LU03b - Rekursion

Die Rekursion ist eine Technik in der Programmierung, bei der eine Funktion sich selbst aufruft, um ein Problem zu lösen. Rekursive Funktionen bestehen normalerweise aus zwei Teilen: dem Basisfall, der das Ende der Rekursion markiert, und dem rekursiven Fall, der die Funktion erneut mit einem Teilproblem aufruft. Rekursion kann für viele Probleme eine elegante und effektive Lösung bieten.

Rekursion muss mit Sorgfalt verwendet werden, da sie ohne einen klaren Basisfall zu einer Endlosschleife führen kann.

Beispiel 1: Fakultät

Die Fakultät einer Zahl ist das Produkt aller ganzen Zahlen von 1 bis zu dieser Zahl. Sie wird oft mit dem Symbol ! dargestellt, z.B. $5!=5\times4\times3\times2\times1=120$.

```
def factorial(n):
    if n == 0:  # Basisfall
        return 1
    else:       # Rekursiver Fall
        return n * factorial(n-1)
if __name__ == '__main__':
    print(factorial(5)) # Ausgabe: 120
```

Beispiel 2: Fibonacci-Reihe

Die Fibonacci-Reihe ist eine Folge von Zahlen, bei der jede Zahl die Summe der beiden vorhergehenden ist. Die ersten zwei Zahlen in der Fibonacci-Reihe sind 0 und 1.

```
def fibonacci(n):
    if n == 0:  # Basisfall 1
        return 0
    elif n == 1:  # Basisfall 2
        return 1
    else:       # Rekursiver Fall
        return fibonacci(n-1) + fibonacci(n-2)
if __name__ == '__main__':
    print(fibonacci(6)) # Ausgabe: 5
```

Rekursion ist eine mächtige Technik, die in vielen verschiedenen Problembereichen eingesetzt werden kann. Durch das Verständnis des Basisfalls und des rekursiven Falls

können Sie rekursive Funktionen erstellen, um eine Vielzahl von Problemen auf eine klare und elegante Weise zu lösen.

M323-LU03