

¿Cuál es el resultado del siguiente fragmento de código?

```
def scope_test():  
    x = 123  
  
scope_test()  
print(x)
```

NameError: name 'x' is not defined

El alcance del parámetro de una función es la función en sí.

¿Cuál es el resultado del siguiente fragmento de código?

```
def my_function():  
    print("¿Conozco a la variable?", var)
```

```
var = 1  
my_function()  
print(var)
```

¿Conozco a la variable? 1

```
def my_function():  
    var = 2  
    print("¿Conozco a la variable?", var)
```

```
var = 1  
my_function()  
print(var)
```

¿Conozco a la variable? 2
1

Una variable definida fuera de la función, dentro de la función **solo puede ser leída**. **Modificar** su valor hace que **la función cree su propia variable**.

Una variable que existe fuera de una función tiene alcance dentro del cuerpo de la función, EXCEPTO si tiene el mismo nombre que una variable de la función.

¿Cuál es el resultado del siguiente fragmento de código?

```
def my_function():  
    global var  
    var = 2  
    print("¿Conozco a aquella variable?", var)  
  
var = 1  
my_function()  
print(var)
```

¿Conozco a aquella variable? 2
2

Dentro de una función, anteponer la palabra reservada `global` a una variable, obliga a Python a abstenerse de crear una nueva variable dentro de la función: se extiende el alcance de una variable al cuerpo de las funciones

¿Cuál es el resultado del siguiente fragmento de código?

```
def my_function(n):  
    print("Yo recibí", n)  
    n += 1  
    print("Ahora tengo", n)  
  
var = 1  
my_function(var)  
print(var)
```

Yo recibí 1
Ahora tengo 2
1

Cambiar el valor de un parámetro ESCALAR no propaga el cambio fuera de la función

¿Cuál es el resultado del siguiente fragmento de código?

```
def my_function(my_list_1):
    print("Print #1:", my_list_1)
    print("Print #2:", my_list_2)
    my_list_1 = [0, 1]
    print("Print #3:", my_list_1)
    print("Print #4:", my_list_2)
```

```
my_list_2 = [2, 3]
my_function(my_list_2)
print("Print #5:", my_list_2)
```

Print #1: [2, 3]
Print #2: [2, 3]
Print #3: [0, 1]
Print #4: [2, 3]
Print #5: [2, 3]

```
def my_function(my_list_1):
    print("Print #1:", my_list_1)
    print("Print #2:", my_list_2)
    del my_list_1[0] # Presta atención a esta línea.
    print("Print #3:", my_list_1)
    print("Print #4:", my_list_2)
```

```
my_list_2 = [2, 3]
my_function(my_list_2)
print("Print #5:", my_list_2)
```

Print #1: [2, 3]
Print #2: [2, 3]
Print #3: [3]
Print #4: [3]
Print #5: [3]

LISTAS como parámetro:

- ⇒ Cambiar el valor del parámetro no afecta a la lista fuera de la función.
- ⇒ Si se modifica la lista, la lista reflejará el cambio

¿Cuál es el resultado del siguiente fragmento de código?

```
var = 2  
  
def mult_by_var(x):  
    return x * var  
  
print(mult_by_var(7))
```

14

```
def mult(x):  
    var = 5  
    return x * var  
  
print(mult(7))
```

35

```
def mult(x):  
    var = 7  
    return x * var  
  
var = 3  
print(mult(7))
```

49

¿Cuál es el resultado del siguiente fragmento de código?

```
def adding(x):  
    var = 7  
    return x + var  
  
print(adding(4))  
print(var)
```

11

NameError: name 'var' is not defined.

```
var = 2  
print(var)  
  
def return_var():  
    global var  
    var = 5  
    return var  
  
print(return_var())  
print(var)
```

2
5
5

```
a = 1  
  
def fun():  
    global a  
    a = 2  
    print(a)  
  
a = 3  
fun()  
print(a)
```

2
2