Sentencias de control

```
n= input("Introduce un número")
numero=int(n)
if numero< 0:
    print ('Negativo')
elif numero> 0:
    print ('Positivo')
else:
    print ('Cero')
```

```
match num: # match case
    case 1:      # casa con un valor simple
        print("Uno")
    case 2 | 3:  # casa usando un or
        print("Dos o tres")
    case num if num < 0:  # casa utilizando un if
        print("Numero negativo")
    case _:     # case por defecto
        print("El número no está entre 1 y 3")
```

Bucles

```
edad=0
while edad <18:
    edad = input("Introduce edad> ")
    edad = int(edad)
```

```
for i in 1, 2, 3, 6:
    print("El valor de i es", i)

for i in range(4):
    print("El valor de i es", i)

for i in secuencia:
    print("El valor i de la sec es", i)
```

Pueden tener una rama else. Se ejecuta siempre, haya entrado o no, mientras el bucle no se termine con un break

```
for x in range(1,10):
    print("*")
else:
    print("fin")
```

Range

1 parámetro	2 parámetros	3 parámetros
`range(5)` Desde 0 hasta el 5 sin incluirlo	`range(1,5)` Desde 1 hasta el 5 sin incluirlo	`range(1,5,2)` Desde 1 hasta el 5 sin incluirlo de dos en dos. Si el tercer parámetro es negativo el rango se crea decreciente.

Break

Salir de un bucle

```
for i in range(10):
    print(i, end=" ")
    if i == 5:
        break
```

Continue

Fuerza a comenzar una nueva iteración

```
for i in range(10):
    if i == 5:
        continue
    print(i, end=" ")
```

Pass

Declaración de relleno

```
def funcion_vacia():
    pass
```

Funciones

```
def nombre_funcion(param1, param2, …):
  cuerpo de la función
  return dato  # Esto es opcional

def press(nombre, apellido):
  print("Hola, soy", nombre, apellido)

Valores por defecto:
def press(nombre, apellido="Graiño"):
```

```
Argumentos en la invocación:
Posicionales:
press ("Lorenzo", "Iglesias")

Con palabra clave
press (nombre="Lorenzo", apellido="Igles")

Combinado ambos. Posicionales al principio
press ("Lorenzo", apellido ="Iglesias")
```

Valor de retorno con return.

None: valor devuelto si no hay return

```
def docu():
    ''' Esto es un docstrint, usado
        Para documentar la función'''
    pass
```

Excepciones

```
try:
    # Código que puede producir la ex.
except:
    # Qué hacemos si se produce la ex.
```

```
try:
    # Código que puede producir la ex.
except ValueError:
    print("hay que introducir un entero")
except:
    # Qué hacemos si se produce la ex
```

finally: se utiliza para ejecutar código al final del try se hayan producido excepciones o no

else: se utiliza para ejecutar código si no se han producido excepciones.

Secuencias

Tipo	Nombre	Descripción	Definición	Mutable
list	Listas	Conjunto de datos que se puede repetir y que pueden ser de distintos tipos.	[v1, v2, ...] []	Si
tuple	tuplas	Iguales que las listas pero inmutables	(v1, v2, ...) () (1,)	No
str	cadenas de caracteres	Permiten guardar secuencias de caracteres.	" " o ' '	No
set	conjuntos	Permiten guardar conjuntos de datos, en los que no se existen repeticiones.	{v1, v2, ...} set()	Si
frozenset	conjuntos	Iguales que los set pero inmutables	frozenset(sec)	No

Métodos

- len(secuencia): devuelve la cantidad de elementos de la lista, tupla o cadena.
- max(secuencia): devuelve el elemento con un valor más alto.
- min(secuencia): devuelve el elemento con un valor más bajo.

Funciones

- secuencia.index(valor): devuelve el índice de la primera aparición de valor.
- secuencia.count(valor): devuelve el número de veces que aparece valor.

Operadores

- in: devuelve verdadero si el elemento pasado pertenece a la secuencia.
- not in: devuelve falso si el elemento pasado no pertenece a la secuencia
- +: une las dos secuencias en una nueva secuencia.
- *: repite una secuencia el número de veces

Desenpaquetado

```
lista = [1, 2, 3, 4, 5]
a, b, c, d, e = lista
```

For en secuencias

```
for color in colores:
    print(color)
```

```
for i in range(len(colores)):
    print(colores[i])
```

Slices

Permite acceder a los elementos de una secuencia usando rangos, sin incluir el fin.

```
secuencia[inicio_rango : fin_rango [: salto]]
```

colores[1:4]	Desde la posición 1 hasta la 3
colores[:4] colores[0:4]	Desde la posición 0 hasta la 3
colores[1:] colores[1:len(colores)]	Desde la posición 1 hasta el final
colores[-2:]	Desde el segundo por el final hasta el final
colores[-6: 5]	Desde el sexto por el final hasta el quinto
colores[:]	Toda la lista
colores[0::2]	Elementos de dos e dos: 0, 2, 4, …
colores[:1:-1]	Al ser el salto negatio: desde el final hasta la posición 2 (el fin no se incluye)
colores[::-1]	Invierte la secuencia

Diccionarios

Son una estructura de datos mutable que permite almacenar pares clave:valor

```
ruedas ={"moto":2, "coche":4, "camion":16, "barco":0}
```