



Programación. Python

Estilo pitónico

Expresiones condicionales

```
expr1 if condition else expr2
```

Manera clásica:

```
def f1(n):  
    if n >= 0:  
        absolute = n  
    else:  
        absolute = -n  
    return absolute**3
```

Mejor:

```
def f2(n):  
    absolute = n if n >= 0 else -n  
    return absolute**3
```

Mejor:

```
def f3(n):  
    return (n if n >= 0 else -n)**3
```

Prueba de funcionamiento:

```
for i in [2, -2, 10, -10]:  
    print(i, "-> ", f1(i), f2(i), f3(i))
```

```
2 -> 8 8 8  
-2 -> 8 8 8  
10 -> 1000 1000 1000  
-10 -> 1000 1000 1000
```

Recorridos de listas

Viejo estilo:

```
def suma_datos(lista):  
    acum = 0  
    n = len(lista)  
    for i in range(n):  
        acum = acum + lista[i]  
    return acum  
  
print(suma_datos([1, 2, 3, 4, 5]))
```

Estilo pitónico:

```
def suma_datos(lista):  
    acum = 0  
    for x in lista:  
        acum = acum + x  
    return acum  
  
print(suma_datos([1, 2, 3, 4, 5]))
```

Más claro aún:

```
def suma_datos(lista):  
    return sum(lista)  
  
print(suma_datos([1, 2, 3, 4, 5]))
```

15

15

15

Notación intensional

Viejo estilo:

```
def lista_de_cuadrados_1(n):  
    lista = []  
    for i in range(n):  
        lista.append(i**2)  
    return lista
```

Estilo pitónico:

```
def lista_de_cuadrados_2(n):  
    return [i**2 for i in range(n)]
```

Prueba de funcionamiento:

```
print(lista_de_cuadrados_1(10))  
print(lista_de_cuadrados_2(10))
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]  
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Notación intensional

Viejo estilo:

```
def lista_de_cuadrados_1(n):  
    lista = []  
    for i in range(n):  
        lista.append(i**2)  
    return lista
```

Estilo pitónico:

```
def lista_de_cuadrados_2(n):  
    return [i**2 for i in range(n)]
```

Prueba de funcionamiento:

```
print(lista_de_cuadrados_1(10))  
print(lista_de_cuadrados_2(10))
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]  
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Viejo estilo:

```
def conj_de_cuadrados_1(n):  
    conj = set()  
    for i in range(n):  
        conj.add(i**2)  
    return conj
```

Estilo pitónico:

Prueba de funcionamiento:

```
print(conj_de_cuadrados_1(10))  
print(conj_de_cuadrados_2(10))
```

```
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}  
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}
```

Notación intensional

Viejo estilo:

```
def lista_de_cuadrados_1(n):  
    lista = []  
    for i in range(n):  
        lista.append(i**2)  
    return lista
```

Estilo pitónico:

```
def lista_de_cuadrados_2(n):  
    return [i**2 for i in range(n)]
```

Prueba de funcionamiento:

```
print(lista_de_cuadrados_1(10))  
print(lista_de_cuadrados_2(10))
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]  
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Viejo estilo:

```
def conj_de_cuadrados_1(n):  
    conj = set()  
    for i in range(n):  
        conj.add(i**2)  
    return conj
```

Estilo pitónico:

```
def conj_de_cuadrados_2(n):  
    return {i**2 for i in range(n)}
```

Prueba de funcionamiento:

```
print(conj_de_cuadrados_1(10))  
print(conj_de_cuadrados_2(10))
```

```
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}  
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}
```

Notación intensional

Viejo estilo:

```
def lista_de_cuadrados_1(n):  
    lista = []  
    for i in range(n):  
        lista.append(i**2)  
    return lista
```

Estilo pitónico:

```
def lista_de_cuadrados_2(n):  
    return [i**2 for i in range(n)]
```

Prueba de funcionamiento:

```
print(lista_de_cuadrados_1(10))  
print(lista_de_cuadrados_2(10))
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]  
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Viejo estilo:

```
def conj_de_cuadrados_1(n):  
    conj = set()  
    for i in range(n):  
        conj.add(i**2)  
    return conj
```

Estilo pitónico:

```
def conj_de_cuadrados_2(n):  
    return {i**2 for i in range(n)}
```

Prueba de funcionamiento:

```
print(conj_de_cuadrados_1(10))  
print(conj_de_cuadrados_2(10))
```

```
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}  
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}
```

Viejo estilo:

```
def dicc_de_cuadrados_1(n):  
    diccionario = dict()  
    for i in range(n):  
        diccionario[i] = i**2  
    return diccionario
```

Estilo pitónico:

Prueba de funcionamiento:

```
print(dicc_de_cuadrados_1(10))  
print(dicc_de_cuadrados_2(10))
```

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25,  
6: 36, 7: 49, 8: 64, 9: 81}  
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25,  
6: 36, 7: 49, 8: 64, 9: 81}
```

Notación intensional

Viejo estilo:

```
def lista_de_cuadrados_1(n):  
    lista = []  
    for i in range(n):  
        lista.append(i**2)  
    return lista
```

Estilo pitónico:

```
def lista_de_cuadrados_2(n):  
    return [i**2 for i in range(n)]
```

Prueba de funcionamiento:

```
print(lista_de_cuadrados_1(10))  
print(lista_de_cuadrados_2(10))
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]  
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Viejo estilo:

```
def conj_de_cuadrados_1(n):  
    conj = set()  
    for i in range(n):  
        conj.add(i**2)  
    return conj
```

Estilo pitónico:

```
def conj_de_cuadrados_2(n):  
    return {i**2 for i in range(n)}
```

Prueba de funcionamiento:

```
print(conj_de_cuadrados_1(10))  
print(conj_de_cuadrados_2(10))
```

```
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}  
{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}
```

Viejo estilo:

```
def dicc_de_cuadrados_1(n):  
    diccionario = dict()  
    for i in range(n):  
        diccionario[i] = i**2  
    return diccionario
```

Estilo pitónico:

```
def dicc_de_cuadrados_2(n):  
    return {i: i**2 for i in range(n)}
```

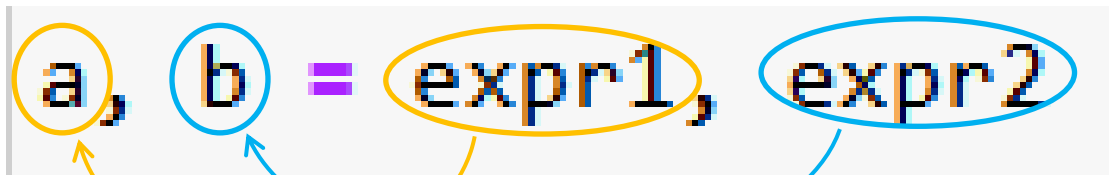
Prueba de funcionamiento:

```
print(dicc_de_cuadrados_1(10))  
print(dicc_de_cuadrados_2(10))
```

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25,  
6: 36, 7: 49, 8: 64, 9: 81}  
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25,  
6: 36, 7: 49, 8: 64, 9: 81}
```


Asignación paralela

`a, b = expr1, expr2`



Intercambio de valores entre dos variables:

Viejo estilo:

```
a = 1
b = 666
print("Iniciales : ", a, b)
```

```
aux = a
a = b
b = aux
```

```
print("Intercambio: ", a, b)
```

Mejor:

`a, b = b, a`



```
print("Intercambio: ", a, b)
```

Asignación paralela

Intercambio de valores entre dos variables:

Viejo estilo:

```
a = 1
b = 666
print("Iniciales : ", a, b)
```

```
aux = a
a = b
b = aux
```

```
print("Intercambio: ", a, b)
```

Mejor:

```
a, b = b, a
```

```
print("Intercambio: ", a, b)
```

Viejo estilo:

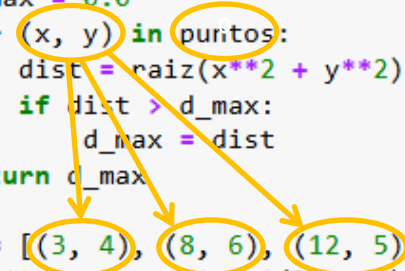
```
from math import sqrt as raiz

def distancia_maxima_1(puntos):
    d_max = 0.0
    for punto in puntos:
        x = punto[0]
        y = punto[1]
        dist = raiz(x**2 + y**2)
        if dist > d_max:
            d_max = dist
    return d_max
```

Estilo pitónico:

```
def distancia_maxima_2(puntos):
    d_max = 0.0
    for (x, y) in puntos:
        dist = raiz(x**2 + y**2)
        if dist > d_max:
            d_max = dist
    return d_max

lista = [(3, 4), (8, 6), (12, 5)]
print(distancia_maxima_1(lista))
print(distancia_maxima_2(lista))
```



13.0

13.0

Asignación paralela

Intercambio de valores entre dos variables:

Viejo estilo:

```
a = 1
b = 666
print("Iniciales : ", a, b)
```

```
aux = a
a = b
b = aux
```

```
print("Intercambio: ", a, b)
```

Mejor:

```
a, b = b, a
```

```
print("Intercambio: ", a, b)
```

Viejo estilo:

```
from math import sqrt as raiz

def distancia_maxima_1(puntos):
    d_max = 0.0
    for punto in puntos:
        x = punto[0]
        y = punto[1]
        dist = raiz(x**2 + y**2)
        if dist > d_max:
            d_max = dist
    return d_max
```

Estilo pitónico:

```
def distancia_maxima_2(puntos):
    d_max = 0.0
    for (x, y) in puntos:
        dist = raiz(x**2 + y**2)
        if dist > d_max:
            d_max = dist
    return d_max

lista = [(3, 4), (8, 6), (12, 5)]
print(distancia_maxima_1(lista))
print(distancia_maxima_2(lista))
```

13.0
13.0

o aún:

```
def distancia_maxima_3(puntos):
    distancias = [raiz(x**2 + y**2) for (x, y) in puntos]
    return max(distancias)

print(distancia_maxima_3(lista))
```



Programación. Python

Estilo pitónico