LU01.A09 - Vom Skript zur Funktion

Bauen Sie ein Lagerverwaltungs-Skript, das ausschliesslich aus Prozeduren mit globalem Zustand besteht, in echte Funktionen um – so, dass jeder Aufruf bei gleicher Eingabe immer dasselbe Ergebnis liefert.

Ausgangslage

Das folgende Skript funktioniert – solange man es genau in dieser Reihenfolge aufruft. Es zeigt gleich mehrere typische Schwächen prozeduralen Codes: globalen Zustand, Berechnung und Ausgabe im selben Atemzug, und Prozeduren, die `None` zurückgeben. Im Repository finden Sie es als `ausgangslage.py`.

```
lager = []
gesamtwert = 0


def artikel_hinzufuegen(name, menge, preis):
    global gesamtwert
    lager.append({'name': name, 'menge': menge, 'preis': preis})
    gesamtwert += menge * preis


def zeige_gesamtwert():
    print(f'Gesamtwert: {gesamtwert:.2f} CHF')


def zeige_teuersten():
    teuerster = None
    for artikel in lager:
        if teuerster is None or artikel['preis'] > teuerster['preis']:
            teuerster = artikel
    print(f"Teuerster Artikel: {teuerster['name']}")


if __name__ == '__main__':
    artikel_hinzufuegen('Maus', 10, 24.90)
    artikel_hinzufuegen('Tastatur', 4, 79.50)
    artikel_hinzufuegen('Monitor', 2, 249.00)
    zeige_gesamtwert()
    zeige_teuersten()
```

Anforderungen

1. **Analyse:** Halten Sie schriftlich fest, welche der drei `def`-Blöcke Prozeduren sind und woran Sie das erkennen.
2. **Umbau:** Implementieren Sie in `main.py` die vier vorgegebenen Funktionen so, dass gilt:
   - Keine globalen Variablen – alle benötigten Daten kommen als Parameter herein.
   - Kein `print` innerhalb der berechnenden Funktionen. Ausgegeben wird ausschliesslich in `main()`.
   - Jede Funktion gibt einen Wert zurück.
   - `artikel_hinzufuegen` verändert die übergebene Liste **nicht**, sondern gibt eine **neue** Liste zurück.

Vorgegebene Funktionen

Die Namen und Signaturen sind fix – die Tests rufen genau diese auf.

Funktion	**Rückgabe**
`artikel_hinzufuegen(lager, name, menge, preis)`	eine **neue** Liste mit dem zusätzlichen Artikel
`gesamtwert(lager)`	Summe aus Menge mal Preis, bei leerem Lager `0`
`teuerster_artikel(lager)`	der Artikel mit dem höchsten Preis, bei leerem Lager `None`
`formatiere_gesamtwert(lager)`	Text im Format `Gesamtwert: 1065.00 CHF`

Beispieloutput

```
Gesamtwert: 1065.00 CHF
Teuerster Artikel: Monitor
```

Kontrollfragen

1. Warum kann man den Ausdruck `gesamtwert(lager)` im Kopf jederzeit durch seine Zahl ersetzen, den Aufruf `artikel_hinzufuegen('Maus', 10, 24.90)` aus der Ausgangslage aber nicht? Wie heisst diese Eigenschaft?
2. Was gibt `zeige_gesamtwert()` zurück? Probieren Sie `ergebnis = zeige_gesamtwert()` und geben Sie `ergebnis` aus.
3. Welche Ihrer neuen Funktionen liessen sich mit einem automatischen Test prüfen, welche der alten Prozeduren nicht? Begründen Sie.

Hinweis

Schleifen sind in dieser Aufgabe ausdrücklich erlaubt – innerhalb einer Funktion ist eine `for`-Schleife völlig in Ordnung. Es geht hier nicht um den schleifenfreien

Stil, sondern um die Grenze zwischen **Funktion** und **Prozedur**: Liefert der Block einen Wert, oder *tut* er etwas?

Achtung beim Linting

`global` kostet in dieser Aufgabe Punkte (`W0603 global-statement` ist bewusst aktiviert) – genau darum geht es ja.

Bewertung

Teil	Punkte
Unittests	11
Linting	5
Total	**16**

Abgabe

Der Code wird über das Classroom-Repository abgegeben und automatisch bewertet. Die drei **Kontrollfragen** geben Sie zusätzlich in Moodle ab.
