

# Probleme lösen

by Woche 3

Beim Programmieren ist es unvermeidlich, dass ab und zu ein `Error` (Fehler) oder ein unerwartetes Ergebnis auftritt. So wirst auch du an diesem Punkt dieses Kurses wahrscheinlich schon mindestens einen Tippfehler gemacht haben, der zu einem Fehler geführt hat. Fehler können durch viele verschiedene Ursachen entstehen, wie z.B. falsche Benutzereingaben oder fehlende Dateien.

Es folgen drei Beispiele für Fehler, die in Python auftreten können. Das erste ist ein `SyntaxError`, der auftritt, wenn der Code nicht korrekt geschrieben ist. Dies kann z.B. durch fehlende Klammern oder Anführungszeichen passieren. Das zweite Beispiel ist ein `ZeroDivisionError`, der auftritt, wenn eine Division durch 0 durchgeführt werden soll, was mathematisch nicht möglich ist. Schließlich wird auch ein `NameError` gezeigt, der auftritt, wenn eine Variable genutzt wird, die nicht definiert ist.

```
print("Hallo)
```

```
SyntaxError: unterminated string literal (detected at line 1) (2791752001.py,
line 1)
  [36mCell[39m[36m [39m[32mIn[1][39m[32m, line 1[39m
[31m    [39m[31mprint("Hallo)[39m
      ^
[31mSyntaxError[39m[31m:[39m unterminated string literal (detected at line
1)
```

```
x = 10 / 0
```

```
ZeroDivisionError: division by zero
[31m-----[
[39m
[31mZeroDivisionError[39m                                Traceback (most recent
call last)
[36mCell[39m[36m [39m[32mIn[2][39m[32m, line 1[39m
[32m----> [39m[32m1[39m x = [32;43m10[39;49m[43m [49m[43m/[49m[43m
[49m[32;43m0[39;49m
[31mZeroDivisionError[39m: division by zero
```

x

```

NameError: name 'x' is not defined
-----
[39m
[31mNameError[39m                                Traceback (most recent
call last)
[36mCell[39m[36m [39m[32mIn[3][39m[32m, line 1[39m
[32m----> [39m[32m1[39m [43mx[49m

[31mNameError[39m: name 'x' is not defined

```

Spätestens am Ende dieses Kurses werdet ihr euch in einer Situation wiederfinden, in der ihr einen Fehler selbstständig finden und beheben müsst, den ihr auch nach mehreren Minuten nicht zu verstehen scheint. Fehler zu finden und zu beheben heißt im Programmieren auch “debuggen”<sup>1</sup>. Es muss klar sein, dass debuggen im Endeffekt keine unvorteilhafte Sondersituation ist, die euch von eurem Lernfortschritt abhält. Stattdessen sollte jede solcher Fehlermeldungen, die ihr erhaltet, als eine debug-Lektion angesehen werden, die euch dabei hilft, besser zu werden. Langfristig werden euch bestimmte Fehler oder Probleme sowieso unterkommen, und ihr werdet sie dann auch beheben müssen.

---

<sup>1</sup>Eine interessante Geschichte zum Begriff “debuggen” findet sich hier.

## Programmers. Everyday.



### Fehlermeldungen lesen

Man erkennt an den Fehlern oben, dass die Fehlermeldungen zumindest ein Stück weit versuchen beim Programmieren zu helfen. Sie geben Hinweise darauf, wo der Fehler aufgetreten ist und was der Fehler ist. In der Regel lernt man aus Fehlern spätestens nach dem zweiten oder dritten Mal und macht sie dann entweder nicht mehr, oder aber man bekommt aufgrund der Fehlermeldung ein gutes Gefühl dafür, wo man welchen Fehler gemacht haben könnte.

### Fehlerposition finden

Die Fehlermeldung verweist oft auf die Zeile, in der der Fehler aufgetreten ist:

```
] : print("Hallo")
```

```
File "<ipython-input-1-acaa7a899a46>", line 1
    print("Hallo")
      ^
```

**SyntaxError:** EOL while scanning string literal

```
] : a = 10 + 0
    b = 10 / 0
    c = 10 * 0
    d = 10 - 0
```

```
-----
ZeroDivisionError                                Traceback (most recent call last)
<ipython-input-2-37ced462d2be> in <module>
      1 a = 10 + 0
----> 2 b = 10 / 0
      3 c = 10 * 0
      4 d = 10 - 0
```

**ZeroDivisionError:** division by zero

Manchmal ist das aber nicht die Zeile, in der der Fehler bzw. das Problem tatsächlich aufgetreten ist, sondern die Zeile, in der es bemerkt wurde. So oder so ist eine simple Möglichkeit die Problemursache zu finden, den Code Zeile für Zeile auszuführen und zu schauen, wo der Fehler bzw. unerwartete Ergebnisse auftreten. Da wir mit Jupyter Notebooks arbeiten, ist das gut möglich, da der Code sowieso in separaten Zellen ausgeführt wird.

Führt man die letzte Zelle der beiden folgenden aus, gibt es einen Fehler. Das liegt nicht daran, dass der Code in der letzten Zelle falsch ist (er ist ja korrekt), sondern daran, dass in der Zelle davor ein Fehler gemacht wurde. Die Zelle davor hat wiederum 3 von 4 Zeilen Code, die einwandfrei funktionieren, sodass der Grund für den Fehler auch hier erst gesucht werden muss. In diesem Fall ist es bei so wenig Code natürlich kein Problem, es muss aber klar sein, dass das bei größeren Projekten mit größeren Zellen ganz schnell weniger offensichtlich wird was die Ursache für einen Fehler ist.

```
a = 10 + 0
b = 10 / 0
c = 10 * 0
d = 10 - 0
```

```
ZeroDivisionError: division by zero
[31m-----[39m
[39m
[31mZeroDivisionError[39m                                Traceback (most recent
call last)
[36mCell[39m[36m [39m[32mIn[4][39m[32m, line 2[39m
[32m      1[39m a = [32m10[39m + [32m0[39m
[32m----> [39m[32m2[39m b = [32;43m10[39;49m[43m [49m[43m/[49m[43m
[49m[32;43m0[39;49m
[32m      3[39m c = [32m10[39m * [32m0[39m
[32m      4[39m d = [32m10[39m - [32m0[39m

[31mZeroDivisionError[39m: division by zero
```

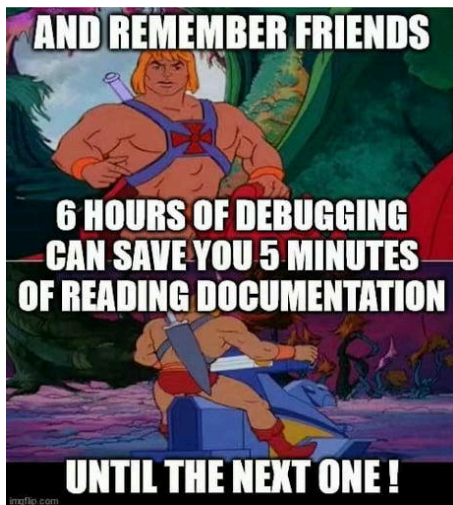
```
a+b+c+d
```

```
NameError: name 'b' is not defined
[31m-----[39m
[39m
[31mNameError[39m                                Traceback (most recent
call last)
[36mCell[39m[36m [39m[32mIn[5][39m[32m, line 1[39m
[32m----> [39m[32m1[39m a+[43mb[49m+c+d

[31mNameError[39m: name 'b' is not defined
```

## Dokumentation lesen

Der Vollständigkeit halber sei gesagt, dass einige Fehler nicht auftreten würden, wenn man vor Nutzung einer Funktion oder Methode die Dokumentation lesen würde. Das ist auch übertragbar aufs echte Leben mit den Bedienungsanleitungen von Geräten. Letztendlich brauchen wir uns aber nichts vormachen und wissen, dass das in der Praxis schlichtweg selten passiert. Trotzdem muss klar sein, dass die Dokumentation die Quelle ist, die von den Autoren des Codes selbst stammt und deshalb ggf. eine bessere Quelle ist als Rat von dritten.



## Google und Stackoverflow



Hat man den Fehler gefunden, versteht aber auch nach genauerem Hinsehen nicht, warum der Fehler aufgetreten ist, ist es ratsam, die Fehlermeldung zu googlen. Die Wahrscheinlichkeit ist hoch, dass jemand anderes schon einmal das gleiche Problem hatte und eine Lösung gefunden hat. Die Seite Stackoverflow ist hierbei eine sehr gute Anlaufstelle und wahrscheinlich sowieso unter den ersten Treffern deiner Google-Suche. Eine kurze Übersicht gibt es im Video weiter unten.

## ChatGPT



Hättest du diesen Kurs vor 2023 gemacht, wäre Stackoverflow wohl der beste Anlaufpunkt gewesen. Seit November 2022 gibt es aber ChatGPT (und andere LLM), also KI, die in der Lage ist, auf Fragen zu antworten und Probleme zu lösen. Das Thema ChatGPT (oder dessen Alternativen) ist so umfangreich, dass es hier nicht ausreichend besprochen werden kann. Hier aber eine Kurzversion:

ChatGPT ist eine sehr gute Alternative zu Stackoverflow und aus dem Alltag vieler Programmierer jetzt schon nicht mehr wegzudenken. Natürlich macht ChatGPT auch Fehler und ist nicht bei allen Problemen hilfreich. Dennoch ist es speziell beim Programmieren oft einfach die von ChatGPT vorgeschlagene Lösung auch selbst zu prüfen, sodass man sich nicht blind darauf verlassen werden muss, was die KI sagt. Betrachtet man also ChatGPT mehr als Assistenten, statt als allwissende Quelle, kann man sehr viel Zeit sparen und dazulernen.

<https://www.youtube.com/embed/mQQOzg56KxQ?si=3eiytLSGCy7exSLR>

#### **i** Hinweis

Mit Hinblick auf diesen Kurs soll natürlich nicht der Eindruck erweckt werden, dass man ab jetzt bei Problemen auf sich allein gestellt ist. Natürlich ist es weiterhin möglich und angedacht, uns bei BioMath während des Kurszeitraums zu kontaktieren um (besonders hartnäckige) Probleme zu lösen. Gleichzeitig ist es aber wie bereits erwähnt wirklich auch eine Kernkompetenz beim Programmieren, dass man in der Lage ist, Probleme selbstständig zu lösen. Im Übrigen fühlt es sich auch gut an ein hartnäckiges Problem nach langer Suche endlich gelöst zu haben. Es liegt also an jedem selbst eine Balance zu finden zwischen eigenständigem Lösen und dem Einholen von Hilfe. Einerseits wird letztere nicht immer da sein, sodass man sich bestmöglich darauf vorbereiten sollte, andererseits soll man natürlich die Ressourcen nutzen, die einem zur Verfügung stehen.

## Übungen

Dies hier sind keine echten Übungen und man darf diesen Teil auch gerne überspringen. Der Vorschlag ist aber, dass man sich bei stackoverflow und bei ChatGPT anmeldet und es ein wenig ausprobiert. Dabei sei nochmals betont, dass hier keine Notwendigkeit vorliegt: Man kann diesen Kurs und auch das Programmieren im Allgemeinen auch ohne diese beiden Plattformen meistern. Es gibt auch ernstzunehmende Gründe gegen die Nutzung von ChatGPT und es kommt auch vor, dass man je nach Arbeitgeber gar kein ChatGPT im Beruf nutzen darf.

Wenn man sich all dessen aber bewusst ist und prinzipiell die Möglichkeit sowie Interesse besteht, dann ist es sicherlich eine wertvolle Erfahrung, zumindest mal mit diesen Plattformen zu experimentieren. Beide sind in der Basisversion kostenfrei.

#### 💡 Weitere Ressourcen

- Stackoverflow verwenden
- Stackoverflow Help: How do I ask a good question?
- ChatGPT vs. StackOverflow
- Python SCHNELL lernen mit ChatGPT (game changer)
- Cheat Sheet: Python for Data Science - Python Basics