**Ejercicios con arreglos**

**#1**

diasSemana = [''] \* 7

for i in range(7):

diasSemana[i] = input(f"Ingrese el nombre del día {i+1}: ")

print("Los días de la semana son :", diasSemana)

**#2**

import numpy as np

dias\_laborables = np.array(['Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes'])

for i in range(len(dias\_laborables)):

dias\_laborables[i] = dias\_laborables[i].upper()

print("Los días laborables en mayúsculas:", dias\_laborables)

**#3**

numeros\_enteros = []

for i in range(10):

numero = int(input(f"Ingrese el número {i + 1}: "))

numeros\_enteros.append(numero)

numeros\_pares = []

numeros\_impares = []

for numero in numeros\_enteros:

if numero % 2 == 0:

numeros\_pares.append(numero)

else:

numeros\_impares.append(numero)

print("Array original:", numeros\_enteros)

print("Números pares:", numeros\_pares)

print("Números impares:", numeros\_impares)

**#4**

numeros\_enteros = []

for i in range(10):

numero = int(input(f"Ingrese el número {i + 1}: "))

numeros\_enteros.append(numero)

suma\_pares = 0

suma\_impares = 0

for numero in numeros\_enteros:

if numero % 2 == 0:

suma\_pares += numero

else:

suma\_impares += numero

print("Array original:", numeros\_enteros)

print("Suma de números pares:", suma\_pares)

print("Suma de números impares:", suma\_impares)

**#5**

def rotar(lst1, lst2):

if len(lst1) > 0 and len(lst2) > 0:

nombre\_rotado1 = lst1.pop(0)

lst2.append(nombre\_rotado1)

nombre\_rotado2 = lst2.pop(0)

lst1.append(nombre\_rotado2)

else:

print("Ambas listas deben tener al menos un nombre.")

ocupados = input("Ingrese nombres para el equipo 'ocupados' separados por coma: ").split(', ')

libres = input("Ingrese nombres para el equipo 'libres' separados por coma: ").split(', ')

print("Array Ocupados antes:", ocupados)

print("Array Libres antes:", libres)

rotar(ocupados, libres)

print("Array Ocupados después:", ocupados)

print("Array Libres después:", libres)

**#6**

asistentes\_html = ["Jim", "Ani", "Lucas", "Miky", "Judas"]

asistentes\_css = ["Ani", "Salomon", "Miky", "Miguel", "Jhina"]

asistentes\_comunes\_set = set(asistentes\_html) & set(asistentes\_css)

asistentes\_comunes = sorted(list(asistentes\_comunes\_set))

print("Asistentes a ambos cursos:", asistentes\_comunes)

**#7**

def gestionar\_lista\_compra(lista\_compra):

pendientes = []

for articulo in lista\_compra:

if not articulo["estado"]:

pendientes.append(articulo["nombre"])

return pendientes

entrada\_usuario = input("Ingrese productos y su estado (nombre1,True/False;nombre2,True/False;...): ")

articulos = [articulo.split(',') for articulo in entrada\_usuario.split(';')]

lista\_compra = [{"nombre": nombre, "estado": estado.lower() == "true"} for nombre, estado in articulos]

pendientes = gestionar\_lista\_compra(lista\_compra)

print("Lista de pendientes:", pendientes)

**#8**

def precio(articulo, articulos, precios):

if articulo in articulos:

indice = articulos.index(articulo)

return precios[indice]

else:

return -1

articulos = input("Ingrese nombres de artículos separados por comas: ").split(',')

precios = [float(precio) for precio in input("Ingrese precios correspondientes separados por comas: ").split(',')]

nombre\_articulo = input("Ingrese el nombre del artículo para obtener el precio: ")

resultado = precio(nombre\_articulo, articulos, precios)

print(f"El precio de {nombre\_articulo} es: {resultado}")

**#9**

def obtener\_extremos\_edad(lista\_familia):

if not lista\_familia:

return None, None

miembro\_mayor = max(lista\_familia, key=lambda x: x.get('EDAD'))

miembro\_menor = min(lista\_familia, key=lambda x: x.get('EDAD'))

return miembro\_mayor, miembro\_menor

familia = [

{"NOMBRE": "Pietro", "EDAD": 25},

{"NOMBRE": "Lucas", "EDAD": 45},

{"NOMBRE": "Juanita", "EDAD": 85},

]

mayor, menor = obtener\_extremos\_edad(familia)

print("Miembro de mayor edad:", mayor)

print("Miembro de menor edad:", menor)