LU07.A07 - Sicherheitsnetz zuerst

Schreiben Sie Characterization Tests für eine Funktion, deren Verhalten niemand mehr kennt - und refactoren Sie sie erst danach.

Ausgangslage

Diese Funktion läuft seit zwei Jahren im Notensystem. Dokumentation gibt es keine, Tests auch nicht.

```
def notenschnitt(punkte, maximum):
    if maximum == 0:
        return 1
    note = 1 + 5 * (sum(punkte) / len(punkte)) / maximum
    if note > 6:
        note = 6
    return round(note * 2) / 2
```

Detaillierte Aufgabenstellung

1. **Verhalten erkunden:** Rufen Sie die Funktion mit mindestens sechs Eingaben auf und protokollieren Sie die Ergebnisse. Denken Sie an Grenzfälle: leere Liste, `maximum = 0`, mehr Punkte als das Maximum, exakt die Höchstpunktzahl.
2. **Festhalten:** Schreiben Sie die beobachteten Werte als `pytest`-Tests. Auch dann, wenn Ihnen ein Ergebnis falsch vorkommt - der Test hält fest, was *ist*, nicht was sein sollte.
3. **Refactoren:** Machen Sie die Funktion lesbar. Sprechende Namen, benannte Konstanten, die Rundungsregel als eigene Funktion.
4. **Bewerten:** Notieren Sie, welche Ergebnisse Sie für fachlich falsch halten - und was zu tun wäre, um das zu ändern.

Leitfragen

- Was passiert bei `notenschnitt([], 20)`? Ist das ein Verhalten, das man testen kann?
- Warum liefert `notenschnitt([10], 0)` den Wert `1` und nicht `1.0`? Spielt das eine Rolle?
- Was bedeutet `round(note * 2) / 2` fachlich?
- Der Test hält ein Verhalten fest, das vielleicht ein Bug ist. Warum ist das trotzdem der richtige erste Schritt?

M323-LU07, M323-D1I

© Kevin Maurizi

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m323/learningunits/lu07/aufgaben/characterization?rev=1788945333

Last update: **2026/09/09 11:15**