FUNCIONES en PYTHON

link de GITHUB: https://github.com/villacis-cristian/ejercicios_en_clase_python.git

CRISTIAN ANDRES VILLACIS MENDOZA

DESARROLLO DE SOFTWARE

EJERCICIOS en Python

1. Escribe un programa que calcule la suma de todos los elementos de una *lista* dada. La lista sólo puede contener elementos numéricos.

2. Dada una lista con elementos duplicados, escribir un programa que muestre una nueva lista con el mismo contenido que la primera pero sin elementos duplicados. Para este ejercicio, no puedes hacer uso de objetos de tipo 'Set'.

3. Escribe un programa que construya un diccionario que contenga un número (entre 1 y *n*) de elementos de esta forma: (x, x*x). Ejemplo: para n = 5, el diccionario resultante sería {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

4. Escribe un programa que, dada una lista de palabras, compruebe si alguna empieza por 'a' y tiene más de 9 caracteres. Si dicha palabra existe, el programa deberá terminar en el momento exacto de encontrarla. El programa también debe mostrar un mensaje apropiado por pantalla que indique el éxito o el fracaso de la búsqueda. En caso de éxito, también se mostrará por pantalla la palabra encontrada.

5. Dada una lista *L* de números positivos, escribir un programa que muestre otra lista (ordenada) que contenga todo índice *i* que cumpla la siguiente condición: *L[i]* es múltiplo de 3. Por ejemplo, dada la lista *L* = [3,5,13,12,1,9] el programa mostrará la lista [0,3,5] dado que *L[0]*, *L[3]* y *L[5]* son, respectivamente, 3, 12 y 9, que son los únicos múltiplos de 3 que hay en *L*.

6. Dado un diccionario cuyos elementos son pares de tipo string y numérico (es decir, las claves son de tipo 'str' y los valores son de tipo 'int' o 'float'), escribe un programa que muestre por pantalla la clave cuyo valor asociado represente el valor numérico más alto de todo el diccionario. Por ejemplo, para el diccionario {'a': 4.3, 'b': 1, 'c': 7.8, 'd': -5} la respuesta sería 'c', dado que 7.8 es el valor más alto de los números 4.3, 1, 7.8 y -5.

7. Dada la lista *a* = [2, 4, 6, 8] y la lista *b* = [7, 11, 15, 22], escribe un programa que itere las listas *a* y *b* y multiplique cada elemento de *a* que sea mayor que 5 por cada elemento de *b* que sea menor que 14. El programa debe mostrar los resultados por pantalla.

8. Escribir un programa que pida un valor numérico X al usuario. Para ello podéis hacer uso de la función predefinida 'input'. El programa deberá mostrar por pantalla el resultado de la división 10/X. En caso de que el usuario introduzca valores no apropiados, el programa deberá gestionar correctamente las excepciones, por ejemplo, mostrando mensajes informativos por pantalla.

9. Escribir un programa que cree un *diccionario* cualquiera. Posteriormente, el programa pedirá al usuario (a través de la función predefinida 'input') que introduzca una clave del diccionario. Si la clave introducida es correcta (es decir, existe en el diccionario), el programa mostrará por pantalla el valor asociado a dicha clave. En caso de que la clave no exista, el programa gestionará de manera apropiada el error, por ejemplo, mostrando un mensaje informativo al usuario.

10. Escribe una *list comprehension* que construya una lista con los números *enteros* positivos de una lista de números dada. La lista original puede incluir números de tipo *float*, los cuales deben ser descartados.

11. Escribe una *set comprehension* que, dada una palabra, construya un conjunto que contenga las vocales de dicha palabra.

12. Escribe una *list comprehension* que construya una lista con todos los números del 0 al 50 que contengan el dígito 3. El resultado será: [3, 13, 23, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 43].

13. Escribe una *dictionary* comprehension que construya un diccionario que incluya los tamaños de cada palabra en una frase dada. Ejemplo: el resultado para la frase "Soy un ser humano" será {'Soy': 3, 'un': 2, 'ser': 3, 'humano': 6}

14. Escribe una *list comprehension* que construya una lista que incluya todos los números del 1 al 10 en orden. La primera mitad se mostrarán en formato numérico; la segunda mitad en texto. Es decir, el resultado será: [1, 2, 3, 4, 5, 'seis', 'siete', 'ocho', 'nueve', 'diez'].

EJERCICIO 01

In [2]:
```
# Definimos una lista de números
numeros = [1, 2, 3, 4, 5]

# Inicializamos una variable para almacenar la suma
suma = 0

# Iteramos sobre cada elemento de la lista
for numero in numeros:
    # Sumamos cada número al acumulador
    suma += numero

# Imprimimos el resultado
print("La suma de los elementos es:", suma)
```

```
La suma de los elementos es: 15
```

EJERCICIO 02

In [2]:
```
# Lista original con elementos duplicados
lista_original = [1, 2, 2, 3, 4, 4, 5]

# Creamos una lista vacía para almacenar elementos únicos
lista_sin_duplicados = []

# Iteramos sobre cada elemento de la lista original
for elemento in lista_original:
    # Verificamos si el elemento no está en la nueva lista
    if elemento not in lista_sin_duplicados:
        # Si no está, lo agregamos
        lista_sin_duplicados.append(elemento)

# Imprimimos la lista sin duplicados
print("Lista sin duplicados:", lista_sin_duplicados)
```

```
Lista sin duplicados: [1, 2, 3, 4, 5]
```

EJERCICIO 03

In [3]:
```
# Definimos el valor máximo para el diccionario
n = 5

# Creamos un diccionario vacío
diccionario = {}

# Iteramos desde 1 hasta n (incluido)
for x in range(1, n + 1):
    # Asignamos la clave x y el valor x**x
    diccionario[x] = x ** x

# Imprimimos el diccionario resultante
print("Diccionario resultante:", diccionario)
```

```
Diccionario resultante: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

EJERCICIO 04

In [8]:
```
# Lista de palabras para buscar
palabras = ["hola", "adios", "amaranjado", "amarillo", "computadora"]

# Bandera para indicar si se encontró la palabra
encontrado = False

# Iteramos sobre cada palabra en la lista
for palabra in palabras:
    # Verificamos si la palabra empieza con 'a' y tiene más de 9 caracteres
    if palabra.startswith('a') and len(palabra) > 9:
        # Si se cumple, imprimimos la palabra y marcamos como encontrado
        print("Palabra encontrada:", palabra)
        encontrado = True
        # Salimos del bucle
        break

# Si no se encontró ninguna palabra, mostramos un mensaje
if not encontrado:
    print("No se encontró ninguna palabra que cumpla los requisitos.")
```

```
Ejercicio 4: aloh
```

EJERCICIO 05

In [3]:
```
# Lista original de números
L = [3, 5, 13, 12, 1, 9]

# Creamos una lista vacía para almacenar los índices
indices = []

# Iteramos sobre los índices y valores de la lista
for i in range(len(L)):
    # Verificamos si el valor en el índice i es múltiplo de 3
    if L[i] % 3 == 0:
        # Si es así, agregamos el índice a la lista
        indices.append(i)

# Ordenamos la lista de índices
indices.sort()

# Imprimimos la lista de índices
print("Índices de elementos múltiplos de 3:", indices)
```

```
Índices de elementos múltiplos de 3: [0, 3, 5]
```

EJERCICIO 06

In [4]:
```
# Diccionario de ejemplo
diccionario = {'a': 4.3, 'b': 1, 'c': 7.8, 'd': -5}

# Inicializamos variables para almacenar la clave y valor máximo
clave_max = None
valor_max = float('-inf')  # Inicializamos con el menor valor posible

# Iteramos sobre cada par clave-valor en el diccionario
for clave, valor in diccionario.items():
    # Comparamos el valor actual con el valor máximo encontrado
    if valor > valor_max:
        # Si es mayor, actualizamos la clave y el valor máximo
        clave_max = clave
        valor_max = valor

# Imprimimos la clave con el valor máximo
print("Clave con el valor más alto:", clave_max)
```

```
Clave con el valor más alto: c
```

EJERCICIO 07

In [5]:
```
# Definimos las listas a y b
a = [2, 4, 6, 8]
b = [7, 11, 15, 22]

# Iteramos sobre cada elemento de la lista a
for num_a in a:
    # Verificamos si el elemento es mayor que 5
    if num_a > 5:
        # Iteramos sobre cada elemento de la lista b
        for num_b in b:
            # Verificamos si el elemento es menor que 14
            if num_b < 14:
                # Multiplicamos y mostramos el resultado
                print(f"{num_a} * {num_b} = {num_a * num_b}")
```

```
6 * 7 = 42
6 * 11 = 66
8 * 7 = 56
8 * 11 = 88
```

EJERCICIO 08

In [6]:
```
# Solicitamos al usuario que ingrese un valor numérico
entrada = input("Por favor, ingrese un número para dividir 10: ")

try:
    # Intentamos convertir la entrada a float
    X = float(entrada)
    # Realizamos la división
    resultado = 10 / X
    # Mostramos el resultado
    print("El resultado de 10 /", X, "es:", resultado)
except ValueError:
    # Manejo de error si la entrada no es un número
    print("Error: Debe ingresar un valor numérico.")
except ZeroDivisionError:
    # Manejo de error si se intenta dividir por cero
    print("Error: No se puede dividir por cero.")
```

```
El resultado de 10 / 10.0 es: 1.0
```

EJERCICIO 09

In [8]:
```
# Creamos un diccionario de ejemplo
diccionario = {'nombre': 'Juan', 'edad': 30, 'ciudad': 'Madrid'}

# Solicitamos al usuario que ingrese una clave
clave = input("Ingrese una clave para buscar en el diccionario: ")

try:
    # Intentamos acceder al valor asociado a la clave
    valor = diccionario[clave]
    # Si existe, mostramos el valor
    print(f"El valor asociado a '{clave}' es: {valor}")
except KeyError:
    # Manejo de error si la clave no existe
    print(f"Error: La clave '{clave}' no existe en el diccionario.")
```

```
Error: La clave 'Juan' no existe en el diccionario.
```

EJERCICIO 10

In [9]:
```
# Lista original con números enteros y flotantes
lista_original = [1, 2.5, 3, 4.7, 5, 6.2]

# List comprehension para filtrar enteros positivos
# Solo incluimos x si es instancia de int y es positivo
lista_enteros = [x for x in lista_original if isinstance(x, int) and x > 0]

# Imprimimos la lista resultante
print("Lista de enteros positivos:", lista_enteros)
```

```
Lista de enteros positivos: [1, 3, 5]
```

EJERCICIO 11

In [10]:
```
# Palabra de ejemplo
palabra = "murciélago"

# Set comprehension para obtener vocales únicas
# Convertimos a minúscula y filtramos caracteres que están en 'aeiou'
vocales = {letra.lower() for letra in palabra if letra.lower() in 'aeiou'}

# Imprimimos el conjunto resultante
print("Vocales en la palabra:", vocales)
```

```
Vocales en la palabra: {'i', 'o', 'a', 'u'}
```

EJERCICIO 12

In [12]:
```
# List comprehension para números del 0 al 50 que contienen el dígito 3
# Convertimos cada número a string y verificamos si '3' está en él
numeros_con_3 = [num for num in range(51) if '3' in str(num)]

# Imprimimos la lista resultante
print("Números que contienen el dígito 3:", numeros_con_3)
```

```
Números que contienen el dígito 3: [3, 13, 23, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 43]
```

EJERCICIO 13

In [11]:
```
# Frase de ejemplo
frase = "Soy un ser humano"

# Dictionary comprehension para crear diccionario con longitudes
# Dividimos la frase en palabras y creamos pares {palabra, longitud}
longitudes = {palabra: len(palabra) for palabra in frase.split()}

# Imprimimos el diccionario resultante
print("Longitudes de palabras:", longitudes)
```

```
Longitudes de palabras: {'Soy': 3, 'un': 2, 'ser': 3, 'humano': 6}
```

EJERCICIO 14

In [10]:
```
# Lista de números en texto para la segunda mitad
numeros_texto = ['seis', 'siete', 'ocho', 'nueve', 'diez']

# List comprehension que combina números y texto
# Para índices 0-4 (1-5) usamos el número, para 5-9 usamos el texto
lista_combinada = [i+1 if i < 5 else numeros_texto[i-5] for i in range(10)]

# Imprimimos la lista resultante
print("Lista combinada:", lista_combinada)
```

```
Lista combinada: [1, 2, 3, 4, 5, 'seis', 'siete', 'ocho', 'nueve', 'diez']
```

In [ ]:
```

```