Lernnachweis zu Kompetenz B1E

Implementierung von Funktionen in zusammenhängenden Algorithmen

Einleitung

Die Fähigkeit, Funktionen in zusammenhängenden Algorithmen zu implementieren, baut auf der vorherigen Kompetenz (B1F) auf, bei den Algorithmen in funktionale Teilstücke mit Rekursion aufgeteilt wurden. Jetzt werden wir betrachten, wie diese Teilstücke in einem übergeordneten Algorithmus verbunden werden können, um eine umfassende Funktionalität zu erreichen.

Beispiel: Rekursive Sortier- und Suchfunktion

Wir nehmen das Beispiel des rekursiven Bubble-Sort-Algorithmus und der rekursiven linearen Suche. Jetzt wollen wir diese beiden Funktionen in einem übergeordneten Algorithmus kombinieren.

```
def sort_and_search_recursive(arr, target):
    sorted_arr = bubble_sort_recursive(arr)
    index = linear_search_recursive(sorted_arr, target)
    return index

def bubble_sort_recursive(arr, n=None):
    if n is None:
        n = len(arr)
    if n == 1:
        return arr

    for i in range(n - 1):
        if arr[i] > arr[i + 1]:
            arr[i], arr[i + 1] = arr[i + 1], arr[i]

    return bubble_sort_recursive(arr, n - 1)

def linear_search_recursive(arr, target, index=0):
    if index == len(arr):
        return -1
    if arr[index] == target:
        return index
    return linear_search_recursive(arr, target, index + 1)
```

Die Hauptfunktion «sort_and_search_recursive» ruft die rekursiven Funktionen «bubble_sort_recursive» und «linear_search_recursive» auf. Zuerst wird das Array rekursiv sortiert, und dann wird die lineare Suche rekursiv angewendet, um das Ziel im sortierten Array zu finden.

Strukturierung der Dokumentation

Diese Dokumentation zeigt, wie die Implementierung von Funktionen in zusammenhängenden Algorithmen eine natürliche Weiterentwicklung der Fähigkeit ist, Algorithmen in funktionale Teilstücke zu zerlegen. Die Beispiele aus der vorherigen Kompetenz werden hier verwendet, um zu illustrieren, wie rekursive Sortier- und Suchfunktionen in einem übergeordneten Algorithmus integriert werden können.

Fazit

Die nahtlose Integration von Funktionen in einem zusammenhängenden Algorithmus ist entscheidend für die Entwicklung komplexer Softwarelösungen. Durch die Anwendung dieser Kompetenz wird nicht nur die Wiederverwendbarkeit von Code gefördert, sondern auch die Strukturierung und Lesbarkeit des Codes verbessert. Die hier gezeigten Beispiele verdeutlichen die praktische Anwendung dieser Fähigkeit in der Softwareentwicklung.