

LICENCIATURA EN **CIENCIA DE DATOS**

EJERCICIOS PROGRAMACIÓN IMPERATIVA PYTHON

OBJETIVOS

- Brindar ejercicios de apoyo y repaso utilizando este paradigma de programación
- Desarrollar habilidades de interpretación de enunciados bajo este paradigma.
- Generar el espacio de interacción y de corrección.
- Posibilitar el entendimiento de nuevos paradigmas de desarrollo

1. Aspectos conceptuales

- ¿Qué ventajas tiene la utilización de funciones?
- ¿Hay algún cuidado en el orden en el que se pasan los parámetros a una función?
- ¿Cuándo uso la sentencia return?
- ¿Qué diferencia hay entre la definición y la invocación de una función?
- ¿Qué son los parámetros formales y para qué sirven? Ejemplifique.
- ¿Qué son los parámetros reales y para qué sirven? Ejemplifique.
- ¿Qué significa el cuerpo de una función? Ejemplifique.
- ¿Existen funciones sin parámetros o argumentos?
- ¿Puede usar una letra o un número como parámetro formal? ¿Y como parámetro real?
- ¿Puedo tener una cantidad distinta de parámetros formales que reales en una función?
- ¿Cómo se puede implementar un módulo con solo definiciones de funciones e importarlo desde tu programa? ¿Cuáles son las formas de importar que ofrece Python?
- ¿Qué diferencias hay entre los siguientes códigos?
 - import math
 - from math import sqrt

2. Ahora a practicar

Ejercicio 1:

- a) Dado el siguiente código indique cuáles son los parámetros reales y los formales:

#Definición de funciones

```
def sumaAlcuadrado(x, y):
    rta= x**2 + 2*x*y + y**2
    return rta
```

#Programa principal

```
print ("Bienvenidos/as a la Suma al Cuadrado")
a=input("Ingrese el valor de a:")
b=input("Ingrese el valor de b:")
print (sumaAlcuadrado(a, b))
```

- b) Mencione los errores en los siguientes códigos. Justifique:

```
a) def suma(par1, par2):
    print(par1+par2)
    suma()
```

```
b) def suma(par1, par2):
    print (par1 + par2)
    print(suma(12, 10))
```

```
c) def suma(par1, par2):
    return (par1 + par2)
    suma(12, 10)
```

```
d) def suma(par1):
    return (par1 + 2)
suma(12, 10)
```

Ejercicio 2: Definir una función denominada **imprimir_mensaje** que imprima el siguiente mensaje en pantalla: “Estudiando Fundamentos de Informática en la UNAJ”. No recibe ninguna información por lo tanto no tiene ningún parámetro formal.

Ejercicio 3: Definir una función denominada **retorno_mensaje** que retorne siguiente mensaje: “Estudiando Fundamentos de Informática en la UNAJ”.

- A. ¿Cómo hago para mostrar ese mensaje en pantalla?
- B. ¿Qué diferencia encuentra con el ejercicio anterior?
- C. Si tuvieras que imprimir mensajes como “Estudiando Matemática I en la UNAJ” y “Estudiando Python en la UNAJ” utilizando la misma función ¿Cómo la modificarías?

Ejercicio 4: Definir una función denominada **imprimo_fecha** que reciba tres cadenas de caracteres como parámetros formales, que representan un día, un mes y un año e imprima la fecha de la siguiente manera: “ 21 de septiembre de 2021”.

Ejercicio 5: Definir una función denominada **cuantos_dias** que reciba el número de mes como parámetro y retorne la cantidad de días que posee. Ejemplo: **cuantos_dias**(1), debería retornar 31.

Ayuda: Pensar en tener una lista de la siguiente manera: [“enero”,31], [“febrero”, 28], ...]

Ejercicio 6: Definir una función que reciba un número como parámetro y mostrar la tabla de multiplicar de dicho número.

Ejercicio 7: Definir una función que calcule el área de un círculo, otra que calcule el área de un rectángulo y otra que calcule el área de un cuadrado. Analice qué parámetros deberían recibir dichas funciones.

Ejercicio 8: Definir una función llamada **calculo_rebaja** que reciba dos números, uno con el precio anterior y otro para el precio rebajado y devuelva un número que represente el porcentaje rebajado.

Ejercicio 9: Definir una función llamada **calculo_nuevoPrecio** que reciba dos números, uno con el precio anterior y otro con el número de porcentaje a aumentar y devuelva el precio aumentado.

Ejercicio 10: Definir una función llamada **calculo_transporte** que reciba cuatro números: la cantidad de alumnos de 1era, 2da y 3er. salita de un jardín de infantes y la cantidad de asientos del transporte escolar. La función debe retornar cuántos micros necesito contratar para una excursión sabiendo que cada salita es acompañada por tres adultos.

Ejercicio 11: Definir una función llamada **armo_cartel** que reciba una cadena de caracteres (para el nombre del producto) y dos números (el precio anterior y el otro para el precio rebajado) e imprima un cartel de la siguiente forma:

Atención!!! Gran rebaja para el producto nombre (recibido como parámetro)

Antes: precio anterior (dato recibido como parámetro)

Ahora: precio rebajado (dato recibido como parámetro)

Ejercicio 12: Definir una función llamada **calculo_litros** que reciba tres números, el alto, ancho y profundidad (en metros) de una pileta y devuelva la cantidad de litros que tiene.

Ejercicio 13: Definir una función llamada **a_pagar** que reciba 4 números: la cantidad de personas, el monto gastado en bebida, el monto gastado en comida y el del alquiler del lugar, y retorne cuánto le toca pagar a cada uno.

Ejercicio 14: Definir tres funciones llamadas **convertir_a_dolar**, **convertir_a_euro** y **convertir_a_real**. Cada función recibe un parámetro que representa un monto en pesos y devuelve su conversión respectiva.

Ejercicio 15: Definir una función llamada **calculo_dosis** que reciba tres números. Uno para la cantidad de días que debe suministrar el remedio, el segundo dato para la cantidad de veces por día que debe tomarlo, y el último dato para la cantidad de comprimidos que trae el envase. La función debe devolver verdadero si el envase alcanza para ese tratamiento y falso si no alcanza.

Ejercicio 16: Definir una función llamada **precio_con_iva** que agrega el IVA (21%) de un producto dado su precio de venta sin IVA.

2. Ejercicios complementarios

Ejercicio 17:

- a) Definir una función que reciba como parámetro una lista de números y retorne la suma del primer elemento con el último.

```
#Zona de definiciones de funciones
def sumaPrimUlt(lis):
    #retorna la suma entre el primer elemento de la lista con el último
    ....

def promedioPrimUlt(lis):
    #retorna el promedio entre el primer elemento de la lista con el último
    ....

#Zona del programa principal
#solicitar al usuario 3 números, armar la lista e invocar las funciones anteriores mostrando los #resultados
.....
```

Ejercicio 18: En este código una fracción está representada por una lista de dos elementos, el numerador y el denominador. Por ejemplo la fracción $\frac{3}{4}$ sería la lista (3,4). Complete el código según corresponda.

```
#Zona de definiciones de funciones
def cargarFraccion():
    #Solicita al usuario el numerador y denominador. Arma la fracción como una lista y la retorna
    ...

def numeradorFraccion(x):
    #Retorna el numerador que se encuentra en la fracción x, representada como una lista
    ....

def denominadorFraccion(x):
    #Retorna el denominador que se encuentra en la fracción x, representada como una lista
    ....

def sumaFracciones(x, y):
    #Retorna la suma de las fracciones, representadas como listas
    ....

def restaFracciones(x, y):
    #Retorna la resta de las fracciones, representadas como listas
    ....

def divisionFracciones(x, y):
    #Retorna la división de las fracciones, representadas como listas:
```

```
....
def multiplicacionFracciones(x, y):
    #Retorna la multiplicación fracciones, representadas como listas
....

    #Zona del programa principal
    print ("Bienvenidos/as a cuentas con Fracciones")
    a=cargarFraccion()
    b=cargarFraccion()
    print ("El denominador de la primera fracción es:", ..... )
    print ("El numerador de la segunda fracción es:", ..... )
    print ("La suma de dichas fracciones es:", ..... )
    print ("La resta de dichas fracciones es:", ..... )
    print ("La multiplicación de dichas fracciones es:", ..... )
    print ("La división es:", ..... )
```

3. Ejercicios extras utilizando programación orientada a objetos

Ejercicio 19: Implemente el **Ejercicio 14** utilizando clases. Realice las correspondientes invocaciones a cada uno de los métodos.

Ejercicio 20: Implemente el **Ejercicio 18** utilizando clases. Realice las correspondientes invocaciones a cada uno de los métodos.