Funktionen höherer Ordnung in Python

Einführung

Funktionen höherer Ordnung sind ein Kernkonzept der funktionalen Programmierung in Python. Sie ermöglichen es, Funktionen, die selbst andere Funktionen als Argumente nehmen oder als Ergebnis zurückgeben, zu definieren und zu verwenden. Diese Art von Funktionen erhöht die Flexibilität und Wiederverwendbarkeit des Codes.

Definition

Eine Funktion höherer Ordnung erfüllt mindestens eines der folgenden Kriterien:

- Sie nimmt eine oder mehrere Funktionen als Argumente.
- Sie gibt eine Funktion als Ergebnis zurück.

Beispiele

Funktionen als Argumente

```
def gruß(funktion):
    return funktion("Welt")

def hallo(name):
    return "Hallo " + name

def tschüss(name):
    return "Tschüss " + name

print(gruß(hallo))  # Ausgabe: Hallo Welt
print(gruß(tschüss))  # Ausgabe: Tschüss Welt
```

Funktionen als Rückgabewerte

```
def multiplikator(n):
    def multipliziere(m):
        return m * n
    return multipliziere

verdopple = multiplikator(2)
verdreifache = multiplikator(3)

print(verdopple(5))  # Ausgabe: 10
print(verdreifache(5))  # Ausgabe: 15
```

Anwendung in der funktionalen Programmierung

Funktionen höherer Ordnung sind besonders nützlich in der funktionalen Programmierung, da sie es ermöglichen, allgemeine Muster der Datenverarbeitung wie Mapping, Filtering und Folding zu abstrahieren.

Mapping mit `map()`

```
zahlen = [1, 2, 3, 4, 5]
quadrate = list(map(lambda x: x**2, zahlen))
print(quadrate)  # Ausgabe: [1, 4, 9, 16, 25]
```

Filtering mit `filter()`

```
zahlen = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
gerade = list(filter(lambda x: x % 2 == 0, zahlen))
print(gerade)  # Ausgabe: [2, 4, 6, 8, 10]
```