**Lernnachweis zu Kompetenz B3G:**

**Kompetenz:** Ich kann einfache Lambda-Ausdrücke schreiben, die eine einzelne Operation durchführen, z.B. das Quadrieren einer Zahl oder das Konvertieren eines Strings in Großbuchstaben.

**Lernnachweis: Verwendung von Lambda-Ausdrücken in Flask-Anwendung**

In dieser Lernsequenz habe ich einfache Lambda-Ausdrücke geschrieben, um einzelne Operationen durchzuführen. Eine Flask-Anwendung wurde erstellt, um dies zu demonstrieren.

**Funktionalität im Code:** Der Code nutzt Lambda-Ausdrücke in einer Flask-Anwendung. Die Route /lambda\_operations führt Lambda-Funktionen für Quadrieren und Großschreibung aus, zeigt die Ergebnisse an.

**Umsetzung im Code mit Flask:**

Ein Bild, das Text, Screenshot, Software enthält.

Automatisch generierte Beschreibung

**Reflexion:** Die Umsetzung zeigt die Anwendung von Lambda-Ausdrücken in Python. Die Flask-Anwendung bietet Zugriff auf die Route /lambda\_operations, wodurch zwei Lambda-Ausdrücke erstellt und auf Werte angewendet werden. Dies verdeutlicht die Einfachheit und Klarheit von Lambda-Ausdrücken für einzelne Operationen.

**Lernnachweis zu Kompetenz B3F:**

**Kompetenz:** Ich kann Lambda-Ausdrücke schreiben, die mehrere Argumente verarbeiten können.

Die Fähigkeit, Lambda-Ausdrücke mit mehreren Argumenten zu erstellen, wurde in dieser Lernsequenz entwickelt. Dabei werden zwei Lambda-Funktionen in einer Flask-Anwendung verwendet: add\_numbers für die Addition von zwei Zahlen und concatenate\_strings für die Konkatenation von zwei Strings.

**Funktionalität im Code:** Die Route /lambda\_multiple\_arguments demonstriert die Verwendung von Lambda-Ausdrücken mit mehreren Argumenten. Die Lambda-Funktionen add\_numbers und concatenate\_strings nehmen jeweils zwei Argumente entgegen und führen die Addition bzw. Konkatenation aus.

**Umsetzung im Code mit Flask:**

**Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung**

**Reflexion:** Die Integration von Lambda-Ausdrücken mit mehreren Argumenten ermöglicht eine kompakte und flexible Gestaltung von Funktionen. In der Flask-Anwendung werden zwei solcher Lambda-Ausdrücke verwendet: add\_numbers für die Addition von zwei Zahlen und concatenate\_strings für die Konkatenation von zwei Strings. Dies betont die Vielseitigkeit und Einfachheit von Lambda-Ausdrücken in Python, insbesondere in funktionalen Programmierparadigmen.

**Lernnachweis zu Kompetenz B3E:**

**Kompetenz:**

Ich kann Lambda-Ausdrücke verwenden, um den Programmfluss zu steuern, z.B. durch Sortieren von Listen basierend auf benutzerdefinierten Kriterien.

**Lernnachweis:**

Während dieser Lernsequenz habe ich Lambda-Ausdrücke eingesetzt, um den Programmfluss zu steuern. Insbesondere wurden Listen basierend auf benutzerdefinierten Kriterien sortiert.

**Funktionalität im Code:**

Die Flask-Anwendung bietet eine Route /lambda\_sort, die eine Liste von Wörtern nach ihrer Länge sortiert. Dies erfolgt mithilfe eines Lambda-Ausdrucks, der die Länge jedes Wortes als Sortierkriterium verwendet.

**Umsetzung im Code mit Flask:**

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

**Reflexion:**

Die Anwendung von Lambda-Ausdrücken zur Steuerung des Programmflusses ermöglicht eine elegante und kompakte Lösung, insbesondere beim Sortieren von Listen nach benutzerdefinierten Kriterien. Diese Technik trägt zu einer klareren und effizienteren Codegestaltung bei.