Lernnachweis B2F

Die Fähigkeit, Funktionen als Argumente für andere Funktionen zu verwenden und dadurch höherwertige Funktionen zu erstellen, wurde durch meine intensive Auseinandersetzung mit fortgeschrittenen Konzepten der funktionalen Programmierung gestärkt. Ein herausragendes Beispiel ist die Implementierung einer höheren Ordnungsfunktion, die verschiedene Filterkriterien akzeptiert.

Codebeispiel:

```
def filter_numbers(numbers, condition):
    # Funktion als Argument verwenden
    return [num for num in numbers if condition(num)]

# Funktionen als Argumente für höherwertige Funktionen verwenden
even_numbers = filter_numbers([1, 2, 3, 4, 5, 6], lambda x: x % 2 == 0)
positive_numbers = filter_numbers([-2, -1, 0, 1, 2], lambda x: x > 0)

print(f'Gerade Zahlen: {even_numbers}')
print(f'Positive Zahlen: {positive_numbers}')
```

Hier wird die Funktion filter_numbers genutzt, die eine Bedingung als Argument akzeptiert und die Liste entsprechend filtert.

Reflexion:

Die Anwendung von Funktionen als Argumente eröffnet die Möglichkeit zur dynamischen Gestaltung von Funktionen. Die Reflexion über die Kombination von Funktionen in höheren Ordnungsfunktionen hat mein Verständnis für die Flexibilität und Ausdrucksstärke funktionaler Programmierung vertieft.

Die Diskussionen in der Entwicklergemeinschaft über die Anwendung von Funktionen als Argumente für höherwertige Funktionen haben meine Fähigkeit gestärkt, effektive und wiederverwendbare Lösungen zu entwickeln. Diese Technik fördert nicht nur die Modularität, sondern ermöglicht auch die Schaffung abstrakter Funktionen, die auf unterschiedliche Szenarien anwendbar sind.

Zukünftige Schritte:

Um meine Fähigkeiten weiter zu vertiefen, plane ich, an Projekten teilzunehmen, die die Anwendung von Funktionen als Argumente für höherwertige Funktionen erfordern. Die kontinuierliche Anwendung und Erweiterung dieser Kompetenz werden meine Fähigkeiten in der funktionalen Programmierung weiter stärken und meine Fähigkeit zur Schaffung flexibler und abstrakter Lösungen fördern.