Lernnachweis A1G

Während meiner Auseinandersetzung mit Python habe ich nicht nur Selektionen und Iterationen mit Bedingungen erlernt, sondern auch die Eigenschaften von Funktionen im Vergleich zu anderen Programmierstrukturen, insbesondere Prozeduren. Eine Funktion ist in Python eine abgeschlossene Codeeinheit, die eine Aufgabe erfüllt und optional einen Wert zurückgeben kann. Im Gegensatz dazu sind Prozeduren sequenzielle Anweisungsblöcke ohne Rückgabewerte.

Code Beispiel:

```
def square_number(x):
    return x ** 2


result = square_number(4)
print(f'Das Quadrat ist {result}')
```

Hier illustriert die Funktion `square_number` den klaren Nutzen von Funktionen, indem sie den Quadratwert eines gegebenen Parameters zurückgibt. Im Vergleich dazu würde eine Prozedur denselben Code ausführen, aber ohne Rückgabewert.

Reflexion:

Die klare Struktur von Funktionen ermöglicht eine modulare und wiederverwendbare Codebasis. Anfangs war der Unterschied zu Prozeduren subtil, doch durch aktive Anwendung und Diskussionen in Peer-Reviews wurde mir die Bedeutung klarer. Die Reflexion über "pure functions" ohne Seiteneffekte stärkte mein Verständnis für sauberen und effizienten Code.

Zukünftige Schritte:

Um mein Verständnis zu vertiefen, plane ich, fortgeschrittene Funktionen wie Lambda-Funktionen zu erforschen und ihre Anwendung in komplexeren Algorithmen zu üben. Die Fähigkeit, Funktionen effektiv einzusetzen, ist entscheidend für sauberen Code, und ich freue mich darauf, diese Kenntnisse in zukünftigen Projekten weiterzuentwickeln.