LU02.A02 - Refactoring unpure -> pure

Sie haben eine Funktion, die in einem E-Commerce-System verwendet wird, um den Gesamtpreis eines Warenkorbs zu berechnen und einen Rabatt anzuwenden. Die vorhandene Funktion ist unpure, da sie globale Variablen verwendet und direkt den Gesamtpreis ausdruckt. Ihre Aufgabe ist es, diese Funktion in eine pure Funktion umzuwandeln.

Anleitung in Schritten

1. Untersuchen Sie die vorhandene unpure function und identifizieren Sie die Seiteneffekte und die Verwendung globaler Variablen.
2. Erstellen Sie eine neue pure function, die alle benötigten Informationen als Argumente erhält.
3. Die neue Funktion sollte alle Berechnungen durchführen und das Ergebnis zurückgeben, ohne den Gesamtpreis direkt auszudrucken.
4. Testen Sie die neue Funktion mit verschiedenen Eingaben, um sicherzustellen, dass sie korrekt funktioniert.

```
BASKET = [{'Produkt': 'T-Shirt', 'Preis': 20}, {'Produkt': 'Hose', 'Preis': 50}]
DISCOUNT = 0.1

# Unpure function
def calculate_total_unpure():
    global BASKET, DISCOUNT
    total = sum(item['Preis'] for item in BASKET)
    discount = total * DISCOUNT
    total = total - discount
    print("Gesamtpreis:", total)
    return total

# Pure function


if __name__ == "__main__":
    print("Unpure function:")
    calculate_total_unpure()

    print("Pure function:")
    total = calculate_total_pure(#TODO)
    print("Gesamtpreis:", total)
```

Zusammenfassung

Durch die Umwandlung der unpure function in eine pure function haben Sie eine bessere Trennung von Bedenken erreicht und die Testbarkeit der Funktion verbessert. Die neue Funktion ist jetzt unabhängig von globalen Zuständen und Seiteneffekten, was sie robuster und leichter zu verstehen macht.
