A.3 Fehlermeldungen verstehen

Eine Fehlermeldung liest du **von unten nach oben**: In der letzten Zeile stehen die Art des Fehlers und eine kurze Beschreibung. Darüber steht, in welcher Zeile (*line*) der Fehler aufgetreten ist. Die Tabellen zeigen die häufigsten Meldungen, ihre Bedeutung und was du tun kannst. Die Meldungen stammen aus Python 3.13; andere Versionen formulieren manche etwas anders, die Fehlerart am Anfang bleibt aber gleich.

A.3.1 Fehler beim Schreiben (das Programm startet gar nicht)

Meldung	Bedeutung	Was du tun kannst
`SyntaxError: unterminated string literal …`	Ein Text ist nicht abgeschlossen.	Fehlendes Anführungszeichen ergänzen.
`SyntaxError: '(' was never closed`	Eine Klammer wurde nicht geschlossen.	Klammern zählen: Zu jeder `(` gehört eine `)`.
`SyntaxError: expected ':'`	Nach `if`, `elif`, `else`, `for`, `while` oder `def` fehlt der Doppelpunkt.	Doppelpunkt am Zeilenende ergänzen.
`SyntaxError: invalid syntax. Maybe you meant '==' …`	In einer Bedingung steht `=` statt `==`.	Zum Vergleichen `==` schreiben.
`IndentationError: expected an indented block …`	Nach einer Zeile mit Doppelpunkt fehlt die Einrückung.	Die folgenden Zeilen um 4 Leerzeichen einrücken.
`IndentationError: unexpected indent`	Eine Zeile ist eingerückt, obwohl sie es nicht sein darf.	Leerzeichen am Zeilenanfang entfernen.
`IndentationError: unindent does not match any outer indentation level`	Die Einrückung passt zu keiner Ebene.	Gleich viele Leerzeichen verwenden wie die Zeilen darüber.

Gymnasium Ottweiler

A.3.2 Fehler beim Ausführen (das Programm bricht ab)

Meldung	Bedeutung	Was du tun kannst
`NameError: name 'x' is not defined`	Der Name ist unbekannt.	Tippfehler? Groß-/Kleinschreibung? Variable vorher angelegt? Anführungszeichen vergessen?
`TypeError: can only concatenate str (not "int") to str`	Text und Zahl wurden mit `+` verbunden.	Zahl mit `str(...)` umwandeln oder Kommas in `print` verwenden.
`TypeError: unsupported operand type(s) for +: 'int' and 'str'`	Mit einem Text wurde gerechnet.	Eingabe mit `int(...)` oder `float(...)` umwandeln.
`TypeError: f() missing 1 required positional argument: 'a'`	Beim Aufruf einer Funktion fehlt ein Wert für einen Parameter.	Beim Aufruf so viele Werte angeben, wie die Funktion Parameter hat.
`ValueError: invalid literal for int() with base 10: '3.5'`	Der Text lässt sich nicht in eine ganze Zahl umwandeln.	Eingabe prüfen; für Kommazahlen `float(...)` verwenden, Punkt statt Komma.
`ValueError: could not convert string to float: '2,5'`	Der Text ist keine Kommazahl (oft: Komma statt Punkt).	Punkt statt Komma eingeben.
`ZeroDivisionError: division by zero`	Es wurde durch 0 geteilt.	Vorher mit `if` prüfen, ob der Teiler 0 ist.
`IndexError: list index out of range`	Es gibt keine Listenposition mit diesem Index.	Index beginnt bei 0; der größte Index ist `len(liste) - 1`.
`IndexError: string index out of range`	Dasselbe bei einer Zeichenkette.	wie oben
`ValueError: list.remove(x): x not in list`	Der Wert, der entfernt werden soll, steht nicht in der Liste.	Vorher mit `in` prüfen, ob der Wert enthalten ist.
`AttributeError: 'int' object has no attribute 'append'`	Die Methode passt nicht zum Datentyp.	Prüfen, ob die Variable wirklich eine Liste (bzw. Zeichenkette) ist.

A.3.3 Fehler ohne Fehlermeldung

Manche Fehler meldet Python nicht, weil das Programm „richtig geschrieben“ ist – es tut nur nicht das, was du willst. Sie sind am schwersten zu finden.

Was passiert	mögliche Ursache
Statt `7` kommt `34` heraus.	Eingaben nicht umgewandelt: `"3" + "4"` hängt Texte aneinander.
Das Programm hört nicht mehr auf.	Endlosschleife: Die Bedingung der `while`-Schleife wird nie falsch. Abbrechen mit dem Stopp-Knopf bzw. **Strg + C**.
Eine Schleife läuft einmal zu wenig.	`range(1, 10)` endet bei 9, nicht bei 10.
Die Summe ist falsch, obwohl die Schleife stimmt.	Die Summenvariable wird in der Schleife (statt davor) auf 0 gesetzt.
Eine Funktion gibt `None` zurück.	In der Funktion steht `print(...)` statt `return ...`, oder `return` fehlt.
Eine Zeile wird zu oft oder zu selten ausgeführt.	Falsche Einrückung: Die Zeile gehört (nicht) zum Schleifen- oder `if`-Block.

Tipp

Wenn du einen Fehler nicht findest: Lass dir mit zusätzlichen `print`-Anweisungen anzeigen, welche Werte die Variablen gerade haben. Oder schreibe ein Ablaufprotokoll. Lösche die zusätzlichen Ausgaben, wenn der Fehler gefunden ist.