Profundiza más

Recurso de Profundización

Condicionales y anidamiento (Parte 1)

Autor: Damián Nicolalde Rodríguez

Objetivo del Documento: Este recurso presenta un ejemplo avanzado que ilustra cómo combinar sentencias condicionales (if, elif, else) con operadores relacionales (>, <, ==, etc.) y validaciones para clasificar productos en distintas categorías. El enfoque va más allá de un simple if-else, empleando condiciones encadenadas (rango de precios, verificación de stock, revisión de fechas) y subcondiciones anidadas en casos puntuales. Se incluyen comentarios y explicaciones paso a paso, destacando mejores prácticas como la legibilidad de cada bloque, la moderación en el anidamiento y el uso de elif para evitar confusiones.

Recurso de Profundización 1: Ejemplo Avanzado – "Sistema de Categorización de Productos con Validaciones y Rangos"

Requerimientos:

1. Ingreso de Datos: Se solicita al usuario el nombre del producto, su precio y la cantidad de stock disponible, validando que el precio y el stock sean numéricos y mayores a cero.
2. Clasificación por Precio: Emplea if precio < 10: "Producto Económico", elif precio < 50: "Producto Estándar", else: "Producto Premium".
3. Verificación de Stock: Dentro de cada categoría, se anida un if adicional para chequear si la cantidad de stock supera un umbral (por ejemplo, 100 unidades) y decide si el producto se marca como "Abundante" o "Limitado".
4. Orden Alfabético (Opcional): Si se ingresa más de un producto, se compara su nombre con otros previamente guardados, determinando si debe insertarse en una lista ordenada lexicográficamente.
5. Salida Formateada: Usa f-strings para presentar un informe final donde se muestra el nombre, la categoría, la disponibilidad (Abundante/Limitada) y un mensaje que resume las decisiones tomadas en las condicionales.

Código Python.

```
# Ejemplo Avanzado: Sistema de Categorización de Productos con Validaciones y Rangos
# ------------------------------------------------------------------------
# Este código ilustra cómo combinar sentencias condicionales (if, elif, else) y
# operadores relacionales (>, <, ==, etc.) para clasificar productos en categorías
# basadas en su precio y stock. Además, demuestra la importancia de la validación
# y la legibilidad a la hora de tomar decisiones en el programa.
```

Profundiza más

```python
productos = []  # Lista para almacenar información de cada producto

while True:
    # Se solicita el nombre del producto
    nombre = input("\nIngrese el nombre del producto (o 'fin' para terminar): ").strip()

    # Si el usuario escribe 'fin', se sale del bucle principal
    if nombre.lower() == "fin":
        break

    # Validamos que el nombre no esté vacío
    if not nombre:
        print("Error: El nombre del producto no puede estar vacío.")
        continue  # Se vuelve al inicio del bucle para pedir un nuevo producto

    # Se solicita el precio, manejando excepciones en caso de que no sea numérico
    try:
        precio = float(input("Ingrese el precio del producto: "))
        if precio <= 0:
            print("Error: El precio debe ser un valor positivo.")
            continue  # Se vuelve al inicio si el precio no es válido
    except ValueError:
        print("Error: Debe ingresar un valor numérico para el precio.")
        continue

    # Se solicita el stock, manejando igualmente las excepciones
    try:
        stock = int(input("Ingrese la cantidad de stock disponible: "))
        if stock < 0:
            print("Error: El stock no puede ser negativo.")
            continue
    except ValueError:
        print("Error: Debe ingresar un número entero para el stock.")
        continue

    # ----------------------------------------------------------------------
    # Clasificación por precio (categoría del producto)
    # ----------------------------------------------------------------------
    if precio < 10:
        categoria = "Económico"
    elif precio < 50:
        categoria = "Estándar"
```

Profundiza más

```
else:
    categoria = "Premium"

# ---------------------------------------------------------------------------
# Verificación de stock (anidando condicionales dentro de la categoría)
# ---------------------------------------------------------------------------
# Se evalúa si el producto tiene mucho o poco stock
if stock > 100:
    disponibilidad = "Abundante"
else:
    disponibilidad = "Limitada"

# Se crea un diccionario con toda la información del producto
producto_info = {
    "nombre": nombre,
    "precio": precio,
    "stock": stock,
    "categoria": categoria,
```