# **Conceptos de funciones – Programación Estructurada**

Hemos trabajado con una metodología de programación lineal. Todas las instrucciones de nuestro archivo\*.py se ejecutan en forma secuencial de principio a fin.

Esta forma de organizar un programa solo puede ser llevado a cabo si el mismo es muy pequeño (decenas de líneas)

Cuando los problemas a resolver tienen a ser más grandes, la metodología de programación lineal se vuelve ineficiente y compleja.

**El segundo paradigma de programación que veremos en la programación estructurada.**

La programación estructurada busca dividir o descomponer un problema complejo en pequeños problemas. La solución de cada uno de esos pequeños problemas nos trae la solución del problema complejo.

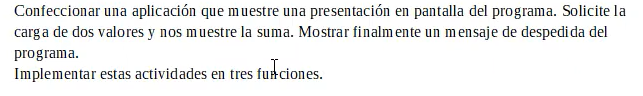
En Python el planteo de esas pequeñas soluciones al problema complejo se hace dividiendo el programa en funciones.

Una función es un conjunto de instrucciones en Python que resuelven un problema específico.

El lenguaje Python ya tiene incorporada algunas funciones básicas. Algunas de ellas ya las utilizamos en conceptos anteriores como son las funciones: print, len y range.

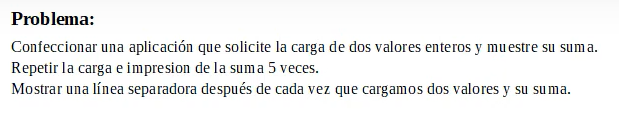
Veamos ahora como crear nuestras propias funciones.

**Problema 1:**

****

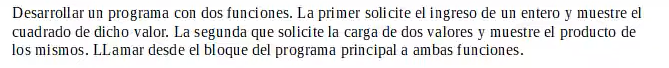
def presentacion():  
 print("programa que permite cargar por teclado dos valores")  
 print("efectua la suma de los valores")  
 print("muestra el resultado de la suma")  
 print("-"\*20)  
  
def carga\_suma():  
 valor1 =int(input("Ingrese primer valor: "))  
 valor2 =int(input("Ingrese segundo valor: "))  
 suma = valor1 + valor2  
 print(f"la suma de los 2 valores es: {suma}")  
def finalizacion():  
 print("Gracias por utilizar este programa")  
  
presentacion()  
carga\_suma()  
finalizacion()

**Problema 2:**

****

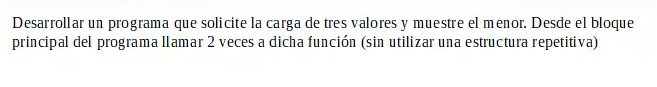
def texto():  
 print(f"solicitar la carga de dos numeros enteros de {i+1} de 5 cargas en total")  
def sumadef():  
 valor1 = int(input("Ingrese: "))  
 valor2 =int(input("Ingrese: "))  
 suma= valor1 + valor2  
 print(suma)  
for i in range(5):  
 texto()  
 sumadef()

**Problema 3:**

****

def primero():  
 valor = int(input("Ingresa un numero entero: "))  
 print(f"El {valor} al cuadrado es {pow(valor,2)}")  
   
def segundo():  
 acumulador = 1  
 final = 7  
 for i in range(final):  
 valor=int(input(f"Ingresa un valor {i+1} de {final}: "))  
 acumulador\*=valor  
 print(f"El valor de los {final} numeros es: {acumulador}")  
  
primero()  
segundo()

**Problema 4:**

****

def primer():  
 final = 3  
 numeros = []  
 for i in range(final):  
 valor= int(input(f"Ingrese un valor {i+1} de {final}"))  
 numeros.append(valor)  
 menor = numeros[0]  
 for i in range(len(numeros)):  
 if numeros[i]<menor:  
 menor = numeros[i]  
 print(f"El menor es: {menor}")  
  
 mayor=numeros[i]  
 for i in range(len(numeros)):  
 if numeros[i]>mayor:  
 mayor = numeros[i]  
 print(f"El mayor es: {mayor}")  
  
  
#Sin estructura for  
  
def segundo():  
 valor1 = int(input("Ingresa 1: "))  
 valor2 = int(input("Ingresa 2: "))  
 valor3 = int(input("Ingresa 3: "))  
  
 if valor1 > valor2 and valor1 >valor3:  
 print(f"El mayor es: {valor1}")  
 elif valor2>valor1 and valor2 >valor3:  
 print(f"El mayor es: {valor2}")  
 else:  
 print(f"El mayor es: {valor3}")  
 if valor1 <valor2 and valor1 <valor3:  
 print(f"El menor valor es: {valor1}")  
 elif valor2 < valor1 and valor2 < valor3:  
 print(F"El menor valor es: {valor2}")  
 else:  
 print(f"El menor valor es: {valor3}")  
  
segundo()

# **Funciones: parámetros**

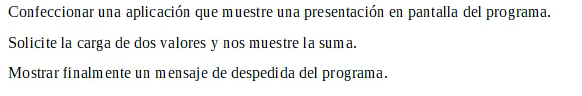
Vimos el concepto anterior que una función resuelve una parte de nuestro algoritmo.

Tenemos por un lado la declaración de la función por medio de un nombre y el algoritmo de la función seguidamente. Luego para que se ejecute la función la llamamos desde el bloque principal de nuestro programa.

Ahora veremos que una función puede tener parámetros para recibir datos. Los parámetros nos permiten comunicarle algo a la función y la hace más flexible.

Una función con parámetros nos hace mas flexible la misma para utilizarla en distintas circunstancias.

**Problema1:**

****

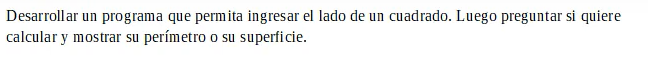
def mensaje(mensaje):  
 print("-"\*20)  
 print(mensaje)  
 print("-"\*20)  
def carga\_suma():  
 acumulador = 0  
 valor1 = int(input("Ingrese primer valor: "))  
 acumulador+=1  
 valor2 = int(input("Ingrese segundo valor: "))  
 acumulador+=1  
 print(f"La suma de los {acumulador} valores es {valor1+valor2}")  
  
mensaje("El programa calcula la suma de dos valores ingresados")  
carga\_suma()  
mensaje("Programa finalizado")

**Problema2:**

****

def MostrarValores(n1,n2,n3):  
 print("El mayor de los 3 numeros es: ")  
 if n1>n2 and n1>n3:  
 print(n1)  
 elif n2>n1 and n2>n3:  
 print(n2)  
 else:  
 print(n3)  
def Mensajedef():  
 print("El menor de los 3 es: ")  
def Menores(n1,n2,n3):  
 Mensajedef()  
 if n1<n2 and n1<n3:  
 print(n1)  
 elif n2<n1 and n2<n3:  
 print(n2)  
 else:  
 print(n3)  
  
def Cargar(Mensaje):  
 print(Mensaje)  
 valor1 = int(input("Ingrese un valor: "))  
 valor2 = int(input("Ingrese un valor: "))  
 valor3 = int(input("Ingrese un valor: "))  
 MostrarValores(valor1,valor2,valor3)  
 Menores(valor1,valor2,valor3)  
  
Cargar("Bienvenio")

**Problema3:**

****

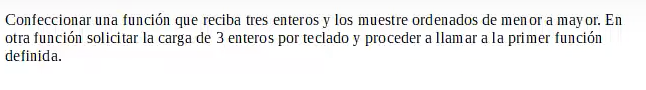
def CalcularPerimetro(n1):  
 print(f"El petrimetro es: {n1\*4}")  
def CalcularSuperficie(n1):  
 print(f"La superficie es: {n1\*n1}")  
  
def cargar\_lado(mensaje):  
 valor = int(input(mensaje))  
 opcion = int(input(f"Quieres calcular el perimetro (1) o el area? (2): "))  
 if opcion == 1:  
 CalcularPerimetro(valor)  
 elif opcion == 2:  
 CalcularSuperficie(valor)  
cargar\_lado("Ingrese un valor: ")

**Problema4:**

****

def Cargar(cadena):  
 cant = 0  
 evaluar = input(cadena)  
 for i in evaluar:  
 if i == "a" or i == "e" or i == "i" or i == "o" or i == "u":  
 cant+=1  
 print(f"La cantidad de volcales son: {cant}")  
for i in range(3):  
 Cargar("Ingresa ctm: ")

**Problema5:**

****

def ordernar(n1,n2,n3):  
 if n1<n2 and n1<n3:  
 if n2<n3:  
 print(n1,n2,n3)  
 else:  
 print(n1,n3,n2)  
 elif n2<n1 and n2<n3:  
 if n1<n3:  
 print(n1,n2,n3)  
 else:  
 print(n2,n3,n1)  
 else:  
 if n1<n2:  
 print(n3,n1,n2)  
 else:  
 print(n3,n2,n1)  
  
def cargar():  
 num1 = int(input("Ingresa: "))  
 num2 = int(input("Ingresa: "))  
 num3 = int(input("Ingresa: "))  
 ordernar(num1,num2,num3)  
cargar()

# **Funciones: retorno de datos**

Hemos comenzado a pensar con la metodología de programación estructurada. Buscamos dividir un problema en subproblemas y plantear algoritmos en Python que los resuelvan.

Vimos que una función la definimos mediante un nombre y que puede recibir datos por medios de sus parámetros.

Los parámetros son la forma que una función reciba datos para ser procesados. Ahora veremos otra característica de las funciones que es la de devolver un dato a quien convoco la función (recordemos que una función la podemos llamar desde el bloque principal de nuestro programa o desde otra función que desarrollemos)

**Problema 1:**

****

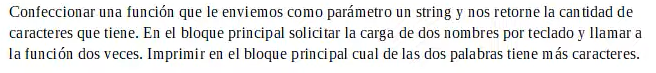
def RetornarSuperficie(lado):  
 superficie = lado\*2  
 return superficie  
  
#Bloque principal  
lado = int(input("Ingresa el valor del cuadrado: "))  
superficie = RetornarSuperficie(lado)  
print(f"La superficie del cuadrado es: {superficie}")  
  
lado = int(input("Ingresa el valor del cuadrado: "))  
print("La superficie del cuadrado es: ",RetornarSuperficie(lado))  
  
lado = int(input("Ingresa el valor del cuadrado: "))  
if RetornarSuperficie(lado) == 100:  
 print("La superficie es 100")  
else:  
 print("La superficie no es 100")

**Problema 2:**



def retornar\_mayor(v1,v2):  
 if v1>v2:  
 mayor = v1  
 else:  
 mayor = v2  
 return mayor  
  
valor1 = int(input("Ingrese primer valor: "))  
valor2 = int(input("Ingrese segundo valor: "))  
print(retornar\_mayor(valor1,valor2))  
  
# otra forma  
  
def retornar\_mayor(v1,v2):  
 if v1>v2:  
 return v1  
 else:  
 return v2  
valor1 = int(input("Ingrese primer valor: "))  
valor2 = int(input("Ingrese segundo valor: "))  
x = retornar\_mayor(valor1,valor2)  
print(x)

**Problema 3:**

****

def largo(cadena):  
 return len(cadena)  
#Bloque principal  
  
nombre1 = input("Ingrese primer nombre: ")  
nombre2 = input("Ingrese segundo nombre: ")  
largo1 = largo(nombre1)  
largo2 = largo(nombre2)  
if largo1 == largo2:  
 print("Los nombres tienen la misma cantidad de caracteres")  
elif largo1>largo2:  
 print(nombre1,"Es mas largo")  
elif largo2>largo1:  
 print(nombre2,"Es mas largo")

**Problema 4:**

****

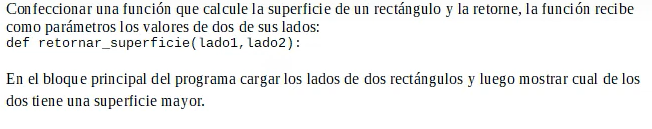
def retornar\_promedio(v1,v2,v3):  
 promedio = (v1 + v2 + v3) / 3  
 return promedio  
  
valor1 = int(input("Ingrese primer valor: "))  
valor2 = int(input("Ingrese segundo valor: "))  
valor3 = int(input("Ingrese tercer valor: "))  
print("El promedio de los 3 valores es: ",retornar\_promedio(valor1,valor2,valor2))

**Problema 5:**

****

def retornar\_perimetro(lado):  
 perimetro = lado \* 4  
 return perimetro  
  
lado = int(input("Ingrese el lado de un cuadrado: "))  
print(f"El perimetro del cuadrado es: ",retornar\_perimetro(lado))

**Problema6:**

****

def retornar\_superficie(lado1, lado2):  
 superficie = lado1 \* lado2  
 return superficie  
  
print("Primer rectangulo")  
lado1 = int(input("Ingrese lado menor del rectangulo: "))  
lado2 =int(input("Ingrese lado mayor del rectangulo: "))  
print("Segundo rectangulo")  
lado3 = int(input("Ingrese lado menor del rectangulo: "))  
lado4 = int(input("Ingrese lado mayor del rectangulo: "))  
  
area1 = retornar\_superficie(lado1, lado2)  
area2 = retornar\_superficie(lado3, lado4)  
  
if area1 == area2:  
 print("Tienen la misma superficie")  
elif area1>area2:  
 print(f"El area 1 tiene mayor superficie que es: {area1}")  
elif area1<area2:  
 print(f"El area 2 tiene mayor superficie que es: {area2}")

**Problema7:**

****

def cantidad\_vocales(palabras):  
 cant = 0  
 for x in range(len(palabras)):  
 if palabras[x]== "a" or palabras[x]=="A":  
 cant = cant + 1  
 return cant  
  
palabra = input("Ingrese una palabra: ")  
print("La palara", palabra, "tiene",cantidad\_vocales(palabra))

# **Funciones: parámetros de tipo lista**

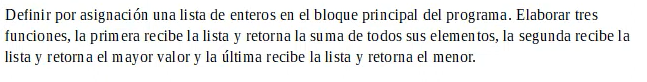
Hemos visto que si trabajamos con programación estructurada debemos dividir nuestro problema en trozos y plantear algoritmos que resuelven cada uno de los trozos.

La división en Python se hace implementando funciones. La comunicación entre estas funciones se produce mediante el envió de datos a través de los parámetros y el valor que retorna la función.

Hasta ahora hemos resuelto problemas enviando datos simples como enteros, float y cadenas de caracteres. En este concepto veremos que una función puede recibir tanto datos simples como estructuras de datos.

La estructura de datos vista hasta este momento y que podemos enviarle a una función en la **lista.**

**Problema 1**

****

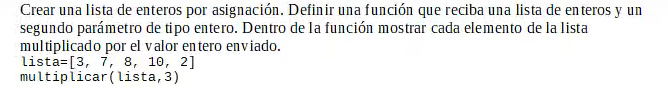
def sumarizar(lista):  
 suma = 0  
 for i in range(len(lista)):  
 suma+=lista[i]  
 return suma  
def mayordef(lista):  
 mayor = 0  
 for i in range(len(lista)):  
 if lista[i]>mayor:  
 mayor = lista[i]  
 return mayor  
def menordef(lista):  
 menor = lista[0]  
 for i in range(len(lista)):  
 if lista[i]<menor:  
 menor = lista[i]  
 return menor  
  
listavalores = [10,20,30,40,50]  
print("Lista Completo")  
print(listavalores)  
print("La suma de todos sus elementos",sumarizar(listavalores))  
print("El mayor de los elementos es: ",mayordef(listavalores))  
print("El menor de los elementos es: ",menordef(listavalores))

**Problema 2:**

****

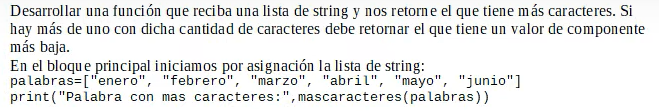
def mayormenor(lista):  
 may = lista[0]  
 men = lista[0]  
 for x in range(1,len(lista)):  
 if lista[x]>may:  
 may = lista[x]  
 else:  
 if lista[x]<men:  
 men = lista[x]  
 print("El valor mayor de la lista: ",may)  
 print("El valor menor de la lista: ",men)  
  
lista = []  
for i in range(5):  
 valor = int(input("Ingrese valor: "))  
 lista.append(valor)  
  
mayormenor(lista)

**Problema 3:**

****

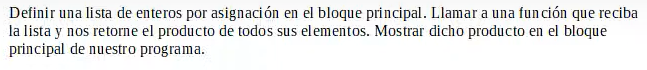
def multiplicacion(lista,valor):  
 for x in range(len(lista)):  
 multi = lista[x] \* valor  
 print(multi)  
  
lista = [3,7,8,2,10]  
print("Lista original")  
print(lista)  
print("Lista de los elementos multiplicados por 3")  
multiplicacion(lista,3)

**Problema 4:**

****

def mascaracteres(lista):  
 pos = 0  
 for i in range(1,len(lista)):  
 if len(lista[i])>len(lista[pos]):  
 pos =i  
 return lista[pos]  
  
palabras = ["enero","febrero","marzo","abril","mayo","junio"]  
print(f"Palabra con mas caracteres: {mascaracteres(palabras)}")

**Problema 5:**

****

def producto(lista):  
 prod = 1  
 for x in range(len(lista)):  
 prod\*=lista[x]  
 return prod  
  
lista = [1,2,3,4,5]  
print(lista)  
print("Multiplicacion de todos sus elementos: ",producto(lista))

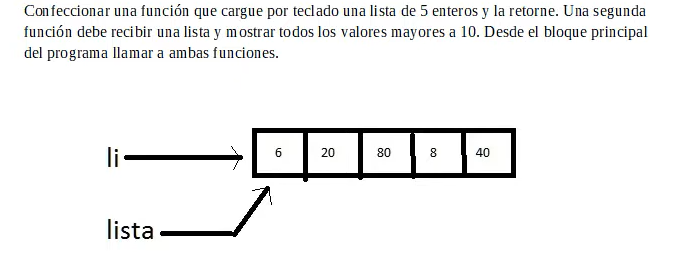
# **Funciones: retorno de una lista**

Hemos avanzado y visto que una función puede recibir como parámetros tipos de datos simples como enteros, flotantes etc, y estructura de datos tipo lista.

También hemos visto que la función mediante la palabra clave return puede retornar un tipo de dato simple desde donde se la invoco.

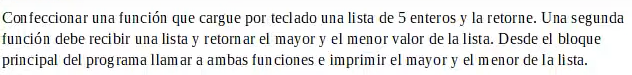
Lo nuevo en este concepto es que una función también puede retornar una estructura de datos tipo lista. Con esto estamos logrando que una función retorne varios datos ya que una lista es una colección de datos.

**Problema 1:**

****

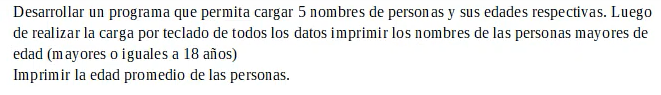
def carga\_lista():  
 li = []  
 for i in range(5):  
 valor = int(input("Ingrese valor: "))  
 li.append(valor)  
 return li  
def imprimir\_mayores10(li):  
 print("Elementos de la lista mayores a 10")  
 for i in range(len(li)):  
 if li[i]>10:  
 print(li[i])  
  
lista = carga\_lista()  
imprimir\_mayores10(lista)

**Problema 2:**

****

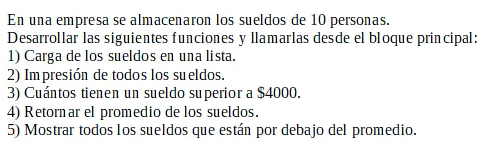
def CargaLista():  
 lista = []  
 for i in range(5):  
 valor = int(input("Ingrese valor: "))  
 lista.append(valor)  
 return lista  
def RetornarMayorMenor(n):  
 mayor = n[0]  
 menor = n[0]  
 for i in range(1,len(n)):  
 if n[i]>mayor:  
 mayor = n[i]  
 elif n[i]<menor:  
 menor = n[i]  
 return [menor,mayor]  
  
lista = CargaLista()  
rango = RetornarMayorMenor(lista)  
print("El menor elementos de la lista es: ",rango[0])  
print("El mayor elemento de la lista es: ",rango[1])

**Problema 3:**

****

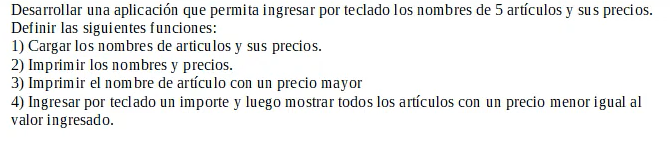
def CargarDatos():  
 nombres = []  
 edades = []  
 final = 5  
 for i in range(final):  
 nombre = input(f"Ingrese el nombre de las personas {i+1}/{final}: ")  
 edad = int(input(f"Ingrese la edad de las personas {i+1}/{final}: "))  
 nombres.append(nombre)  
 edades.append(edad)  
 return nombres,edades  
  
def MayorEdad(nombres,edades):  
 print("Personas mayores de edad")  
 for i in range(len(nombres)):  
 if edades[i]>=18:  
 print(nombres[i])  
def PromedioEdades(edades):  
 suma = 0  
 for i in range(len(edades)):  
 suma+=edades[i]  
 promedio = suma / 5  
 print(f"Edad promedio de las personas {promedio}")  
nombres, edades = CargarDatos()  
MayorEdad(nombres,edades)  
PromedioEdades(edades)

**Problema 4:**

****

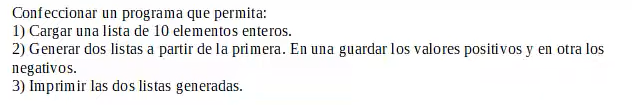
def CargarSueldos():  
 sueldos = []  
 for i in range(10):  
 sueldo = int(input("Ingrese el sueldo: "))  
 sueldos.append(sueldo)  
 return sueldos  
def ImprimirSueldos(n):  
 print("Listado de sueldos")  
 for x, sueldo in enumerate(n):  
 print(x+1,sueldo)  
def SueldoMayor4000(n):  
 cant = 0  
 for i, sueldo in enumerate(n):  
 if sueldo>4000:  
 cant+=1  
 print(f"Cantidad de sueldos mayor a 4000 {cant}")  
  
def Promedio(sueldos):  
 suma = 0  
 for i, sueldo in enumerate(sueldos):  
 suma +=sueldo  
 promedio = suma / len(sueldos)  
 return promedio  
def SueldoBajo(sueldos):  
 pro = Promedio(sueldos)  
 print(f"Sueldo promedio de los empleados: {pro}")  
 print("Sueldo inferiores al promedio")  
 for i, sueldo in enumerate(sueldos):  
 if sueldo<pro:  
 print(sueldo)  
  
sueldos = CargarSueldos()  
ImprimirSueldos(sueldos)  
SueldoMayor4000(sueldos)  
SueldoBajo(sueldos)

**Problema 5:**

****

def CargarDatos():  
 articulos = []  
 precios = []  
 for i in range(3):  
 articulos.append(input("Ingrese el nombre del articulo: "))  
 precios.append(int(input("Ingrese el precio del articulos: ")))  
 return [articulos,precios]  
def Imprimir(articulos,precios):  
 print("Listado completo de articulos y precios")  
 for i,articulo in enumerate(articulos):  
 print(articulo, precios[i])  
def PrecioMayor(articulos,precios):  
 mayor = 0  
 nombre = 0  
 for i, precio in enumerate(precios):  
 if precio>mayor:  
 mayor = precio  
 nombre = articulos[i]  
 print(f"Articulo con un precio mayor es: {nombre} que es: {mayor}")  
def ConsultaPrecio(articulos,precios):  
 valor = int(input("Ingrese un importe para mostrar los articulos con un precio menor: "))  
 for i, precio in enumerate(precios):  
 if precio<=valor:  
 print(articulos[i], precio)  
  
articulos, precios = CargarDatos()  
Imprimir(articulos,precios)  
PrecioMayor(articulos,precios)  
ConsultaPrecio(articulos,precios)

**Problema 6:**

****

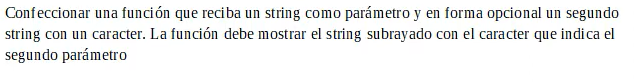
def cargar():  
 lista = []  
 for i in range(5):  
 valor = int(input("Ingrese un valor: "))  
 lista.append(valor)  
 return lista  
def GenerarListas(lista):  
 ListaNegativos = []  
 ListaPositivos = []  
 for i, valor in enumerate(lista):  
 if valor<0:  
 ListaNegativos.append(valor)  
 elif valor>0:  
 ListaPositivos.append(valor)  
 return [ListaNegativos,ListaPositivos]  
def Imprimir(lista):  
 for i, valor in enumerate(lista):  
 print(valor)  
lista = cargar()  
listanega, listapos = GenerarListas(lista)  
print("Valores positivos")  
Imprimir(listapos)  
print("Valores negativos")  
Imprimir(listanega)

# **Funciones: parámetros con valor por defecto**

En Python se pueden definir los parámetros y asignarles un dato en la misma cabecera de la función. Luego cuando llamamos a la función podemos o no enviarle un valor al parámetro.

Los parámetros por defecto nos permiten crear funciones mas flexibles y que se pueden emplear en distintas circunstancias.

**Problema 1:**



def Titulo(titulo, caracter = "\*"):  
 print(titulo)  
 print(caracter\*len(titulo))  
Titulo("Sistema de administracion")  
Titulo("sistema de ventas","----")

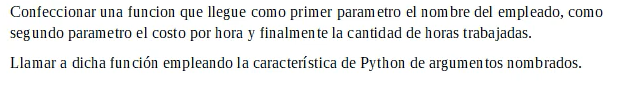
**Problema 2:**

****

def suma(v1,v2,v3 = 0,v4 = 0,v5 = 0):  
 a = v1 + v2 + v3 + v4 + v5  
 return a  
  
print("La suma de 5 + 6: ",suma(5,6))  
print("La suma de 5 + 6 + 3 = ",suma(5,6,3))  
print("La suma de 1 + 1 +1 +1 = ",suma(1,1,1,1))  
print(f"La suma de 1+1+1+1+1 = ",suma(1,1,1,1,1))

# **Funciones: Llamada a la función con argumentos nombrados**

Esta característica de Python nos permite llamar a la función indicando en cualquier orden los parámetros de la misma, pero debemos especificar en la llamada el nombre del parámetro y el valor a enviarle.



**Problema 1:**

def CalcularSueldo(nombre, costohora, cantidadhora):  
 sueldo = costohora \* cantidadhora  
 print(nombre, "trabajo",cantidadhora,"y cobra un sueldo de",sueldo)  
  
CalcularSueldo("Ana",10,100)  
CalcularSueldo(cantidadhora=50,nombre="juan",costohora=45)

**Otra mirada a la función de print por defecto 1:**

print("hola",end="-")  
print("ajasjsaj")

**Problema 2:**

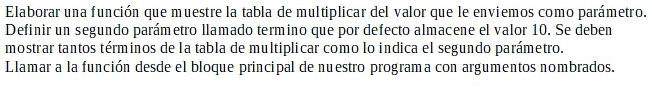
****

def Cargar():  
 lista = []  
 for i in range(4):  
 lista.append(int(input("Ingrese: ")))  
 return lista  
def Imprimir(lista):  
 for i, valor in enumerate(lista):  
 print(valor,end="-")  
lista = Cargar()  
Imprimir(lista)

**Otra mirada a la función de print por defecto 2:**

print("uno","dos","tres",sep="+",end="-")  
print("cuatro","cinco",sep="-")  
print("Fin del programa")

**Problema 3**

****

def tabla(numero, terminos = 10):  
 for i in range(terminos):  
 print(i\*numero,",",sep=" ",end="")  
 print()  
  
tabla(3)  
tabla(3,5)  
tabla(terminos=20,numero=3)

# **Funciones: con cantidad variable de parámetros**

Otra variable en la declaración de una función en Python es la definición de una cantidad variable de parámetros.

Para definir una cantidad variable de parámetros debemos anteceder el carácter asterisco (\*) al último parámetro de la función.

**Problema 1:**

****

def sumar(v1,v2,\*lista):  
 suma = v1 + v2  
 for i, valor in enumerate(lista):  
 suma+=valor  
 return suma  
print(sumar(1,2))  
print(sumar(1,2,3,4,5))

# **Desempaquetar una lista**

Puede ser que tengamos una función que recibe una cantidad fija de parámetros y necesitemos llamarla enviando valores que se encuentran en una lista. La forma mas sencilla es anteceder el carácter \* al nombre de la variable.

**Problema 2:**

****

def sumar(v1,v2,v3):  
 return v1 + v2 + v3  
lista = [1,2,3]  
print(sumar(lista[0],lista[1],lista[2]))  
print(sumar(\*lista))

**Problema 3:**

****

def CantidadMayores18(edad,\*edades):  
 cant = 0  
 if edad>=18:  
 cant+=1  
 for i, valor in enumerate(edades):  
 cant+=1  
 return cant  
print("Cantidad de personas mayores de 18 años son")  
print(CantidadMayores18(23,2,3,5,123,23,123,2))