



FACULTAD DE INGENIERÍA Y ARQUITECTURA

CARRERA

TECNICO EN DESARROLLO DE SOFTWARE

TRABAJO

PRACTICA 6 MED

ING.

JOSUE ISAI HERRERA BENITEZ

ESTUDIANTE

ALVIN EZEQUIEL ROSALES HERNÁNDEZ-U20230560

CICLO

02-2023

CIUDAD UNIVERSITARIA "UNIVO" 28 DE SEPTIEMBRE DE 2023

# Tema: Programación Orientada a Objetos en Python.

## OBJETIVO.

Utilizar nociones de programación orientada a objetos para la resolución de problemas con Python.

```
alvinrhd Practica Estructura 6 5d3ab22 · 3 minutes ago Historia

Código Culpa 19 líneas (17 loc) · 748 Bytes Codifique un 55% más rápido con GitHub Copilot Crudo

1 class Artículo:
2     def __init__(self, nombre, cantidad, precio):
3         # Inicializar los atributos del artículo
4         self.nombre = nombre
5         self.cantidad = cantidad
6         self.precio = precio
7
8     def vender(self, cantidad_vendida):
9         if cantidad_vendida <= self.cantidad:
10            # Verificar si hay suficiente cantidad en stoc para la venta
11            self.cantidad -= cantidad_vendida
12            total_venta = cantidad_vendida * self.precio
13            return f"Venta realizada. Total a pagar: ${total_venta}"
14        else:
15            return "No hay suficiente stoc para realizar la venta."
16
17 # Ejemplo de uso
18 producto1 = Artículo("Camiseta", 50, 20.99)
19 print(producto1.vender(10)) # Realizar una venta de 10 unidades
```

```
alvinrhd Practica Estructura 6 5d3ab22 · 4 minutes ago Historia

Código Culpa 24 líneas (21 loc) · 920 Bytes Codifique un 55% más rápido con GitHub Copilot Crudo

1 class Usuario:
2     def __init__(self, nombre, correo, saldo_inicial):
3         # Inicializar los atributos del usuario
4         self.nombre = nombre
5         self.correo = correo
6         self.saldo = saldo_inicial
7
8     def depositar(self, monto):
9         # Realizar un depósito en la cuenta del usuario
10        self.saldo += monto
11        return f"Se ha depositado ${monto}. Saldo actual: ${self.saldo}"
12
13    def retirar(self, monto):
14        if monto <= self.saldo:
15            # Verificar si hay suficiente saldo para el retiro
16            self.saldo -= monto
17            return f"Se ha retirado ${monto}. Saldo actual: ${self.saldo}"
18        else:
19            return "Saldo insuficiente para realizar el retiro."
20
21 # Ejemplo de uso
22 usuario1 = Usuario("Alvin Rosales", "alvin@ejemplo.com", 1000)
23 print(usuario1.depositar(500)) # Realizar un depósito de $500
24 print(usuario1.retirar(300)) # Realizar un retiro de $300
```

```
alvinrhd Practica Estructura 6 5d3ab22 · 4 minutes ago Historia

Código Culpa 34 líneas (28 loc) · 1.17 KB Codifique un 55% más rápido con GitHub Copilot

1 class Estudiante:
2     def __init__(self, nombre, apellido, carnet, carrera):
3         # Inicializar los atributos del estudiante
4         self.nombre = nombre
5         self.apellido = apellido
6         self.carnet = carnet
7         self.carrera = carrera
8
9     def actualizar_nombre(self, nuevo_nombre):
10        # Actualizar el nombre del estudiante
11        self.nombre = nuevo_nombre
12
13    def actualizar_apellido(self, nuevo_apellido):
14        # Actualizar el apellido del estudiante
15        self.apellido = nuevo_apellido
16
17    def actualizar_carnet(self, nuevo_carnet):
18        # Actualizar el carnet del estudiante
19        self.carnet = nuevo_carnet
20
21    def actualizar_carrera(self, nueva_carrera):
22        # Actualizar la carrera del estudiante
23        self.carrera = nueva_carrera
24
25    def __str__(self):
26        # Devolver una representación en cadena del estudiante
27        return f"Nombre: {self.nombre} {self.apellido}\nCarnet: {self.carnet}\nCarrera: {self.carrera}"
28
```

```
Practica_Estructura6 / ejercicio3.py ↑ Arriba

Código Culpa 34 líneas (28 loc) · 1.17 KB Codifique un 55% más rápido con GitHub Copilot

1 class Estudiante:
2     def __init__(self, nombre, apellido, carnet, carrera):
3         # Actualizar el nombre del estudiante
4         self.nombre = nuevo_nombre
5
6     def actualizar_apellido(self, nuevo_apellido):
7         # Actualizar el apellido del estudiante
8         self.apellido = nuevo_apellido
9
10    def actualizar_carnet(self, nuevo_carnet):
11        # Actualizar el carnet del estudiante
12        self.carnet = nuevo_carnet
13
14    def actualizar_carrera(self, nueva_carrera):
15        # Actualizar la carrera del estudiante
16        self.carrera = nueva_carrera
17
18    def __str__(self):
19        # Devolver una representación en cadena del estudiante
20        return f"Nombre: {self.nombre} {self.apellido}\nCarnet: {self.carnet}\nCarrera: {self.carrera}"
21
22 # Ejemplo de uso
23 estudiante1 = Estudiante("Alvin", "Rosales", "U20230560", "Tec. En Desarrollo de Soft.")
24 print(estudiante1)
25 estudiante1.actualizar_nombre("Alvin")
26 estudiante1.actualizar_carnet("u20230560")
27 print(estudiante1)
28
```

ENLACE DEL REPOSITORIO DE GITHUB:

[https://github.com/AlvinRHD/Practica\\_Estructura6.git](https://github.com/AlvinRHD/Practica_Estructura6.git)

