# Eigene Exceptions und gezielte Fehlerbehandlung in Python

**Übung 1: Alter validieren mit raise**

Ein Formular prüft das Alter eines Benutzers.

Es gelten folgende Regeln:

- Das Alter darf nicht negativ sein.
- Es darf höchstens 130 Jahre alt sein.

Wenn eine der Regeln verletzt wird, soll eine Exception ausgelöst werden.

- Verwende keine eigene Exception-Klasse, sondern löse ValueError mit raise manuell aus.
- Die Funktion soll überprüfen, ob das eingegebene Alter in Ordnung ist. Wenn es passt, soll eine Nachricht erscheinen. Wenn nicht, soll ein Fehler ausgelöst werden.
- Nutze try-except, um ungültige Fälle zu erkennen und eine passende Fehlermeldung auszugeben.

# Eigene Exceptions und gezielte Fehlerbehandlung in Python

**Übung 2: Benutzerdefinierte Exception mit Argumenten**

Erstellen Sie eine benutzerdefinierte Exception-Klasse namens UngueltigerBereichError, die ausgelöst wird, wenn eine Zahl außerhalb eines vorgegebenen Bereichs liegt.

Die Exception-Klasse soll den ungültigen Wert als Argument annehmen und eine spezifische Fehlermeldung generieren, die den Wert enthält.

Schreiben Sie eine Funktion, die eine Zahl überprüft, ob sie innerhalb eines bestimmten Bereichs liegt (z.B zwischen 1 und 10), und die Exception auslöst, wenn dies nicht der Fall ist.

Diese Übung soll die Verwendung von Argumenten in benutzerdefinierten Exceptions vertiefen.

# Eigene Exceptions und gezielte Fehlerbehandlung in Python

**Übung 3- Benutzerdefinierte Exception: Ungültiger Rabatt**

Du entwickelst ein System für einen Online-Shop. Der Shop erlaubt Rabatte zwischen 0 % und 50 %. Wenn ein Benutzer versucht, einen Rabatt über 50 % einzugeben, soll eine eigene Exception namens UngültigerRabatt ausgelöst werden.

- Definiere eine eigene Exception-Klasse mit Attributen für den übergebenen und den maximal erlaubten Rabatt.
- Erzeuge eine aussagekräftige Fehlermeldung.
- Überlege dir, in welcher Funktion die Exception mit raise geworfen werden sollte.
- Behandle den Fehler in einem try-except-Block und gib eine verständliche Fehlermeldung aus.

**Übung 4 - Bankkonto prüfen mit eigener Exception**

Ein Benutzer möchte Geld von seinem Bankkonto abheben. Wenn der gewünschte Betrag das aktuelle Guthaben übersteigt, soll eine eigene Exception ausgelöst werden, z. B. NichtGenugGeld.

- Erstelle eine eigene Exception-Klasse mit Attributen für den gewünschten Betrag und das verfügbare Guthaben.
- Baue eine Funktion abheben(betrag, guthaben), die den Fehler prüft , ob genügend Guthaben vorhanden ist, und ggf. eine Exception auslöst.
- Behandle die Exception im try-except-Block.
- Drucke im Fehlerfall eine personalisierte Nachricht, z. B. "Sie möchten 100 €, aber es sind nur 20 € verfügbar."

**Übung 5 - Buchungssystem mit raise und eigenen Fehlerklassen**
In einem Buchungssystem können maximal 5 Plätze für eine Veranstaltung gebucht werden. Ein Benutzer versucht, eine bestimmte Anzahl von Plätzen zu buchen.

- Erstelle eine eigene Exception ZuVielePlaetzeGebucht, wenn ein Benutzer mehr Plätze anfragt, als verfügbar sind ( maximal 5).
- Speichere die Attribute angefragt und verfügbar in der Fehlerklasse.
- Erstelle eine Funktion buchen(anzahl, verfügbar), die die Buchung vornimmt oder den Fehler auslöst.
- Gib eine sinnvolle Fehlermeldung im Fehlerfall aus.

**Übung 6 - Prüfung der Passwortqualität**

Du sollst eine Funktion schreiben, die prüft, ob ein Passwort "sicher" ist. Ein Passwort gilt als sicher, wenn es:

- mindestens 8 Zeichen hat,
- mindestens eine Zahl enthält,
- mindestens einen Großbuchstaben enthält.

Wenn eine dieser Bedingungen nicht erfüllt ist, soll jeweils eine eigene Exception geworfen werden.

- Definiere drei benutzerdefinierte Exception-Klassen (eine pro Regel).
- Erstelle eine Funktion check_passwort(passwort) mit entsprechender Logik.
- Baue eine sinnvolle try-except-Struktur zur Behandlung der Fehler.
- Drucke passende Fehlernachrichten je nach Art des Fehlers.

## Hilfreiche Ressourcen:

https://www.datacamp.com/tutorial/exception-handling-python

https://docs.python.org/3/library/exceptions.html#exception-hierarchy

https://www.geeksforgeeks.org/python-exception-handling/

https://www.programiz.com/python-programming/exceptions

https://www.w3schools.com/python/python_try_except.asp