

Constructores, Destrucciones, Clases, Objetos y Métodos

Alumno: Puma Huanca Anthony Rusbel
Docente: Ing. Coyla Idme Leonel
Lenguajes de programación II – FINESI
Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática

Tema: Constructores, Destrucciones, Clases, Objetos y Métodos

1. Clase Libro: Mostrar información y destruir objeto

Problema:

Crea una clase `Libro` que contenga:

Atributos: título, autor, año.

Constructor que reciba los tres atributos y muestre el mensaje: “Libro [título] de [autor] creado.”

Método `mostrar_info()` que muestre los datos del libro.

Destructor que imprima: “Libro [título] eliminado de la biblioteca.”

Luego, crea un objeto de esta clase, llama al método y destrúyelo.

Código en Python:

```
1 class Libro:
2     def __init__(self, titulo, autor):
3         self.titulo = titulo
4         self.autor = autor
5         print("Objeto Libro creado")
6
7     def mostrar_informacion(self):
8         print(f"Titulo: {self.titulo}")
9         print(f"Autor: {self.autor}")
10
11 titulo = input("Ingrese el titulo del libro: ")
12 autor = input("Ingrese el autor del libro: ")
13
14 libro = Libro(titulo, autor)
```

```

15 libro.mostrar_informacion()
16
17 del libro
18 try:
19     print(libro)
20 except NameError:
21     print("Objeto libro destruido")

```

Listing 1: Código Practica 1

2. Clase CuentaBancaria: Depósito y retiro

Problema:

Crea una clase CuentaBancaria con:

Atributos: titular, saldo.

Constructor que inicialice los datos y muestre un mensaje de bienvenida.

Método depositar(monto) que aumente el saldo.

Método retirar(monto) que reste del saldo si hay fondos suficientes.

Destructor que muestre: "Cuenta de [titular] cerrada."

Crea un objeto, realiza un depósito y un retiro, y luego destrúyelo.

Código en Python:

```

1 class CuentaBancaria:
2     def __init__(self, saldo_inicial):
3         self.saldo = saldo_inicial
4         print("Cuenta creada con éxito . Bienvenido al banco.")
5
6     def depositar(self, monto):
7         self.saldo += monto
8         print(f"Depósito de {monto} realizado. Saldo actual: {self.saldo}")
9
10    def retirar(self, monto):
11        if monto <= self.saldo:
12            self.saldo -= monto
13            print(f"Retiro de {monto} realizado. Saldo restante: {self.saldo}")
14        else:
15            print("Fondos insuficientes para el retiro.")
16
17    def mostrar_saldo(self):
18        print(f"Saldo actual: {self.saldo}")
19
20 saldo_inicial = float(input("Ingrese el saldo inicial: "))

```

```

21 cuenta = CuentaBancaria(saldo_inicial)
22
23 monto_deposito = float(input("Ingrese el monto a depositar: "))
24 cuenta.depositar(monto_deposito)
25
26 monto_retiro = float(input("Ingrese el monto a retirar: "))
27 cuenta.retirar(monto_retiro)
28
29 cuenta.mostrar_saldo()
30
31 del cuenta
32 try:
33     print(cuenta)
34 except NameError:
35     print("Objeto cuenta bancaria destruido")

```

Listing 2: Código Practica 2

3. Clase Temperatura: Conversión de Celsius a Fahrenheit

Problema:

Crea una clase Temperatura con:

Atributo: celsius.

Constructor que reciba la temperatura en °C.

Método a_fahrenheit() que convierta y retorne la temperatura en °F.

Destructor que indique: “Objeto Temperatura destruido”

Crea un objeto, muestra la temperatura en ambas escalas, y elimina el objeto.

Código en Python:

```

1 class Temperatura:
2     def __init__(self, celsius):
3         self.celsius = celsius
4         print("Objeto Temperatura creado")
5
6     def a_fahrenheit(self):
7         fahrenheit = (self.celsius * 9/5) + 32
8         return fahrenheit
9     def mostrar_temperatura(self):
10        print(f"Temperatura en Celsius: {self.celsius} C ")
11        print(f"Temperatura en Fahrenheit: {self.a_fahrenheit()} F ")
12
13 cel = float(input("Ingrese grados Celsius: "))
14 temp = Temperatura(cel)
15 temp.mostrar_temperatura()

```

```
15
16 del temp
17 try:
18     print(temp)
19 except NameError:
20     print("Objeto temperatura destruido")
```

Listing 3: Código Practica 3