# Python Básico

# Intoducción

Este tutorial práctico está diseñado para desarrolladores que trabajaran con Kubernetes y desean reforzar o aprender los fundamentos de Python. Los ejemplos están enfocados en un dominio común para aplicaciones distribuidas en contenedores.

# 1. Secuencias en Python

Las secuencias en Python permiten agrupar datos. Las más comunes son listas, tuplas y diccionarios.

## 1.1 Listas

Las listas son colecciones ordenadas y modificables. Se utilizan para representar, por ejemplo, transacciones recientes.

Ejemplo:

transacciones = ["depósito", "retiro", "transferencia"]  
print(transacciones[1]) # Salida: retiro

## 1.2 Tuplas

Las tuplas son colecciones ordenadas pero inmutables. Útiles para representar configuraciones fijas como coordenadas.

Ejemplo:

ubicacion\_cajero = (12.4567, -76.8932)  
print(ubicacion\_cajero)

## 1.3 Diccionarios

Los diccionarios permiten asociar claves con valores. Son muy usados para representar cuentas bancarias.

Ejemplo:

cuenta = {"titular": "Juan", "saldo": 1500.75}  
print(cuenta["saldo"])

# 2. Ciclos en Python

Los ciclos permiten repetir instrucciones. Python incluye `for` y `while` como estructuras principales.

## 2.1 Bucle for

Ejemplo: mostrar operaciones bancarias.

for operacion in transacciones:  
 print(f"Procesando: {operacion}")

## 2.2 Bucle while

Ejemplo: simular intento de autenticación.

intentos = 0  
while intentos < 3:  
 print("Intentando autenticación...")  
 intentos += 1

# 3. Introducción a la Programación Orientada a Objetos (POO)

La POO permite representar entidades del mundo real como objetos en código. En banca, es útil para modelar cuentas, usuarios, etc.

## 3.1 Clases y Objetos

Ejemplo: clase CuentaBancaria.

class CuentaBancaria:  
 def \_\_init\_\_(self, titular, saldo):  
 self.titular = titular  
 self.saldo = saldo  
  
 def depositar(self, monto):  
 self.saldo += monto  
 print(f"Depósito exitoso. Saldo actual: {self.saldo}")  
  
cuenta = CuentaBancaria("Ana", 1000)  
cuenta.depositar(500)

# Ejercicios Propuestos

1. Crea una lista con cinco tipos de operaciones bancarias y recórrala con un ciclo `for`.

2. Crea un diccionario para representar una cuenta con número, tipo y saldo. Modifica el saldo.

3. Implementa una clase `Cliente` que almacene nombre, número de cuenta y permita actualizar el número de cuenta.