

3001A-V22 ESTRUCTURA DE DATOS Y ALGORITMOS



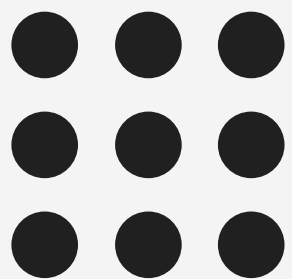
PROGRAMACIÓN DE ALTO NIVEL CON PYTHON

Profesor: Rafael Pérez Aguirre



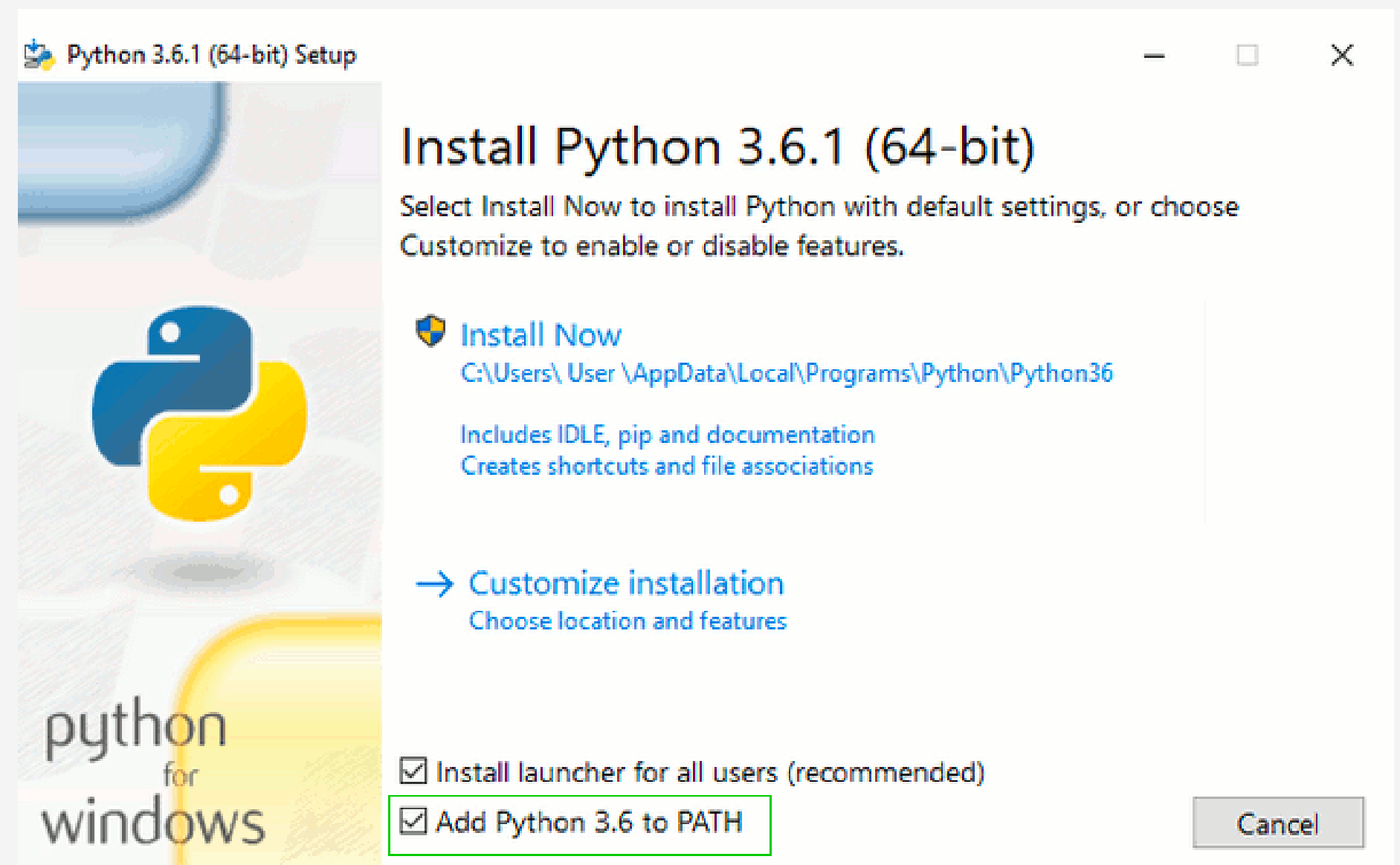
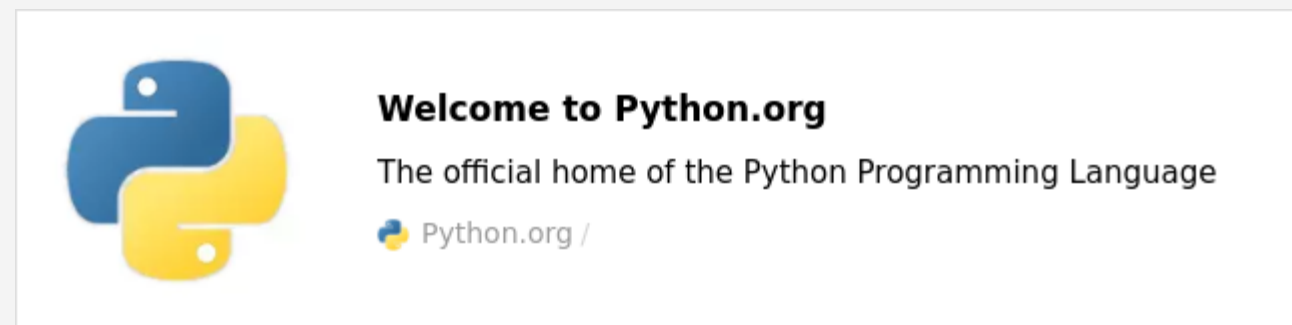
Introducción

- Fue desarrollado por **Guido van Rossum**, un programador holandés, y fue lanzado en **1991**
- Es un **lenguaje interpretado**. Utiliza el intérprete **CPython** para compilar el código de Python en código de bytes.
- La filosofía detrás de Python es que el código debe ser **legible**.
- Dada su practicidad y **robustez** es utilizado en diversos campos



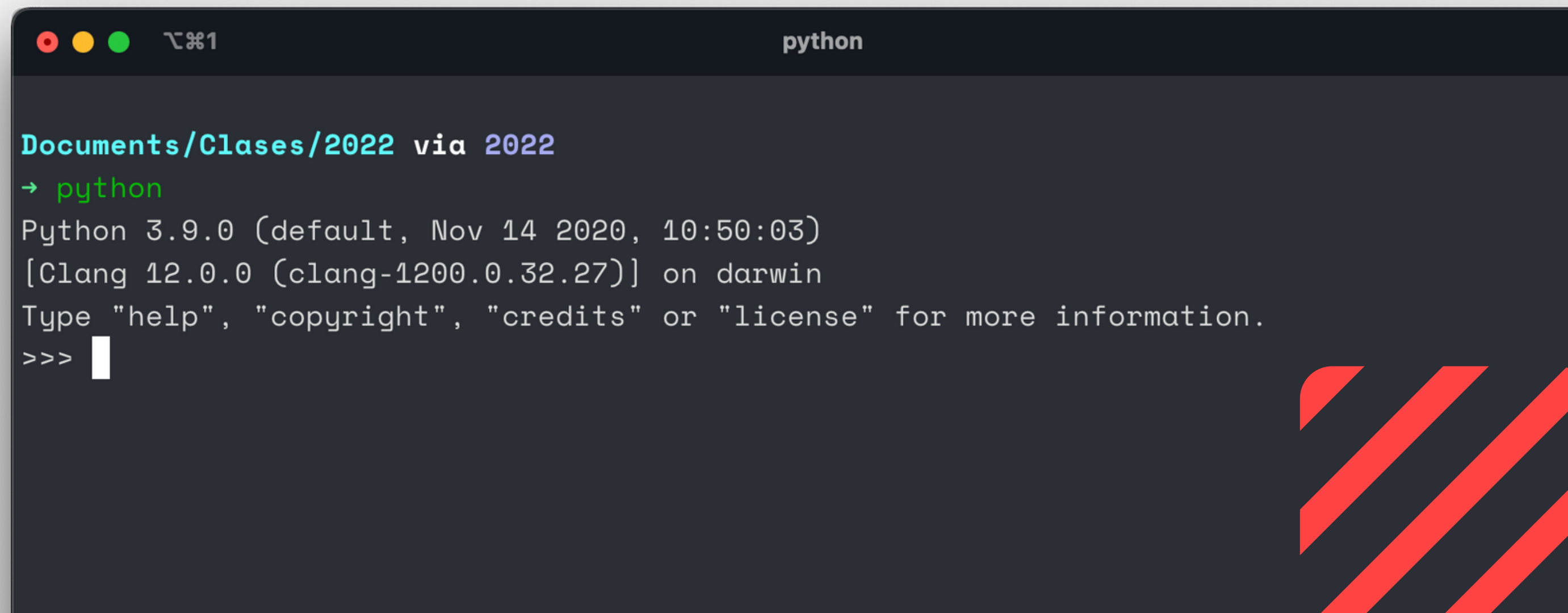
Instalación

- Marca la opción Add Python to PATH para añadirlo a las variables del sistema, esto nos permitirá tener acceso a Python desde cualquier ubicación

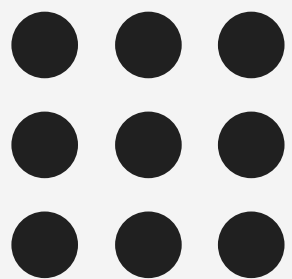


Python interactivo

- En una terminal o símbolo de sistema (CMD), escribe la palabra **python** y presiona enter

A terminal window with a dark background and light text. The title bar shows three colored circles (red, yellow, green) and a keyboard icon. The terminal content shows the path 'Documents/Clases/2022 via 2022', the command 'python' being executed, and the Python 3.9.0 startup message. The prompt '>>>' is followed by a white cursor.

```
Documents/Clases/2022 via 2022
→ python
Python 3.9.0 (default, Nov 14 2020, 10:50:03)
[Clang 12.0.0 (clang-1200.0.32.27)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```



Variables y tipo de datos

```
python
>>> x = 5
>>> x
5
>>> y = 2.1416
>>> y
2.1416
>>> z = "Hola clase arquitecturas programables avanzadas"
>>> z
'Hola clase arquitecturas programables avanzadas'
>>> a = True
>>> a
True
>>> type(x)
<class 'int'>
>>> 
```

Operadores

- Aritméticos
- Lógicos

```
pyt
>>> x * y
10.708
>>> x / 8
0.625
>>> x > y
True
>>> x > y and x == 5
True
>>> x % 2
1
>>> 
```

Condicionales

```
python
>>> if x > y:
...     print("Si soy mayor")
... else:
...     print("Soy menor")
...
Si soy mayor
>>>
>>> if x < 5:
...     print("Soy menor que 5")
... elif x < 10:
...     print("Soy mayor que 5 pero menos que 10")
... elif x < 15:
...     print("Soy mayor que 10 pero menos que 15")
... else:
...     print("Soy un numero gigante")
...
Soy mayor que 5 pero menos que 10
>>> 
```

Loop for

```
python
>>> for i in range(3):
...     print(i)
...
0
1
2
>>> for nombre in alumnos:
...     print(nombre)
...
Miranda
Ivan
Luis
Abril
>>>
>>> for i, nombre in enumerate(alumnos):
...     print(i, nombre)
...
0 Miranda
1 Ivan
2 Luis
3 Abril
>>> 
```

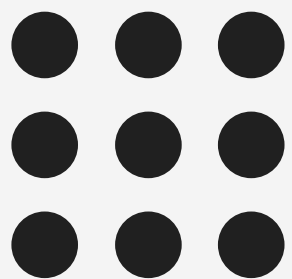

Loop while

```
python
>>> x = 7
>>> while x < 10:
...     print(x)
...     x += 1
...
7
8
9
>>>
>>> i = 0
>>> while i < len(alumnos):
...     print(alumnos[i])
...     i += 1
...
Miranda
Ivan
Luis
Abril
>>> 
```

Listas

- Listas son como arreglos flexibles.
- Pueden ser escritas como una lista de valores separados por coma (ítems) entre corchetes.

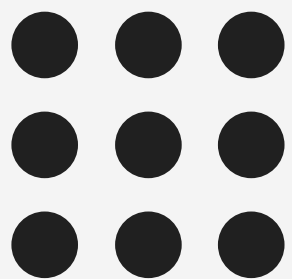
```
python
>>> []
[]
>>> x = [4,2,7,4,2,9,1]
>>> x[0]
4
>>> x[-1]
1
>>> x[2:5]
[7, 4, 2]
>>> 
```



Diccionarios

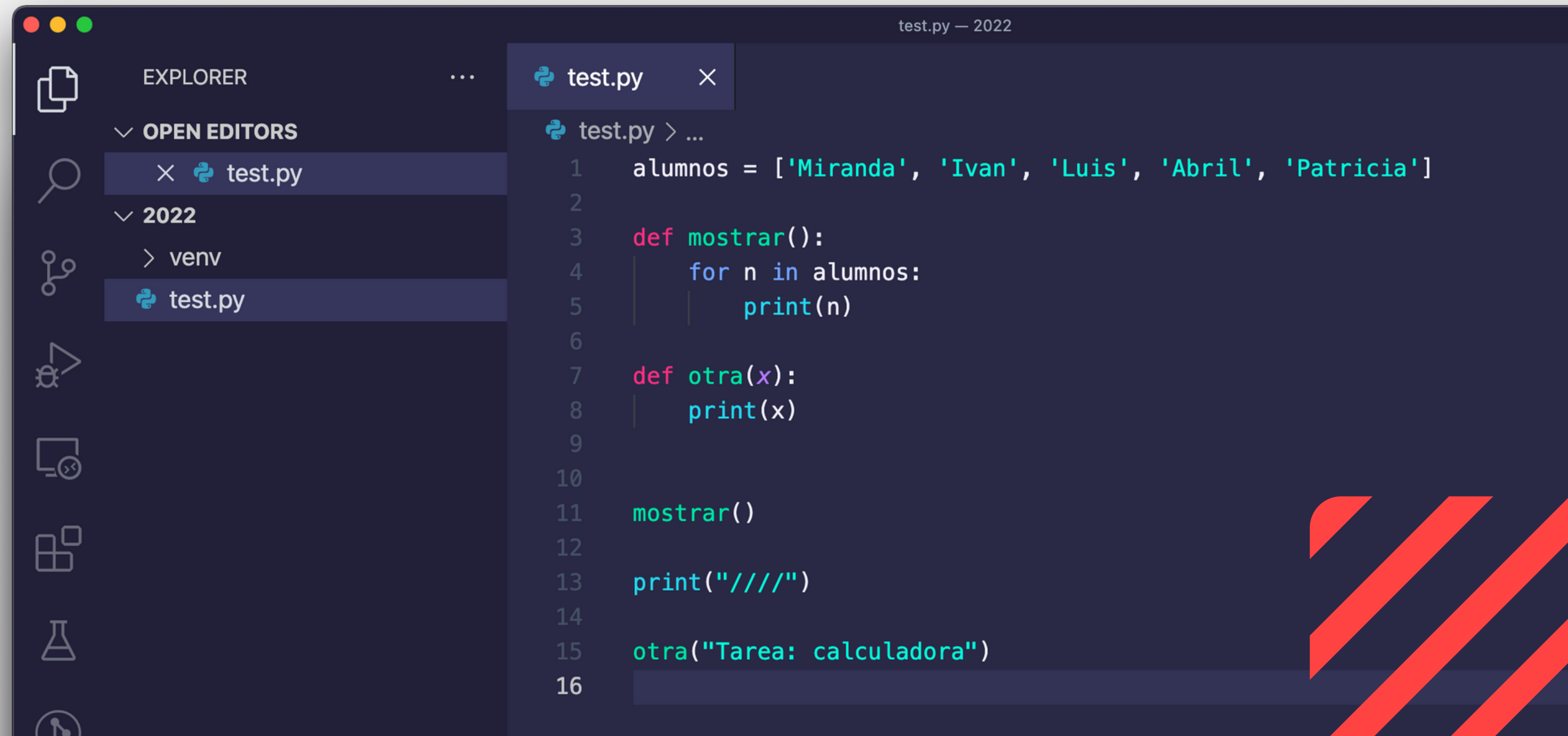
- Un diccionario es similar a una lista, pero accedes a valores usando una llave en vez de un índice.
- Una llave puede ser cualquier cadena o número.

```
python
>>> {}
{}
>>> dic = {'llave': 'valor'}
>>> dic = {'1234': {'nombre': 'Dimas', 'materias': 7}, '4567': {'nombre': 'Rosa', 'materias': 7}}
>>> dic['1234']
{'nombre': 'Dimas', 'materias': 7}
>>> dic['1234']['materias']
7
>>> 
```



Funciones

- Las funciones son un bloque de código que nos ayuda en la reutilización de la lógica repetitiva



```
test.py — 2022

EXPLORER
  OPEN EDITORS
    test.py
  2022
    venv
    test.py

test.py
1  alumnos = ['Miranda', 'Ivan', 'Luis', 'Abril', 'Patricia']
2
3  def mostrar():
4      for n in alumnos:
5          print(n)
6
7  def otra(x):
8      print(x)
9
10
11 mostrar()
12
13 print("////")
14
15 otra("Tarea: calculadora")
16
```

