Funciones

Gutiérrez Cruz Abel Isaías

Modularidad

Su principio principal es el "Divide y vencerás"

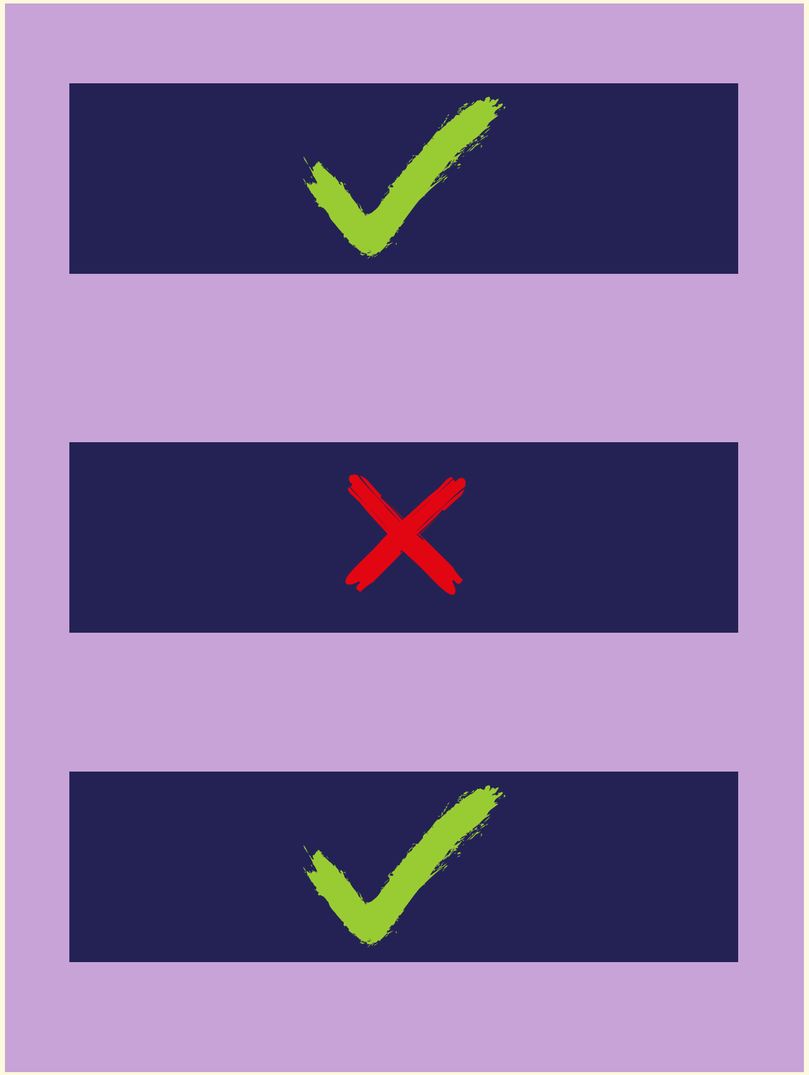
Se trata de un subprograma que se encarga de hacer una tarea en específico. Esta tarea es repetitiva, por lo que cada que el algoritmo principal tenga que llevar a cabo ese procedimiento solo llamará al subprograma.

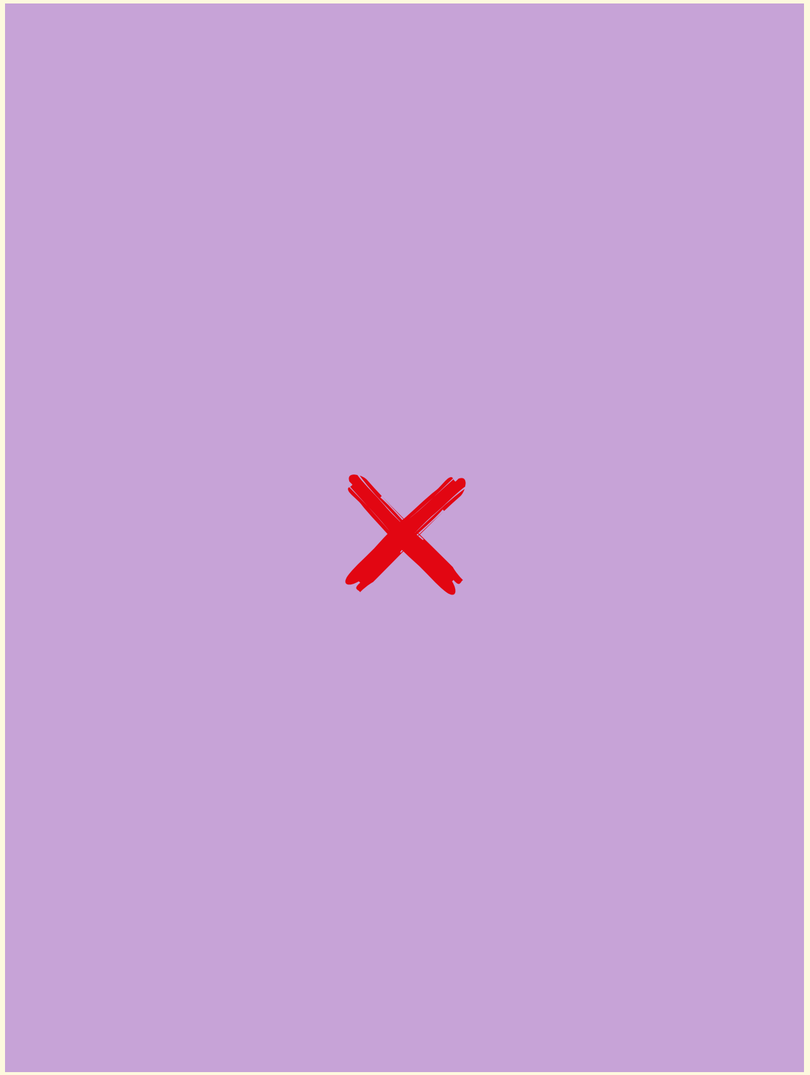
El subprograma recibe los datos del programa principal y puede devolver o no algún resultado.

Ventajas

- Si el algoritmo que estamos diseñando es largo, esta técnica lo vuelve más simple
- Cada modulo se elabora de forma independiente
- Encontrar bugs es más fácil

Modularidad

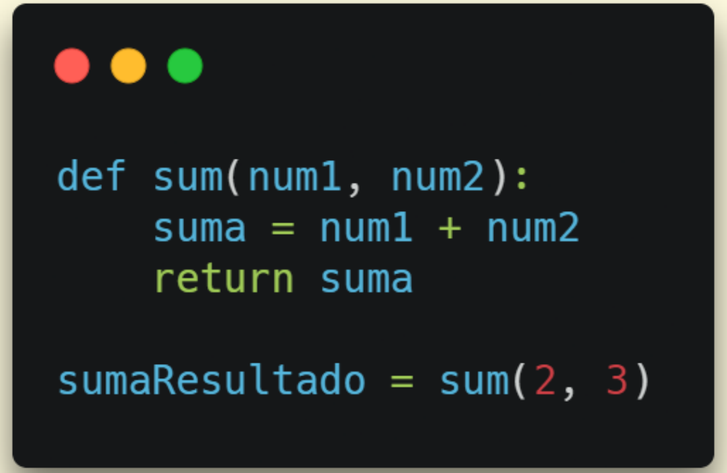




Funciones

```
def <nombre>(<parametros>):
    # Do something
    return <variable> # opcional
```

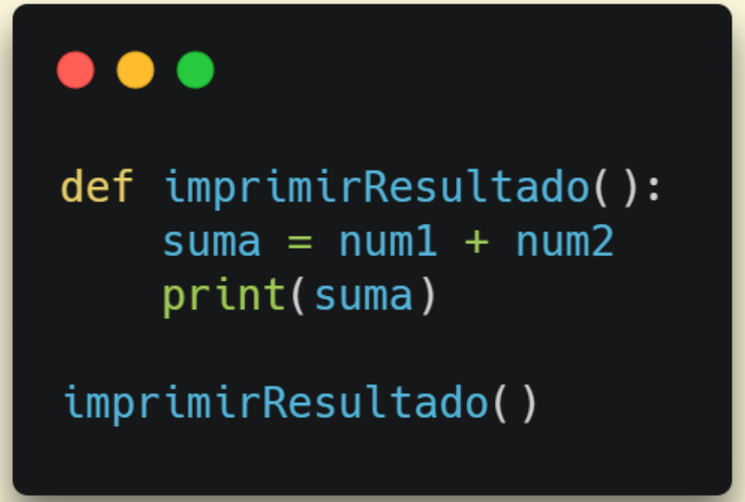
Funciones regrasando elementos



```python
def sum(num1, num2):
    suma = num1 + num2
    return suma

sumaResultado = sum(2, 3)
```

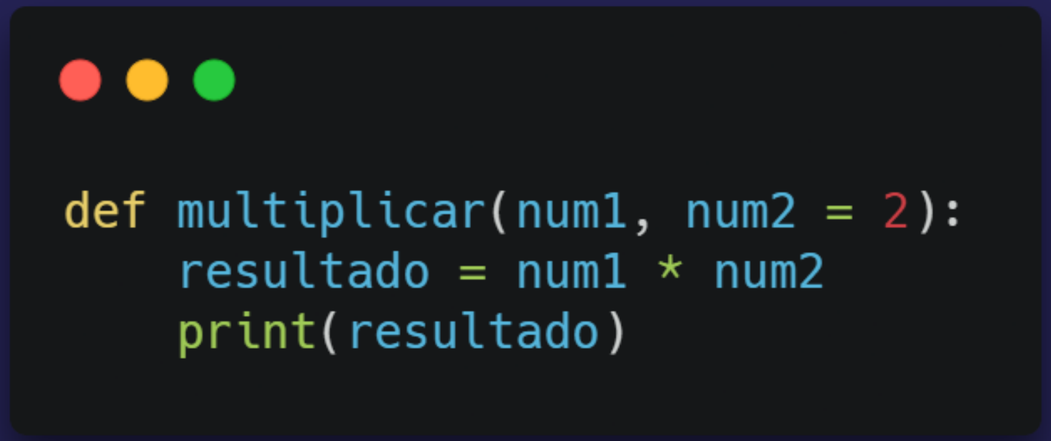
Funciones sin regresar elementos



```
def imprimirResultado():
    suma = num1 + num2
    print(suma)

imprimirResultado()
```

Parametros

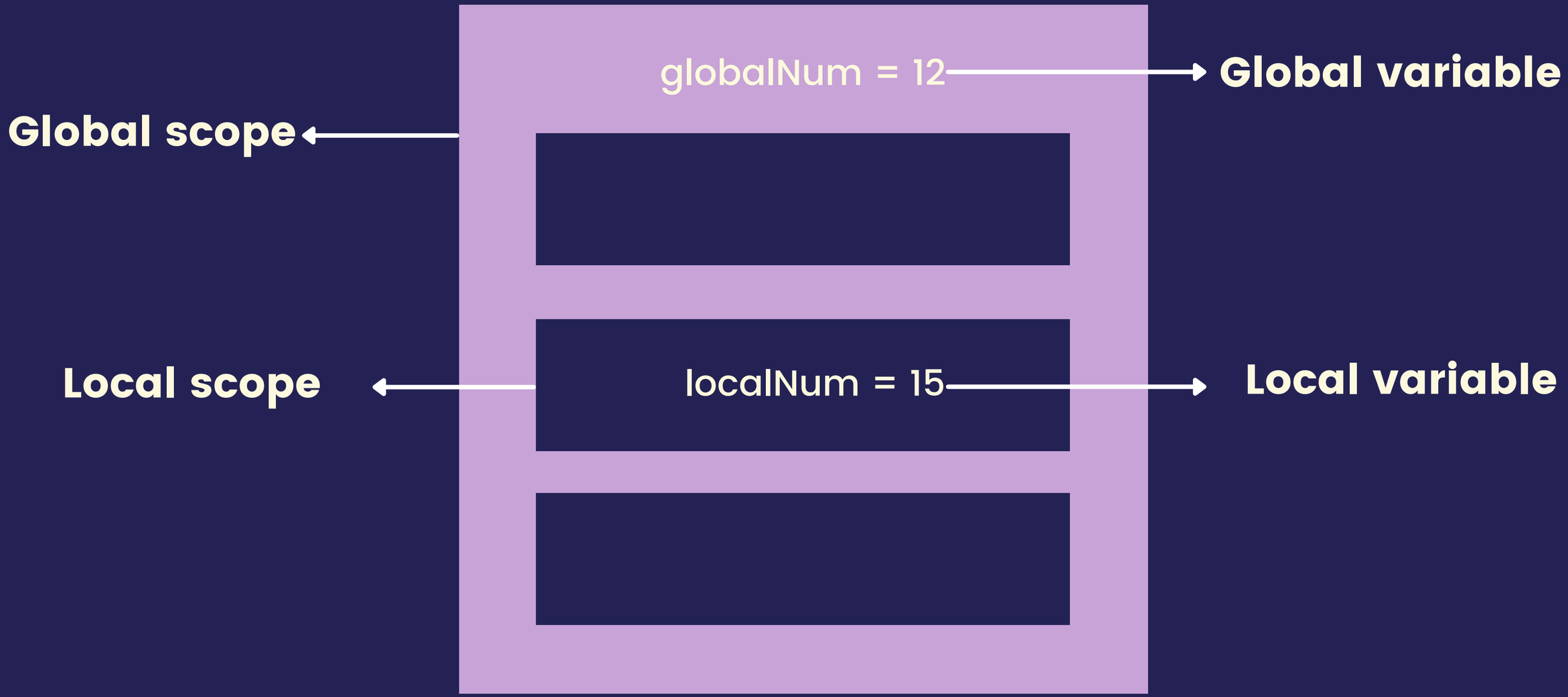


```
def multiplicar(num1, num2 = 2):
    resultado = num1 * num2
    print(resultado)
```

Lambda functions

```
area_triangulo = lambda base,altura:(base*altura)/2
```

Scope


globalNum = 12
Global variable
Global scope
Local scope
localNum = 15
Local variable

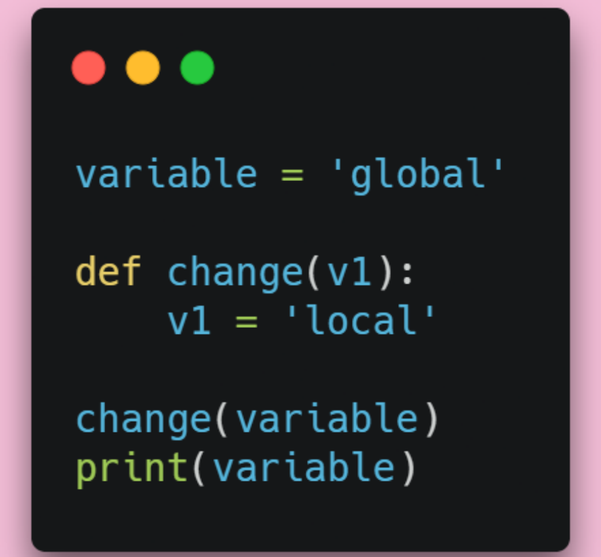
Scope

- Cuando el programa o la función termina todas sus variables son olvidadas
- Un local scope es creada siempre que una función es llamada

Principales principios

- El código en el global scope no puede usar ninguna variable local
- Un local scope puede acceder a las variables globales
- El código en el local scope de una función no puede usar variable de ningun otro local scope

Ejemplo



```
variable = 'global'

def change(v1):
    v1 = 'local'

change(variable)
print(variable)
```

Usar una variable global

Para utilizar una variable global desde una función se puede usar la palabra reservada "global"

```
def spam():
    global eggs
    eggs = 'spam'

eggs = 'global'
spam()
print(eggs)
```