Aufgabe 4 - Exceptions auslösen und bearbeiten

Ziele

- Sie können eigene Exceptions erzeugen und werfen.
- Sie können mit `try…except` den Fehler auffangen und behandeln.

Ausgangslage

In der Applikation können eine Anzahl Noten erfasst und ausgelesen werden. Bisher wurde aber noch keine Fehlerbehandlung realisiert. Zum Beispiel stürzt das Programm ab, wenn man eine undefinierte Noten lesen will (falscher Index). Ungültige Noten werden antgegengenommen und wenn zuviele Noten erfasst werden, passiert auch nichts.

Für eine gezielte Fehlerbehandlung definieren wir daher 3 eigene Exceptions:

- ListIndexException: Die Liste hat weniger Elemente als der angegebene Index.
- ListRangeException: Es sollen mehr Elemente als zulässig eingefügt werden.
- ValueRangeException: Der Notenwert ist ungültig (kleiner 1.0 oder grösser 6.0)

Vorbereitung

1. Laden Sie das Repo von github-classroom.
2. Führen Sie das Programm aus. Was fällt Ihnen bezüglich dem Programmablauf auf?

Auftrag 1: Ungültiger Index bei ''take_grade''

Prüfen Sie in der Methode `take_grade(self, index)` ob der Index auf ein gültiges Listenelement zeigt.
Beachten Sie, dass die maximale Grösse - durch `self._MAX_GRADE_COUNT = 5` festgelegt - nicht dem Index der erfassten Noten entspricht!

1. Erstellen Sie die Klasse `ListIndexException` in der Datei `main.py`.
   Die Fehlermeldung lautet: „*Fehler: Ungültiger Index: …*“, wobei … für den aktuellen Wert des Index steht.
2. Passen Sie die Methode `take_grade()` so an, dass bei einem ungültigen Index eine `ListIndexException` erzeugt wird.
3. Testen Sie die Methode `take_grade()` mit den Testfällen `test_list_index_ok` und `test_list_index_exception`.

Auftrag 2: Zufügen von Noten in ''add_grade'' überwachen

Prüfen Sie in der Methode `add_grade(self, grade)` ob der Wert einer gültigen Note (1.0 ... 6.0) entspricht und ob nicht zuviele Noten zugefügt werden.
Die Grösse der Liste wird - wie oben erwähnt - durch `self._MAX_GRADE_COUNT` festgelegt.

1. Erstellen Sie die Klasse `ListRangeException` und `ValueRangeException` in der Datei `main.py`.
   Die Fehlermeldung für `ListRangeException` lautet : „*Fehler: Zu viele Werte eingegeben*“.
   Die Fehlermeldung für `ValueRangeException` lautet: „*Fehler: Der Notenwert muss im Bereich 1.0 bis 6.0 liegen. Er beträgt jedoch ...*“, wobei ... für den aktuellen Notenwert steht.
2. Passen Sie die Methode `add_grade()` so an, dass bei einem falschen Notenwert die Exception `ValueRangeException` und bei zu vielen zugefügten Werten `ListRangeException` erzeugt werden.
3. Testen Sie die Methode `add_grade()` mit den Testfällen `test_value_range_ok`, `test_value_range_low`, `test_value_range_high` und `test_number_too_large`.

Auftrag 3: Exceptions in ''main'' überwachen und beabreiten

In der Methode `main()` müssen die Exception bearbeitet werden. Überall wo die Methode `add_grade()` bzw. `take_grade()` aufgerufen werden, müssen Sie dies in einem `try` / `except`-Block machen.

Innerhalb eines `try`-Blocks wird die Programmausführung unterbrochen, sobald eine Exception auftritt, d.h. dass der nachfolgende Code im Block nicht ausgeführt wird. Damit in unserem Fall das Programm weiter läuft, schützen Sie bitte nur die kritischen Bereiche explizit mit `try-except` (Also nur den einen Aufruf der Methoden `add_grade(…)` bzw- `take_garde(…)`).

Falls eine Exception auftritt, wird die Standardmeldung der Exception ausgegeben.

1. Prüfen Sie in einem ersten Teil des Codes die Anzahl der zugefügten Noten.
2. Prüfen Sie dann einen ungültigen Notenwert.
3. Prüfen Sie dann den Zugriff auf eine nicht vorhandene Note.
4. Führen Sie das Programm aus. Es darf zu keinem Programmabbruch mehr kommen.

Dauer

45 Minuten

Abgabe

auf github-classroom

---

m320-LU03