Source Code 1. Mutable und immutable Datentypen in Python.

```
# Jede Variable in Python bezeichnet ein "Objekt". Wenn ein Objekt
# erzeugt wird, erhält es eine eindeutige "object id" (automatisch,
# für die Programmiererin unsichtbar) und einen Datentyp, der später
# nicht mehr verändert werden kann.
# Beispiel: Variable a bezeichnet ein neu erzeugtes Objekt vom
# Datentyp int (integer, englisch für "ganze Zahl")
a = 42
# Sehr wohl kann aber ein und dieselbe Variable a zuerst auf int
# zeigen und dann auf string (englisch für: Zeichenkette); das
# ursprünglich erzeugte int-Objekt wird dadurch gelöscht (denn
# es gibt nun keinen "Namen" mehr, der das Objekt bezeichnet):
a = "zweiundvierzig"
# Bei den Datentypen in Python gibt es einen wichtigen Unterschied:
# - Immutable (unveränderbar) sind int, float, bool, string, unicode, tuple
#   (also ganze Zahlen, Fließkommazahlen, Boolesche Wahrheitswerte,
#   Zeichenketten, Zeichenketten mit "unicode"-Buchstaben und geordnete
#   Tupel)
# - Mutable (veränderbar) sind list, dict, set sowie (in der Regel)
#   abgeleitete Klassen (also geordnete Listen, Dictionaries, Mengen
#   und "selbst programmierte Datentypen" - dazu später mehr)
# Beispiel: Liste ist veränderbar
a = [1,2,3]
print(a)
a[0] = 42
print(a)
# Beispiel: Tupel ist unveränderbar
a = (1,2,3)
print(a)
# Der Befehl "a[0] = 42" führt daher zu einer Fehlermeldung, die
# wir vorsorglich "abfangen" (mit try - except) und als
# formatierten String ausgeben (wird später etwas genauer
# erklärt):
try:
    a[0] = 42
except TypeError as i_told_you_so:
    print(f'Das war ein Fehler: "{i_told_you_so}!!!"')
```