

# LU02.A11 - Immutable Dataclass



Erstellen Sie mehrere pure functions, die zusammenarbeiten, um eine komplexe Datenstruktur auf Basis von immutable Dataclasses zu manipulieren.

## Aufgabenstellung

Im Kontext der funktionalen Programmierung spielt die Unveränderlichkeit von Datenstrukturen eine zentrale Rolle. Durch die Verwendung von immutable Dataclasses stellen wir sicher, dass einmal erstellte Objekte nicht unbeabsichtigt verändert werden können. Dies führt zu vorhersehbarem Verhalten, erleichtert das Debugging und reduziert die Wahrscheinlichkeit von Seiteneffekten in Ihrem Code. In dieser Aufgabe lernen Sie, wie man immutable Dataclasses verwendet, um komplexe Datenstrukturen zu manipulieren, ohne den ursprünglichen Zustand zu verändern. Das fördert die Entwicklung von stabilen und wartbaren Programmen.

1. Definieren Sie eine `@dataclass(frozen=True)` namens `Student` mit den Attributen `name` (str), `grades` (List[int]), und `graduated` (bool).
2. Implementieren Sie eine Funktion `add_grade(student: Student, grade: int) → Student`, die eine neue Instanz der Dataclass `Student` zurückgibt, wobei die neue Note der Liste hinzugefügt wird.
3. Implementieren Sie eine Funktion `calculate_average(student: Student) → float`, die den Durchschnitt der Noten auf zwei Stellen gerundet berechnet .
4. Implementieren Sie eine Funktion `graduate_student(student: Student) → Student`, die eine neue Instanz der Dataclass zurückgibt, bei der das Attribut `graduated` auf `True` gesetzt wird, wenn der Notendurchschnitt 70 oder mehr beträgt.
5. Schreiben Sie ein Programm, das mehrere Noten hinzufügt, den Durchschnitt berechnet und den Studenten „absolviert“, falls die Kriterien erfüllt sind. Geben Sie die Ergebnisse aus.

## Schritt für Schritt

1. Definieren Sie die immutable Dataclass `Student` in einer separaten Datei `student.py`.
2. Implementieren Sie die Funktion `add_grade`, die eine neue Instanz zurückgibt und die Note hinzufügt.
3. Implementieren Sie die Funktion `calculate_average`, die den Notendurchschnitt berechnet.
4. Implementieren Sie die Funktion `graduate_student`, die eine neue Instanz zurückgibt, wenn der Student die Anforderungen erfüllt.
5. Führen Sie mehrere Funktionsaufrufe durch, um die Noten zu aktualisieren, den Durchschnitt zu berechnen und den Studenten bei Erfüllung der Kriterien zu absolvieren.



© Kevin Maurizi

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



Permanent link:  
<https://wiki.bzz.ch/modul/m323/learningunits/lu02/aufgaben/dataclass2>

Last update: **2024/08/28 11:46**