Funktionsobjekte in Python

Einführung

In Python sind Funktionen Objekte erster Klasse. Das bedeutet, dass Funktionen genau wie andere Werte behandelt werden: Sie können als Argumente übergeben, von anderen Funktionen zurückgegeben und Variablen zugewiesen werden.

Funktionen als Objekte

Jede Funktion in Python ist ein Funktionsobjekt. Diese Objekte können behandelt werden wie jeder andere Wert.

Beispiel

```
def gruß():
    print("Hallo Welt!")

f = gruß
f()
```

In diesem Beispiel wird die Funktion `gruß` einer Variablen `f` zugewiesen und dann durch `f()` aufgerufen.

Funktionen als Argumente

Funktionen können anderen Funktionen als Argumente übergeben werden.

Beispiel

```
def gruß(funktion):
    funktion()

def sage_hallo():
    print("Hallo!")

gruß(sage_hallo)
```

Funktionen als Rückgabewerte

Funktionen können auch von anderen Funktionen zurückgegeben werden.

Beispiel

```
def erstelle_addierer(x):
    def addierer(y):
        return x + y
    return addierer

addiere_5 = erstelle_addierer(5)
print(addiere_5(10))  # Ausgabe: 15
```

Funktionen höherer Ordnung

Eine Funktion höherer Ordnung ist eine Funktion, die eine oder mehrere Funktionen als Argumente akzeptiert oder eine Funktion als Ergebnis zurückgibt.

Beispiel

```
def operation_anwenden(x, y, operation):
    return operation(x, y)

def addiere(x, y):
    return x + y

def subtrahiere(x, y):
    return x - y

print(operation_anwenden(5, 3, addiere))  # Ausgabe: 8
print(operation_anwenden(5, 3, subtrahiere))  # Ausgabe: 2
```

Anonyme Funktionen: `lambda`

In Python können anonyme Funktionen mit dem Schlüsselwort `lambda` erstellt werden. Diese sind nützlich, wenn eine einfache Funktion für eine kurze Zeit benötigt wird.

Beispiel

```
addiere = lambda x, y: x + y
print(addiere(5, 3))  # Ausgabe: 8
```