1000 PROGRAMADORES

# Introducción a la **Programación en Python**

Ministerio de Educación Cultura, Ciencia y Tecnología Gobierno de Salta

Ministerio de Economía y Servicios Públicos Gobierno de Salta

Universidad Nacional de Salta

# MÓDULO 5

**Funciones. Funciones integradas.**
**Retorno y envío de valores.**
**Argumentos, parámetros, valor y referencia.**

# Aunque usted no lo crea

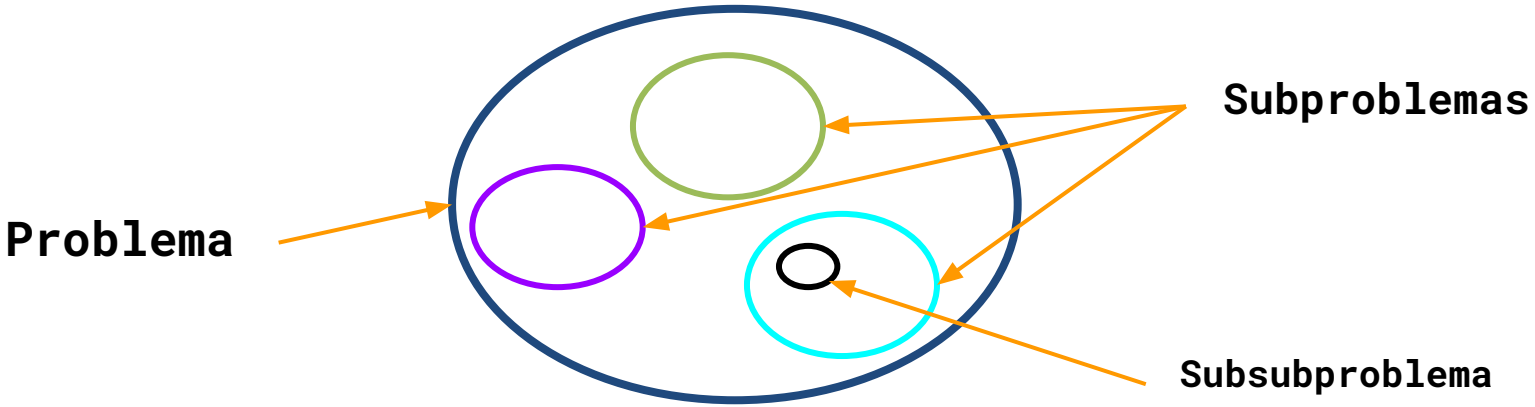
En el año 208 aC, cerca de la Isla de Salamina (Grecia), 1200 barcos de guerra de origen persa, se enfrentaron con las fuerzas navales griegas, inferiores en poderío y número (menos de 400 naves).

A pesar de la diferencia en las fuerzas, los barcos griegos eran más pequeños y, por esa razón, capaces de moverse con mayor agilidad. Los persas, tras 8 horas de combate, iniciaron la retirada. La clave de triunfo griego se debió al uso de embarcaciones más pequeñas pero más manejables que las persas.


*Este es uno de los numerosos ejemplos de una estrategia aplicable al ámbito de la guerra, la política, la economía y las ciencias, que se conoce como* ***"divide y vencerás"***.

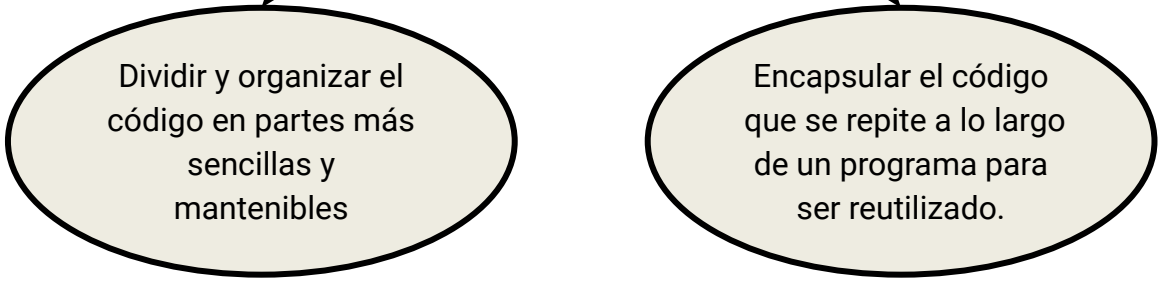
# MODULARIDAD

Para resolver problemas complejos y/o de gran tamaño es conveniente descomponerlos en subproblemas, y de ser necesario, éstos a su vez en problemas más pequeños todavía hasta que el problema original quede reducido a un conjunto de actividades básicas.



# FUNCIONES

***Una función es un grupo de instrucciones que constituyen una unidad lógica del programa y resuelven un problema muy concreto.***

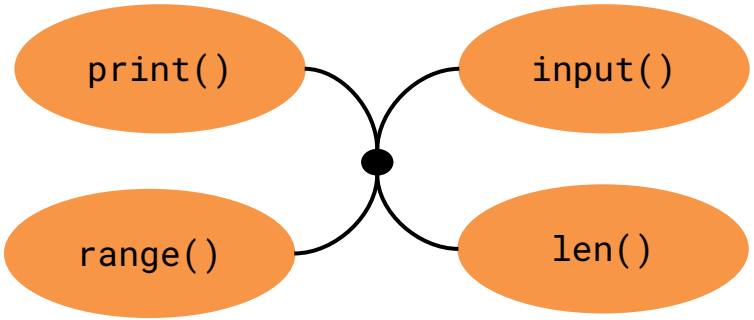



**principio de modularidad**

**principio de reusabilidad**

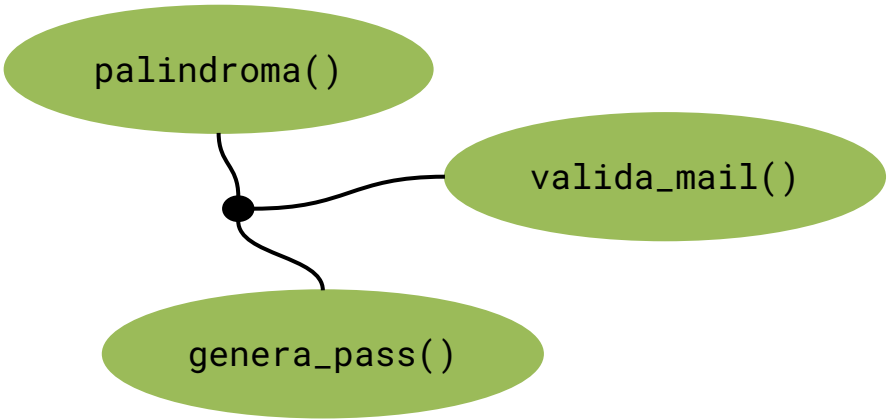
## Funciones Integradas

conjunto de funciones que podemos utilizar directamente en nuestras aplicaciones

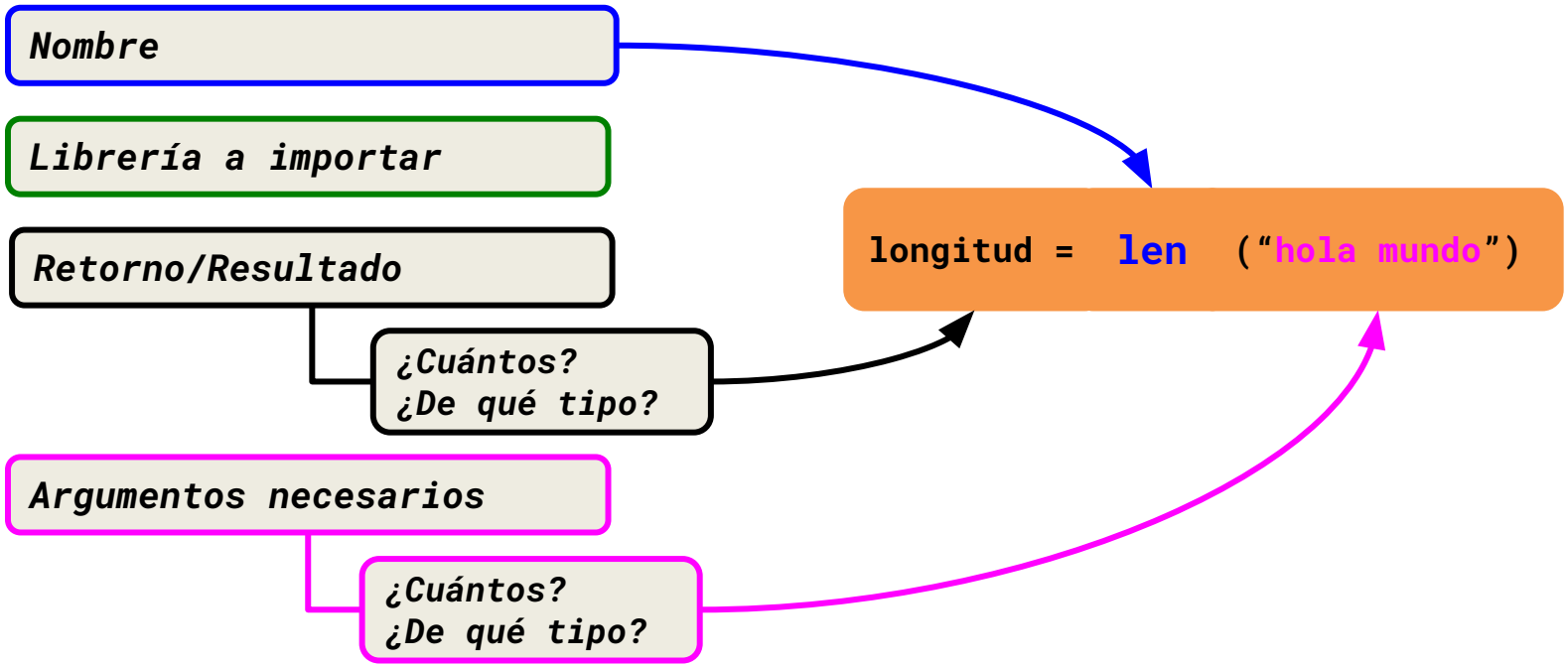



## Funciones definidas por el programador

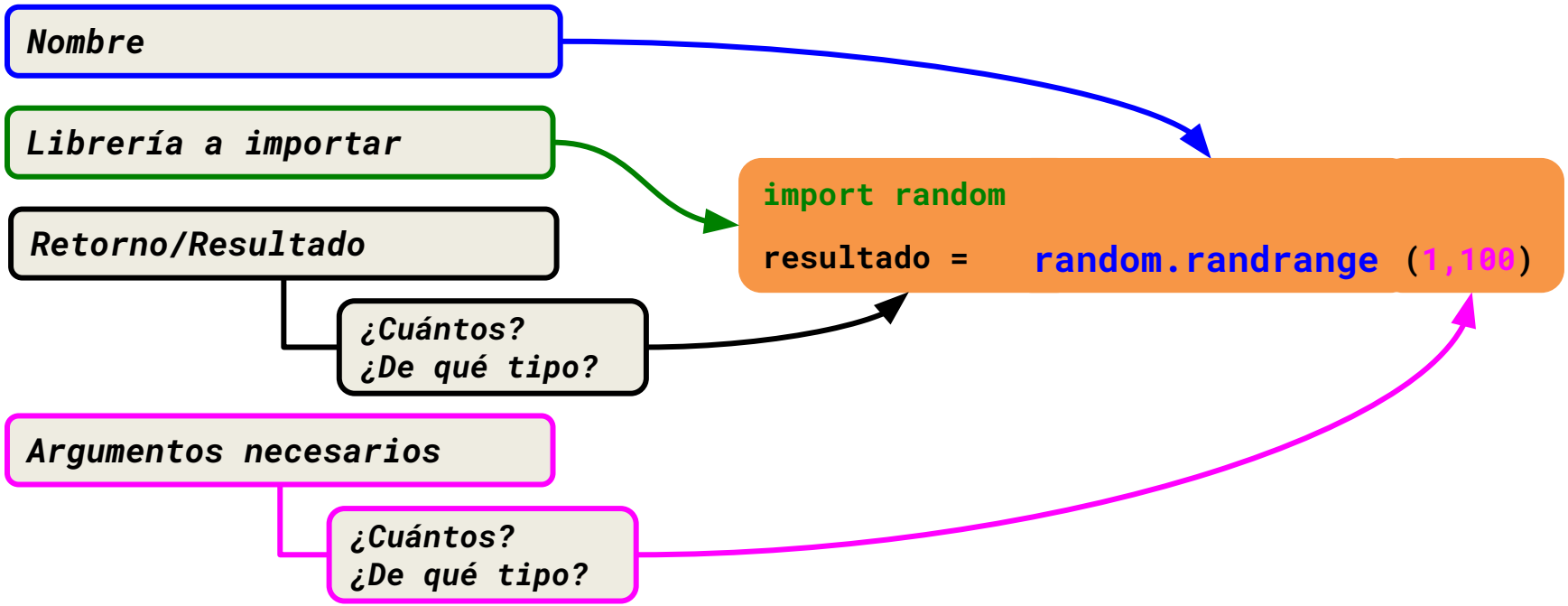
conjunto de funciones que deben definirse e implementarse antes de poder ser utilizadas



# FUNCIONES PREDEFINIDAS EN PYTHON



# FUNCIONES PREDEFINIDAS EN PYTHON



**Ejemplo:**

Dado el nombre, apellido y dni de una persona, se desea generar un usuario y contraseña para ser utilizados en una aplicación.

El nombre de usuario tendrá 8 caracteres, debe ser generado usando las 3 primeras letras del nombre, las 3 primeras letras del apellido y las últimas 2 cifras del dni.

La contraseña tendrá también 8 caracteres, debe generarse usando un número aleatorio de 3 cifras seguido de la suma de los ultimos 3 digitos del dni seguido de la suma de los siguientes 3 dígitos del dni seguido del mayor dígito del dni.

Nota 1: Si la suma resulta menor a 10 debe anteponer un 0.
Nota 2: Todos los caracteres deben estar escritos en mayúsculas.

Caso de prueba 1:

Nombre: Luis
Apellido: Alvarez
DNI: 1998129

usuario: LUIALV29
pass: 10012269

Nombre: ana
Apellido: velarde
DNI: 30112111

usuario: ANAVEL11
pass: 23103043

## Solución

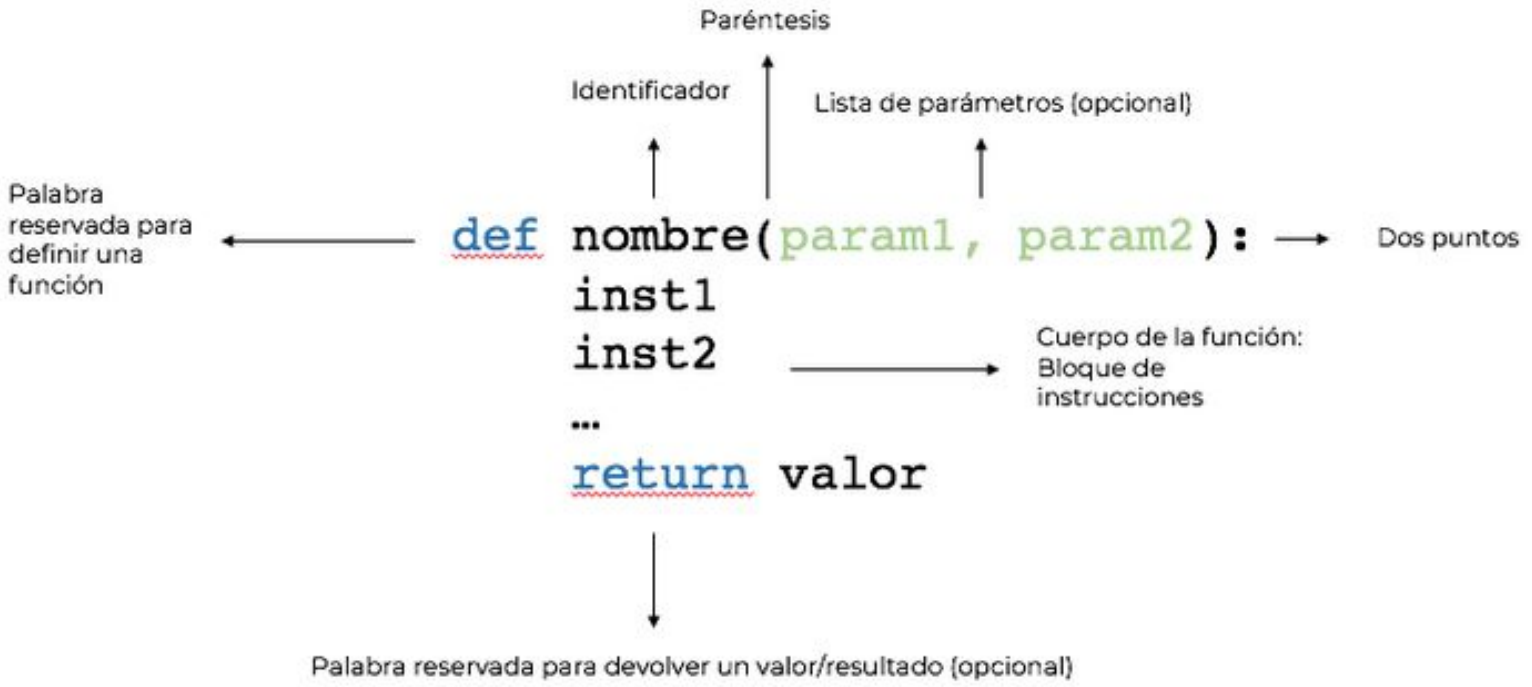
```
import random

nombre=input("Ingresar nombre: ")
apellido=input("Ingresar apellido: ")
dni=input("Ingresar dni: ")

usu=''
for i in range(0,3):
 usu= usu + nombre[i]
for i in range(0,3):
 usu=usu+apellido[i]
longitud=len(dni)-1
usu=usu + dni[longitud-1] + dni[longitud]
usu=usu.upper()
print(usu)
```

```
x=random.randrange(100, 1000)
num1=0
for i in range(longitud,longitud-3,-1):
 num1=num1+int(dni[i])
num1=str(num1)
if (len(num1)==1):
 num1='0'+num1
num2=0
for i in range(longitud-3,longitud-6,-1):
 num2=num2+int(dni[i])
num2=str(num2)
if (len(num2)==1):
 num2='0'+num2
may=max(dni)
num=str(x)+num1+num2+may
print(num)
```

# ESQUEMA PARA DEFINIR UNA FUNCIÓN



**Ejemplo:**

Diseñe una función que indique la cantidad de palabras de un texto.

```
#definición de la función
def contar_palabras(texto):
 cant=1
 for caracter in texto:
   if (caracter==' '):
     cant=cant+1
 return cant
#invocación de la función
c1=contar_palabras("mi mama me mima")
print(c1)
c2=contar_palabras("esta cadena tiene 5 palabras")
print(c2)
```

**Ejemplo:**

Diseñe una función que indique la cantidad de números pares de una lista.

```
#definición de la función
def contar_pares(lista):
 cant=0
 for num in lista:
   if (num%2==0):
     cant=cant+1
 return cant
```

```
#invocación de la función
l=[1, 2, 3]
c=contar_pares(l)
print(c)
```

**Ejemplo:**

Diseñe una función que indique la cantidad de palabras y la cantidad de vocales de un texto.

```
#definición de la función
def contar(texto):
 cant_pal=1
 cant_voc=0
 for car in texto:
   if (car==' '):
     cant_pal=cant_pal+1
   elif (car=='a' or car=='e' or car=='i' or car=='o' or car=='u'):
      cant_voc=cant_voc+1
 return cant_pal, cant_voc
```

```
#invocación de la función
c1=contar("mi mama me mima")
print(c1)
c2=contar("esta cadena tiene 5 palabras")
print(c2)
```

**Ejemplo:**

Dado el nombre, apellido y dni de una persona, se desea generar un usuario y contraseña para ser utilizados en una aplicación.

El nombre de usuario tendrá 8 caracteres, debe ser generado usando las 3 primeras letras del nombre, las 3 primeras letras del apellido y las últimas 2 cifras del dni.

La contraseña tendrá también 8 caracteres, debe generarse usando un número aleatorio de 3 cifras seguido de la suma de los ultimos 3 digitos del dni seguido de la suma de los siguientes 3 dígitos del dni seguido del mayor dígito del dni.

Nota 1: Si la suma resulta menor a 10 debe anteponer un 0.
Nota 2: Todos los caracteres deben estar escritos en mayúsculas.

## Solución MODULARIZADA

```
import random

def primeras_tres(palabra):
 pal=''
 for i in range(0,3):
   pal= pal + palabra[i]
 return pal

def sumar_tres(numero,largo):
 num1=0
 for i in range(largo,largo-3,-1):
   num1=num1+int(dni[i])
 num1=str(num1)
 if (len(num1)==1):
   num1='0'+num1
 return num1
```

```
nombre=input("Ingresar nombre: ")
apellido=input("Ingresar apellido: ")
dni=input("Ingresar dni: ")

usu=primeras_tres(nombre)+primeras_tres(apellido)

longitud=len(dni)-1
usu=usu + dni[longitud-1] + dni[longitud]
usu=usu.upper()
print(usu)

x=random.randrange(100, 1000)

num1=sumar_tres(dni,len(dni)-1)
num2=sumar_tres(dni,len(dni)-4)

may=max(dni)
num=str(x)+num1+num2+may
print(num)
```

**Una función puede definir, opcionalmente, una secuencia de parámetros que poseen una característica muy importante.**

## *Parámetros por valor*

Lo que hace es copiar el valor de las variables en los respectivos parámetros. Una modificación del valor del parámetro, no afecta a la variable externa correspondiente.

## *Parámetros por referencia*

Lo que hace es copiar en los parámetros la dirección de memoria de las variables que se usan como argumento. Una modificación del valor en el parámetro afectará a la variable externa correspondiente.

*Python no sigue estas reglas*

# Python

distingue

## Valores INMUTABLES

Los valores simples como enteros, flotantes, cadenas y lógicos NO reflejan cambios cuando son pasados como parámetros y sufren cambios dentro de la función.

## Valores MUTABLES

Los valores compuestos o colecciones como listas, diccionarios y conjuntos SÍ reflejan cambios cuando son pasados como parámetros y sufren modificaciones dentro de la función.

**Ejemplo:**

Diseñe una función que retorne el sucesor de un número entero.

x conserva su valor 5 aunque sufra modificación en la función

```
def sucesor(x):
  x=x+1
  return x

x=5
num=sucesor(x)
print(x)
print(num)
```

**Ejemplo:**

Diseñe una función que retorne el sucesor del primer número entero de una lista.

x es modificado en su valor original, dado que lo afecta el cambio realizado en la función

```
def sucesor_lis(x):
  x[0]=x[0]+1
  return x

x=[5, 8, 1]
num=sucesor_lis(x)
print(x)
print(num)
```

# ¿JUGUEMOS?

**Adivina un número.**

Diseñemos un script que genere un número entre 1 y 100. El objetivo es que el usuario adivine el número. Si falla se debe indicar si es mayor o menor que el número buscado. También debe contar la cantidad de intentos hasta ganar.

# Solución

```
import random
cant=0
x=random.randrange(1,100)
num=int(input("Ingrese un número: "))
while(x!=num):
  cant=cant+1
  if (x>num):
    mensaje="mayor"
  else:
    mensaje="menor"
  num=int(input("Ingrese un número "+mensaje+" a "+str(num)+" "))
print("FELICIDADES, ganaste despues de "+str(cant)+" intentos")
```

# ¿Le ponemos pimienta?

## Adivina un número.

Diseñemos una función que genere un número entre 1 y 100. El objetivo es que el usuario adivine el número. Si falla se debe indicar si es mayor o menor que el número buscado. También debe contar la cantidad de intentos hasta ganar.

Luego diseñemos un script que permita jugar a dos participantes, debe pedir el nombre de los jugadores y felicitar a quien adivine más rápido.

Nota: Juegan de a uno a la vez.

# Solución -> Función

```
import random
def juego():
 cant=0
 x=random.randrange(1,100)
 num=int(input("Ingrese un número: "))
 while(x!=num):
   cant=cant+1
   if (x>num):
     mensaje="mayor"
   else:
     mensaje="menor"
   num=int(input("Ingrese un número "+mensaje+" a "+str(num)+" "))
 return cant
```

# Solución -> Script

```
jugador1=input("ingrese nombre de jugador 1: ")
jugador2=input("ingrese nombre de jugador 2: ")
resultado1=juego()
resultado2=juego()
if (resultado1>resultado2):
 print("FELICIDADES, "+jugador2+" GANASTE")
 print("Adivinaste en",resultado2,"versus",resultado1)
else:
 print("FELICIDADES, "+jugador1+" GANASTE")
 print("Adivinaste en",resultado1,"versus",resultado2)
```