Lernnachweis zu Kompetenz A1F

Der Einsatz von Immutable Values in der funktionalen Programmierung

Ziel: Erläuterung des Konzepts und der Bedeutung von Immutable Values im Rahmen der funktionalen Programmierung und der Abgrenzung zu Mutable Values in anderen Programmiersprachen.

Inhalt:

Immutable Values (unveränderliche Werte) sind essenziell in der funktionalen Programmierung und stellen sicher, dass Werte nach ihrer Initialisierung konstant bleiben. Im Unterschied dazu stehen Mutable Values, die nachträglich modifizierbar sind. Ein Beispiel für Immutable Values in Python ist das Tupel, welches nach der Erstellung nicht mehr verändert werden kann.

Codebeispiel für Immutable Values:

```
# Immutable Value - Tupel
my_tuple = (1, 2, 3)
my_tuple.append(4)  # AttributeError: 'tuple' object has no attribute 'append'
```

Codebeispiel für Mutable Values:

```
# Mutable Value - Liste
my_list = [1, 2, 3]
my_list.append(4)  # my_list ist jetzt [1, 2, 3, 4]
```

Relevanz:

Das Verständnis von Immutable Values ist entscheidend für die Entwicklung von reinen Funktionen, die keine Seiteneffekte verursachen. In der parallelen Verarbeitung verhindert die Unveränderlichkeit von Daten Zustandskonflikte zwischen Threads. Im Vergleich zu referenzierten Objekten in Sprachen wie Java, die verändert werden können, bieten Immutable Objects in funktionalen Sprachen wie Haskell oder Clojure einen präzisen und vorhersagbaren Codeablauf.