pares de valores (clave, valor).**

**Las CLAVES son ÚNICAS. El VALOR es el que está asociado a una clave**

```
persona =dict(nombre = 'Laura' , edad = 25, dirección = 'calle Falsa 123')
```

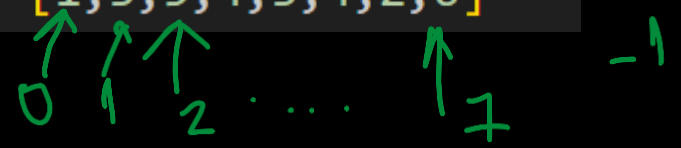
```
persona ={
    "nombre" : "Laura",
    "edad" : 25,
    "direccion" : "calle Falsa 123"
    }
```

# ESTRUCTURAS DE DATOS: LISTA

**Las LISTAS permiten almacenar un conjunto arbitrario de datos (no es necesario que sean del mismo tipo).**

**Características:**

```
lista = [1,3,5,4,3,4,2,6]
```




- **Tienen un orden**

```
primero = lista[0]
ultimo = lista[-1]
```

```
longitud = len(lista)
```

- **Se pueden anidar**

```
lista_de_listas = [[1,2],[5,3,8],[4,2,8,9]]
```

- **Son mutables y dinámicas**

```
lista = [1,3,5,4,3,4,2,6]
lista[5]=899
lista.append(55)
print(lista)
```

```
[1, 3, 5, 4, 3, 899, 2, 6, 55]
```

# ESTRUCTURAS DE DATOS: LISTA

**Listas que más usamos:**

## CADENA DE CARACTERES

```
nombre=['J','o','s','é',' ','d','e',' ','S','a','n',' ','M','a','r','t','í','n']
```

```
print(nombre[0])
print(nombre[8])
print("Longitud del nombre: ",len(nombre))
```

```
J
S
Longitud del nombre:  18
```

```
nombres = ["José de San Martín","Pedro","Marcela"]
```

```
print(nombres[0])
```

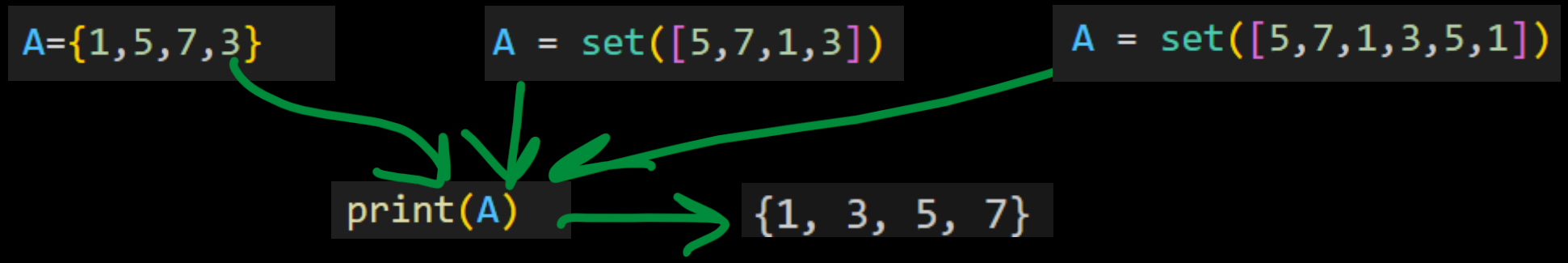
```
José de San Martín
```

```
print(nombres[0][0])
print(nombres[0][8])
print("Longitud del nombre: ",len(nombres[0]))
```

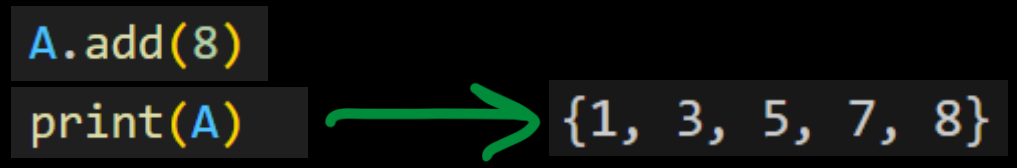
```
J
S
Longitud del nombre:  18
```

# ESTRUCTURAS DE DATOS: CONJUNTO

**Los CONJUNTOS son colecciones similares a las listas, pero sus elementos son únicos y no tienen un orden establecido (como pasa con las listas)**




**Agregar un elemento:**



# ESTRUCTURAS DE DATOS: CONJUNTO

**Los CONJUNTOS son colecciones similares a las listas, pero sus elementos son únicos y no tienen un orden establecido (como pasa con las listas)**

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

```
B={3,7,8}
```

**Unión e intersección:**

```
AunionB=A.union(B)
AintB=A.intersection(B)
print("A unión B:", AunionB)
print("A inters B:", AintB)
```

```
A unión B: {1, 3, 4, 5, 7, 8}
A inters B: {3, 7}
```

# EJERCICIOS

**7.1:** **Armar una lista de compras. Utilizar tuplas de tres elementos en donde el primer elemento es el producto a comprar, el segundo es la cantidad y el tercero es la unidad de medida.**

**7.2:** **Armar la estructura de una agenda en el que se guarden los siguientes datos de cada persona:**

- **Nombre**
- **Apellido**
- **Fecha Nacimiento**
- **Teléfono**
- **Dirección**
- **email**

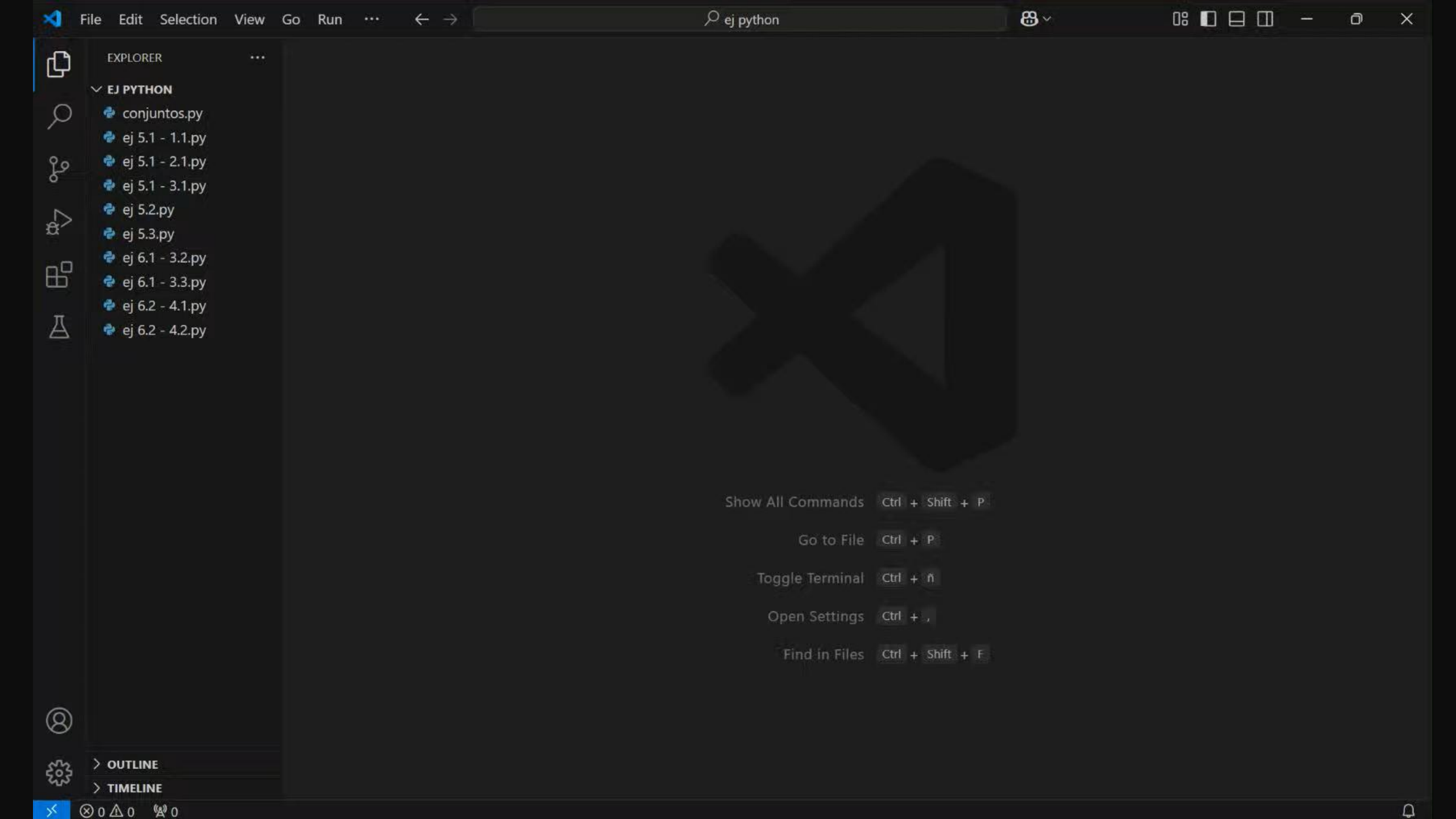
**7.3:** **Dados los siguientes conjuntos:**

**A = { 1,4,5,7,9,34,22}**

**B = { 1,5,4,34,22,67,5,2}**

**C = { 3,5,4,88,23,11}**

**Mostrar por pantalla: A∩B , A∩B∩C, (B∩C)∪A y A∪B∪C**


File
Edit
Selection
View
Go
Run
ej python
EXPLORER
EJ PYTHON
conjuntos.py
ej 5.1 - 1.1.py
ej 5.1 - 2.1.py
ej 5.1 - 3.1.py
ej 5.2.py
ej 5.3.py
ej 6.1 - 3.2.py
ej 6.1 - 3.3.py
ej 6.2 - 4.1.py
ej 6.2 - 4.2.py
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Toggle Terminal Ctrl + ñ
Open Settings Ctrl + ,
Find in Files Ctrl + Shift + F
OUTLINE
TIMELINE
0 0 0

Si te quedó alguna duda: ¡ mandame un mail a
cursos.se.programar@gmail.com

indicando la clase y la pregunta!