Reto 4

1. ¿Se aplicó el paradigma funcional?

No, el código proporcionado no se ha escrito siguiendo estrictamente el paradigma funcional. El código se centra más en un enfoque procedural, donde las funciones interactúan con variables globales, mutan el estado y generan efectos secundarios.

Algunas falencias son:

Uso de variables globales: costos, valor\_arroba, y kilos\_recolectados.

Mutación de estados: la función gestion\_costos() modifica directamente el diccionario global costos.

Efectos secundarios: la mayoría de las funciones no sólo calculan valores, sino que también tienen efectos secundarios, como imprimir en consola o cambiar estados externos.  
  
Aciertos:

Encapsulación de la lógica en funciones: aunque no es exclusivo del paradigma funcional, el hecho de que la lógica esté encapsulada en funciones ayuda a que pueda ser reutilizada fácilmente.

* Sugerencias:
  1. Inmutabilidad: Evite cambiar el estado. En su lugar, devuelva nuevos valores. Por ejemplo, en lugar de modificar el diccionario costos directamente, puede devolver un nuevo diccionario con los valores actualizados.
  2. Funciones puras: Las funciones deben ser puras, es decir, dada una entrada, siempre devolverán la misma salida sin generar efectos secundarios.
  3. Evite el uso de variables globales: En lugar de depender de estados globales, pase los estados requeridos como argumentos a las funciones y devuelva nuevos estados como resultado.
  4. Map, Filter, Reduce: Aproveche estas funciones de alto orden para procesar colecciones de datos sin la necesidad de bucles explícitos.
  5. Funciones de alto orden: Cree funciones que toman otras funciones como argumentos y/o devuelven funciones como resultados.
  6. Ejemplo de aplicación del paradigma funcional:
  7. Podemos refactorizar gestion\_costos() para que sea más funcional:

def actualizar\_costo(costos, clave, valor):

nuevo\_costos = costos.copy()

nuevo\_costos[clave] = valor

return nuevo\_costos

def gestion\_costos\_funcional(costos):

eleccion = input("Seleccione una opción: ")

if eleccion == "1":

valor = float(input("Ingrese el costo de mano de obra: "))

return actualizar\_costo(costos, 'mano\_de\_obra', valor)

# ... Repetir para otras opciones

else:

return costos