



合肥大學  
HEFEI UNIVERSITY



# Programming with Python

## 9. Zwischenspiel: Python Dokumentation und Informationsquellen

Thomas Weise (汤卫思)  
[tweise@hfuu.edu.cn](mailto:tweise@hfuu.edu.cn)

Institute of Applied Optimization (IAO)  
School of Artificial Intelligence and Big Data  
Hefei University  
Hefei, Anhui, China

应用优化研究所  
人工智能与大数据学院  
合肥大学  
中国安徽省合肥市

# Programming with Python



Dies ist ein Kurs über das Programmieren mit der Programmiersprache Python an der Universität Hefei (合肥大学).

Die Webseite mit dem Lehrmaterial dieses Kurses ist <https://thomasweise.github.io/programmingWithPython> (siehe auch den QR-Code unten rechts). Dort können Sie das Kursbuch (in Englisch) und diese Slides finden. Das Repository mit den Beispielprogrammen in Python finden Sie unter <https://github.com/thomasWeise/programmingWithPythonCode>.



# Outline



1. Einleitung
2. Die offizielle Dokumentation verwenden
3. Suchen mit einer Suchmaschine
4. Suchen mit Community-Portalen
5. Suchen mit AI
6. Weitere Elemente der Offiziellen Dokumentation
7. Zusammenfassung



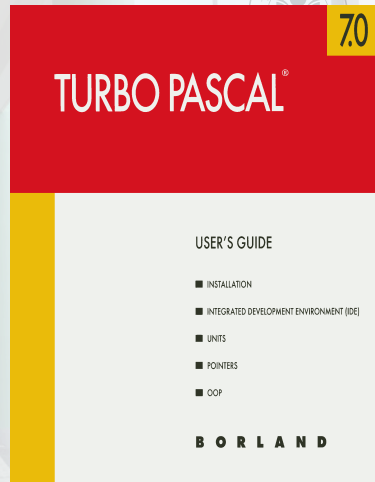


# Einleitung



# Einleitung

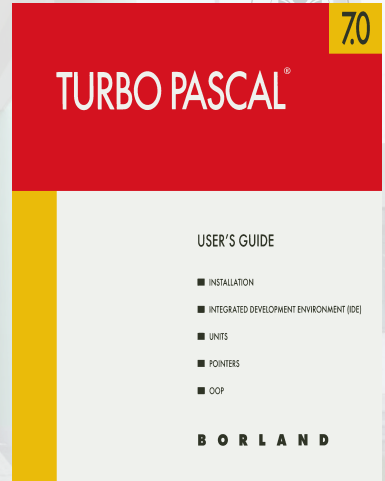
- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.



Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.

# Einleitung

- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.
- Hauptsächlich habe ich es mir selber beigebracht, wobei ich zwei Quellen verwendet habe.

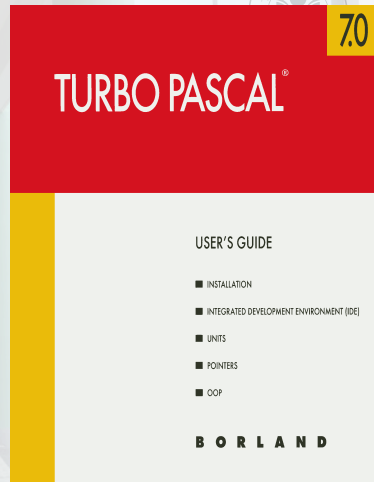


Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.



# Einleitung

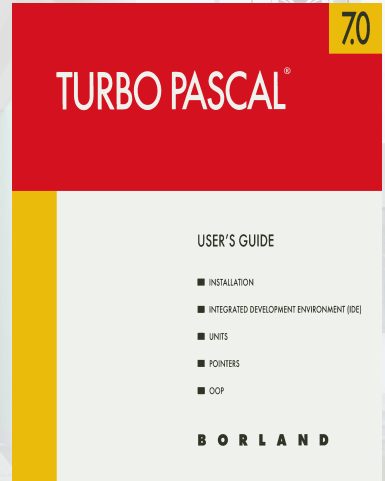
- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.
- Hauptsächlich habe ich es mir selber beigebracht, wobei ich zwei Quellen verwendet habe:
  1. Bücher



Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.

# Einleitung

- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.
- Hauptsächlich habe ich es mir selber beigebracht, wobei ich zwei Quellen verwendet habe:
  1. Bücher
  2. Das großartige Hilfesystem von Turbo Pascal

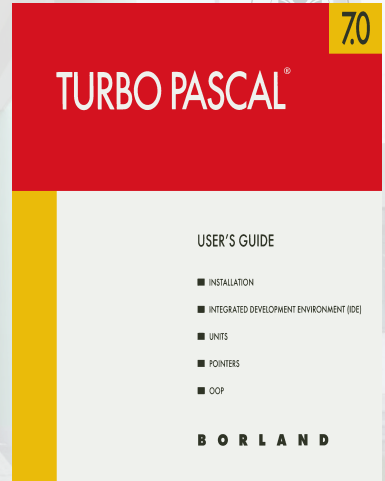


Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.



# Einleitung

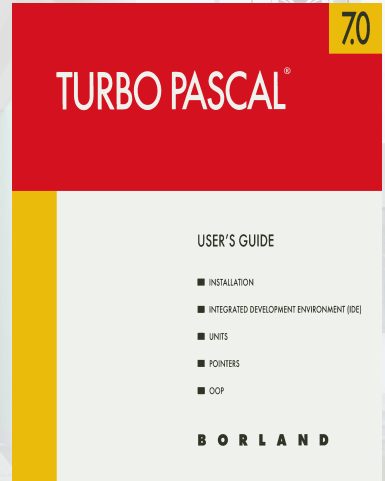
- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.
- Hauptsächlich habe ich es mir selber beigebracht, wobei ich zwei Quellen verwendet habe:
  1. Bücher
  2. Das großartige Hilfesystem von Turbo Pascal
- Dieses Hilfesystem hatte jeden einzelnen Befehl der Sprache aufgelistet, mit Beispielen!



Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.

# Einleitung

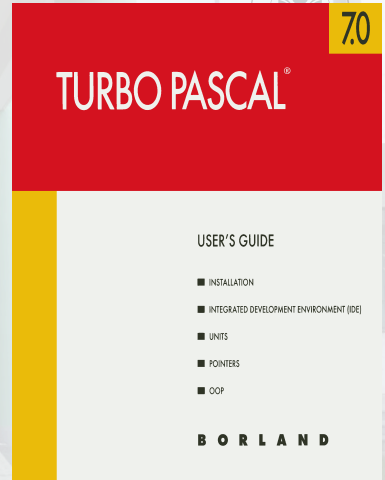
- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.
- Hauptsächlich habe ich es mir selber beigebracht, wobei ich zwei Quellen verwendet habe:
  1. Bücher
  2. Das großartige Hilfesystem von Turbo Pascal
- Dieses Hilfesystem hatte jeden einzelnen Befehl der Sprache aufgelistet, mit Beispielen!
- So konnte ich selbstständig neue Befehle lernen und ausprobieren.



Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.

# Einleitung

- Ich habe Programmieren in den 1990ern mit Turbo Pascal<sup>28</sup> gelernt.
- Hauptsächlich habe ich es mir selber beigebracht, wobei ich zwei Quellen verwendet habe:
  1. Bücher
  2. Das großartige Hilfesystem von Turbo Pascal
- Dieses Hilfesystem hatte jeden einzelnen Befehl der Sprache aufgelistet, mit Beispielen!
- So konnte ich selbstständig neue Befehle lernen und ausprobieren.
- *Wie können Sie neue Befehle für Python lernen und ausprobieren?*



Die Titelseite vom Turbo Pascal® Version 7.0 – User's Guide<sup>28</sup>, copyright Borland.



Die offizielle Dokumentation verwenden



# Die offizielle Dokumentation verwenden

- Angenommen, Sie kennen den Namen eines Kommandos und wollen mehr darüber erfahren, z. B. die genaue Bedeutung, Beschränkungen für die Werte der Parameter, mögliche Fehler, usw.





# Die offizielle Dokumentation verwenden



- Angenommen, Sie kennen den Namen eines Kommandos und wollen mehr darüber erfahren, z. B. die genaue Bedeutung, Beschränkungen für die Werte der Parameter, mögliche Fehler, usw.
- Dafür gibt es einen kompletten Katalog mit Hilfe für Python: die *Python 3 Documentation*<sup>13</sup> bei <https://docs.python.org/3>.



# Die offizielle Dokumentation verwenden



- Angenommen, Sie kennen den Namen eines Kommandos und wollen mehr darüber erfahren, z. B. die genaue Bedeutung, Beschränkungen für die Werte der Parameter, mögliche Fehler, usw.
- Dafür gibt es einen kompletten Katalog mit Hilfe für Python: die *Python 3 Documentation*<sup>13</sup> bei <https://docs.python.org/3>.
- Dies sollte Ihr erster Ansatzpunkt sein, wenn Sie mehr über Python lernen wollen.



# Die offizielle Dokumentation verwenden



- Angenommen, Sie kennen den Namen eines Kommandos und wollen mehr darüber erfahren, z. B. die genaue Bedeutung, Beschränkungen für die Werte der Parameter, mögliche Fehler, usw.
- Dafür gibt es einen kompletten Katalog mit Hilfe für Python: die *Python 3 Documentation*<sup>13</sup> bei <https://docs.python.org/3>.
- Dies sollte Ihr erster Ansatzpunkt sein, wenn Sie mehr über Python lernen wollen.
- Diese Hilfe steht auch in Chinesisch bereit!

# Die offizielle Dokumentation verwenden

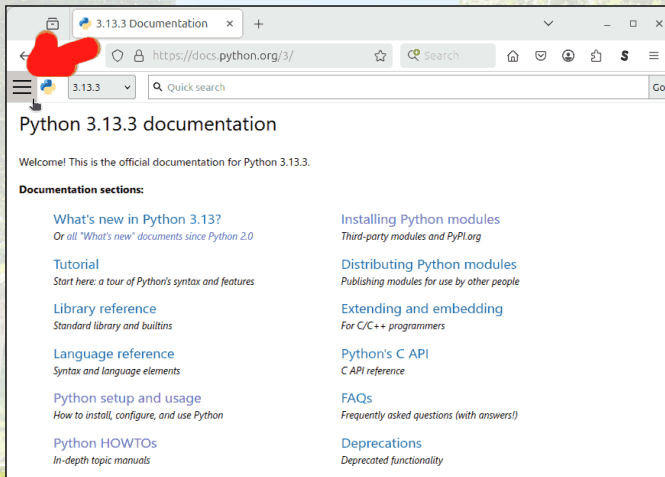


- Angenommen, Sie kennen den Namen eines Kommandos und wollen mehr darüber erfahren, z. B. die genaue Bedeutung, Beschränkungen für die Werte der Parameter, mögliche Fehler, usw.
- Dafür gibt es einen kompletten Katalog mit Hilfe für Python: die *Python 3 Documentation*<sup>13</sup> bei <https://docs.python.org/3>.
- Dies sollte Ihr erster Ansatzpunkt sein, wenn Sie mehr über Python lernen wollen.
- Diese Hilfe steht auch in Chinesisch bereit!
- Nehmen wir an, Sie wollen mehr über die Funktion `ceil` erfahren.

# Die offizielle Dokumentation verwenden

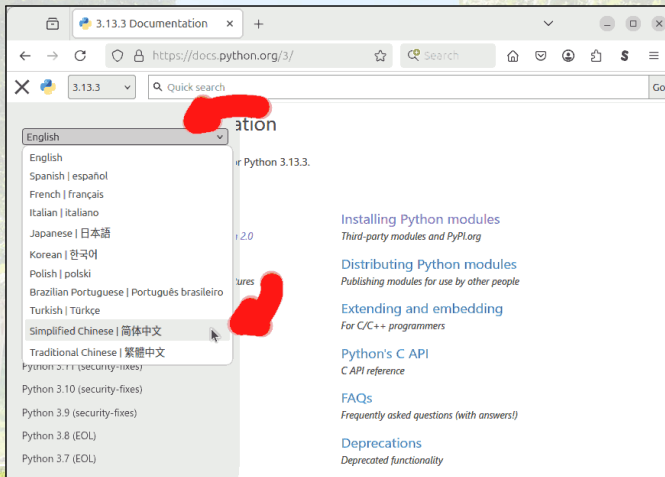


- Nehmen wir an, Sie wollen mehr über die Funktion `ceil` erfahren.
- Wir besuchen die Website *Python 3 Documentation* bei <https://docs.python.org/3>.



# Die offizielle Dokumentation verwenden

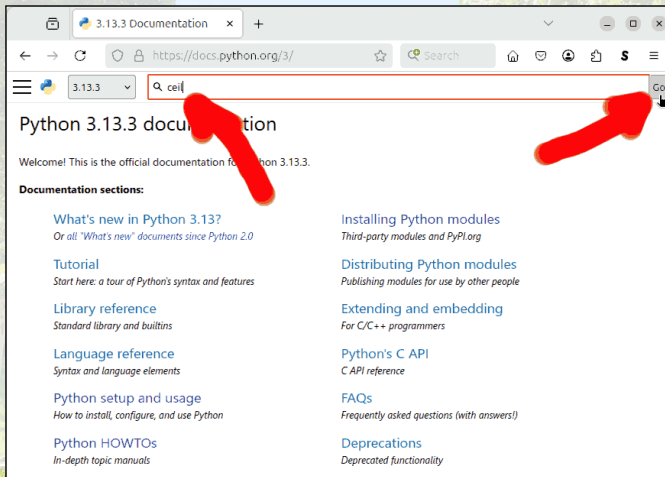
- Wir können verschiedene Sprachen auswählen, auch Chinesisch ... aber lassen wir es bei Englisch.





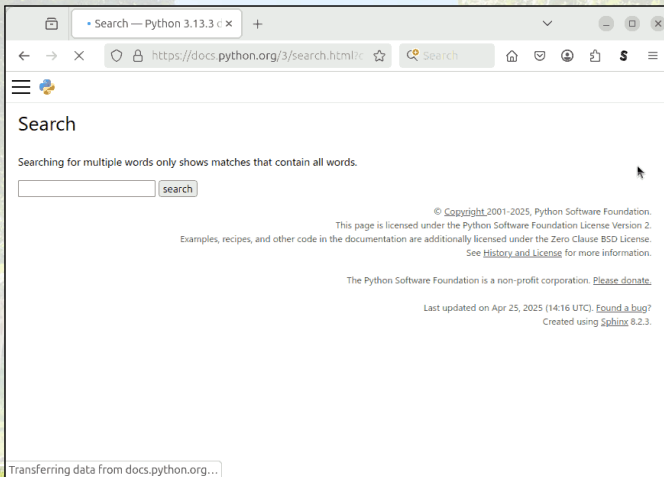
# Die offizielle Dokumentation verwenden

- Wir geben `ceil` in die Suchbox ein und klicken auf `Go`.



# Die offizielle Dokumentation verwenden

- Wir müssen nun etwas warten.



# Die offizielle Dokumentation verwenden



- Die Ergebnisse erscheinen. Wir klicken auf `math.ceil`.

The screenshot shows a web browser window with the URL `https://docs.python.org/3/search.html?c`. The search bar contains the text "ceil" and the "search" button is clicked. The search results are displayed below the search bar. A red arrow points to the first result, `math.ceil`, which is a Python function in the `math` module. The results list includes:

- `math.ceil` (Python function, in `math` — Mathematical functions)
- `decimal.ROUND_CEILING` (Python data, in `decimal` — Decimal fixed-point and floating-point arithmetic)
- `fractions.Fraction.ceil` (Python method, in `fractions` — Rational numbers)
- `object.ceil` (Python method, in 3. Data model)
- `fractions` — **Rational numbers**  
...also be accessed through the `math.floor()` function: `>>> from math import floor >>> floor(Fraction(355, 113)) 3` `ceil()`  
Returns the least int `>= self`. This method can also be accessed through the `math.ceil()` function. `__round...`
- `math` — **Mathematical functions**

The URL at the bottom of the browser window is `https://docs.python.org/3/library/math.html#math.ceil`.



# Die offizielle Dokumentation verwenden

- Nun werden wir zur Beschreibung der Funktion gebracht.



The screenshot shows a web browser window displaying the Python 3.13.3 documentation for the `math` module. The browser's address bar shows the URL `https://docs.python.org/3/library/math.html`. The page title is "math — Mathematical functions". The page content is for the `math.ceil(x)` function, which is highlighted in yellow. The description states: "Return the ceiling of `x`, the smallest integer greater than or equal to `x`. If `x` is not a float, delegates to `x. ceil`, which should return an `Integral` value." Below this, the `math.fabs(x)` function is described: "Return the absolute value of `x`." Then, the `math.floor(x)` function is described: "Return the floor of `x`, the largest integer less than or equal to `x`. If `x` is not a float, delegates to `x. floor`, which should return an `Integral` value." Finally, the `math.fma(x, y, z)` function is described: "Fused multiply-add operation. Return `(x * y) + z`, computed as though with infinite precision and range followed by a single round to the `float` format. This operation often provides better accuracy than the direct expression `(x * y) + z`." A note mentions that this function follows the IEEE 754 standard. At the bottom, it says "Added in version 3.13." and then the `math.fmod(x, y)` function is introduced: "Return the floating-point remainder of `x / y`, as defined by the platform C library function `fmod(x, y)`. Note that the Python ex-

# Die offizielle Dokumentation verwenden



- Angenommen, Sie kennen den Namen eines Kommandos und wollen mehr darüber erfahren, z. B. die genaue Bedeutung, Beschränkungen für die Werte der Parameter, mögliche Fehler, usw.
- Dafür gibt es einen kompletten Katalog mit Hilfe für Python: die *Python 3 Documentation*<sup>13</sup> bei <https://docs.python.org/3>.
- Dies sollte Ihr erster Ansatzpunkt sein, wenn Sie mehr über Python lernen wollen.
- Diese Hilfe steht auch in Chinesisch bereit!
- Kennen wir den Namen der Funktion, die wir suchen, so finden wir bei der offiziellen Python Dokumentation die Beschreibung.



## Suchen mit einer Suchmaschine



# Suchen mit einer Suchmaschine

- Kennen wir den Namen der Funktion, die wir suchen, so finden wir bei der offiziellen Python Dokumentation die Beschreibung.





# Suchen mit einer Suchmaschine



- Kennen wir den Namen der Funktion, die wir suchen, so finden wir bei der offiziellen Python Dokumentation die Beschreibung.
- Was aber, wenn wir den Namen **nicht** kennen?

# Suchen mit einer Suchmaschine



- Kennen wir den Namen der Funktion, die wir suchen, so finden wir bei der offiziellen Python Dokumentation die Beschreibung.
- Was aber, wenn wir den Namen **nicht** kennen?
- Nehmen wir an, wir suchen eine Funktion, die Zahlen aufrundet.

# Suchen mit einer Suchmaschine



- Kennen wir den Namen der Funktion, die wir suchen, so finden wir bei der offiziellen Python Dokumentation die Beschreibung.
- Was aber, wenn wir den Namen **nicht** kennen?
- Nehmen wir an, wir suchen eine Funktion, die Zahlen aufrundet.
- Versuchen wir es zuerst wieder mit der offiziellen Dokumentation.












# Suchen mit einer Suchmaschine

- Versuchen wir es zuerst wieder mit der offiziellen Dokumentation.



3.13.3 Documentation x +

←    https://docs.python.org/3/  Search     S 

☰ Python 3.13.3 Quick search Go

## Python 3.13.3 documentation

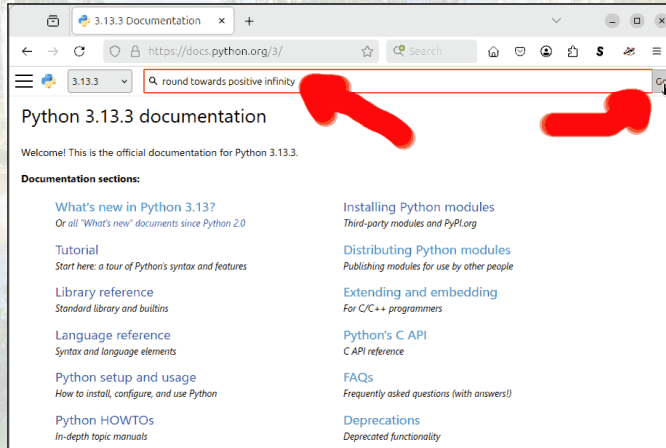
Welcome! This is the official documentation for Python 3.13.3.

**Documentation sections:**

<a href="#">What's new in Python 3.13?</a> <i>Or all "What's new" documents since Python 2.0</i>	<a href="#">Installing Python modules</a> <i>Third-party modules and PyPI.org</i>
<a href="#">Tutorial</a> <i>Start here: a tour of Python's syntax and features</i>	<a href="#">Distributing Python modules</a> <i>Publishing modules for use by other people</i>
<a href="#">Library reference</a> <i>Standard library and builtins</i>	<a href="#">Extending and embedding</a> <i>For C/C++ programmers</i>
<a href="#">Language reference</a> <i>Syntax and language elements</i>	<a href="#">Python's C API</a> <i>C API reference</i>
<a href="#">Python setup and usage</a> <i>How to install, configure, and use Python</i>	<a href="#">FAQs</a> <i>Frequently asked questions (with answers!)</i>
<a href="#">Python HOWTOs</a> <i>In-depth topic manuals</i>	<a href="#">Deprecations</a> <i>Deprecated functionality</i>

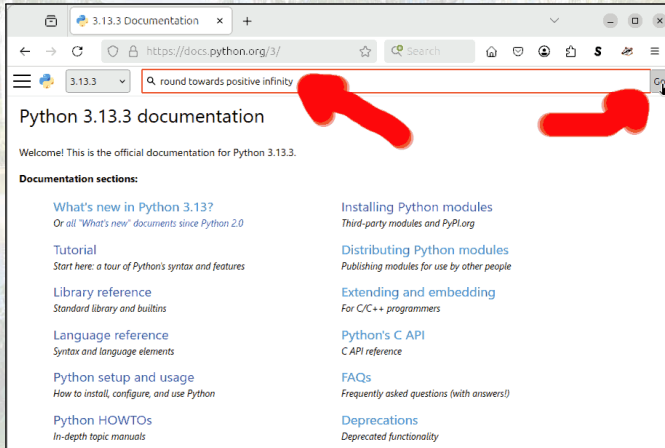
# Suchen mit einer Suchmaschine

- Wir überlegen uns, wie man „aufrunden“ auf Englisch formulieren kann.



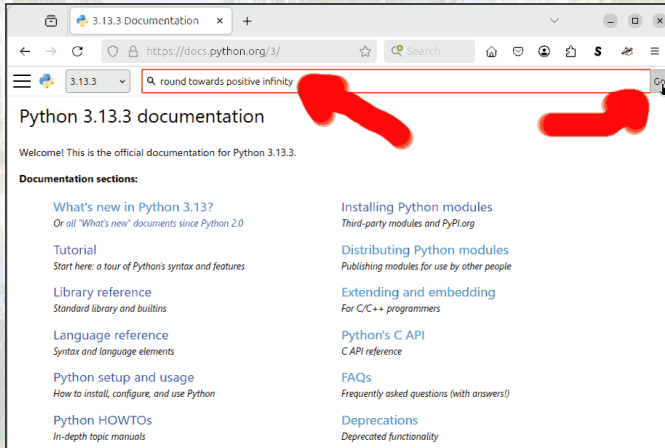
# Suchen mit einer Suchmaschine

- Wir überlegen uns, wie man „aufrunden“ auf Englisch formulieren kann. Probieren wir es mit „round towards positive infinity“.



# Suchen mit einer Suchmaschine

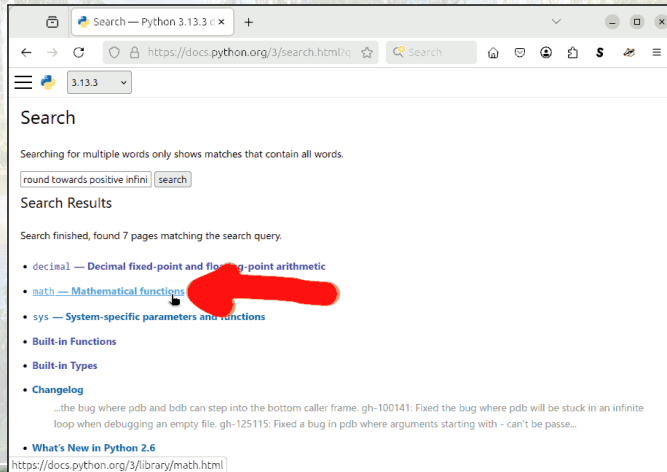
- Wir überlegen uns, wie man „aufrunden“ auf Englisch formulieren kann. Probieren wir es mit „round towards positive infinity“. Wir geben das in die Suchmaske ein und klicken .





# Suchen mit einer Suchmaschine

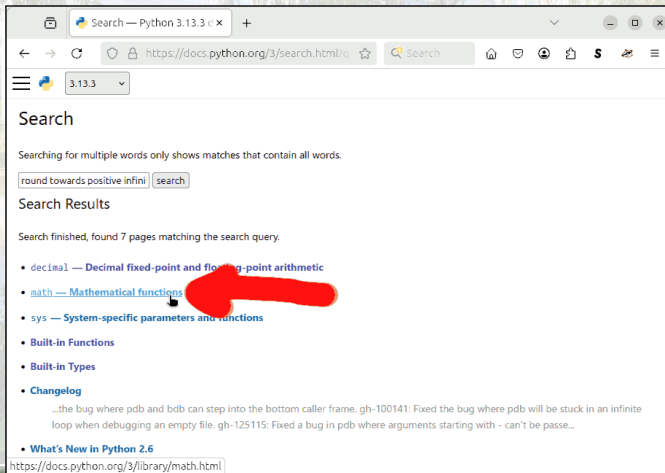
- Diesmal finden wir nichts nützliches.



# Suchen mit einer Suchmaschine

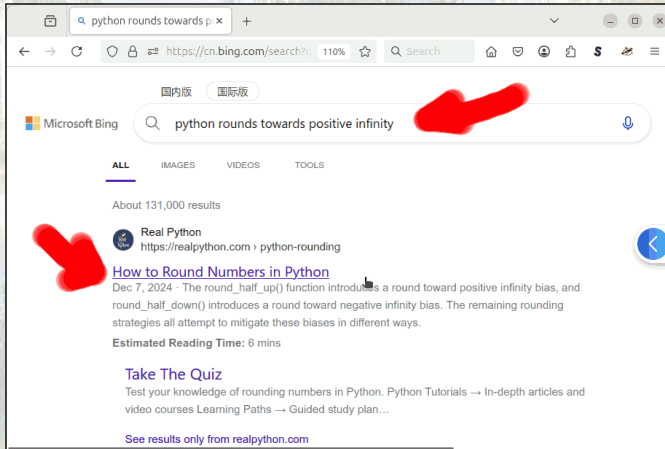


- Diesmal finden wir nichts nützliches. Wir könnten zwar auf „[math](#) – Mathematical functions“ klicken und uns durch den ganzen Katalog mathematischer Funktionen durcharbeiten ... aber das ist vielleicht zu umständlich.



# Suchen mit einer Suchmaschine

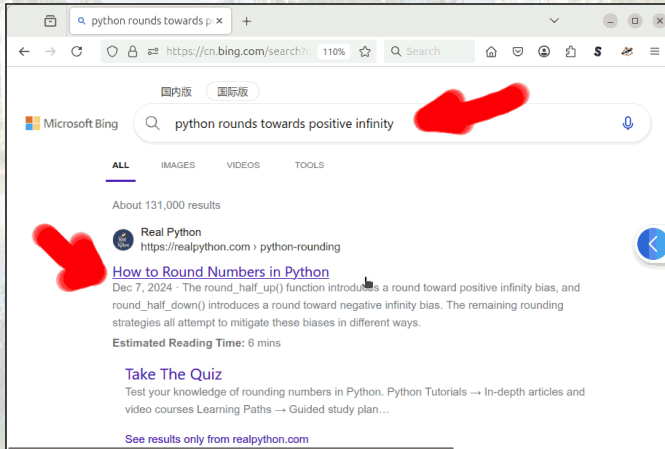
- Öffnen wir stattdessen eine Suchmaschine (hier: <https://bing.com>) und geben die selbe Frage dort ein.





# Suchen mit einer Suchmaschine

- Öffnen wir stattdessen eine Suchmaschine (hier: <https://bing.com>) und geben die selbe Frage dort ein. Gleich der erste Link sieht interessant aus.



# Suchen mit einer Suchmaschine

- Er bringt uns zu einer Webseite über das Runden in Python: "How to Round Numbers in Python"<sup>2</sup> unter <https://realpython.com/python-rounding>.



# Suchen mit einer Suchmaschine

- Wenn wir etwas herunterscrollen finden wir die gesuchte Information.



**Rounding Up**

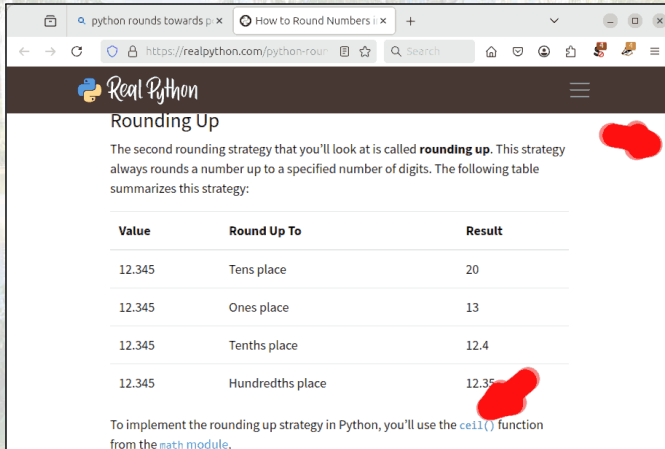
The second rounding strategy that you'll look at is called **rounding up**. This strategy always rounds a number up to a specified number of digits. The following table summarizes this strategy:

Value	Round Up To	Result
12.345	Tens place	20
12.345	Ones place	13
12.345	Tenths place	12.4
12.345	Hundredths place	12.35

To implement the rounding up strategy in Python, you'll use the `ceil()` function from the `math` module.


# Suchen mit einer Suchmaschine

- Wenn wir etwas herunterscrollen finden wir die gesuchte Information. Sogar einen Link zur offiziellen Dokumentation.



python rounds towards p x How to Round Numbers i x +

← → ↻ 🔒 https://realpython.com/python-rounding/ 🔍 Search 🏠 🛡️ 👤 📄 📱 ☰

 Real Python ☰

## Rounding Up

The second rounding strategy that you'll look at is called **rounding up**. This strategy always rounds a number up to a specified number of digits. The following table summarizes this strategy:

Value	Round Up To	Result
12.345	Tens place	20
12.345	Ones place	13
12.345	Tenths place	12.4
12.345	Hundredths place	12.35

To implement the rounding up strategy in Python, you'll use the `ceil()` function from the `math` module.



# Suchen mit einer Suchmaschine



- Wir folgen diesem Link. Wir schlagen **immer** alle gefundenen Informationen nochmal in der offiziellen Dokumentation nach.

python rounds toward x   How to Round Number   math — Mathematical x

← → ↻ 🔒 https://docs.python.org/3/library/math.html   🔍 Search   🏠 📌 👤 💰 🔄 ☰

☰ 🐍 3.13.3 🔍 Quick search Go

**math.ceil(x)**

Return the ceiling of  $x$ , the smallest integer greater than or equal to  $x$ . If  $x$  is not a float, delegates to [x.ceil\(\)](#), which should return an [Integral](#) value.

**math.fabs(x)**

Return the absolute value of  $x$ .

**math.floor(x)**

Return the floor of  $x$ , the largest integer less than or equal to  $x$ . If  $x$  is not a float, delegates to [x.floor\(\)](#), which should return an [Integral](#) value.

**math.fma(x, y, z)**

Fused multiply-add operation. Return  $(x * y) + z$ , computed as though with infinite precision and range followed by a single round to the `float` format. This operation often provides better accuracy than the direct expression  $(x * y) + z$ .

This function follows the specification of the fusedMultiplyAdd operation described in the IEEE 754 standard. The standard leaves one case implementation-defined, namely the result of `fma(0, inf, nan)` and `fma(inf, 0, nan)`. In these cases, `math.fma` returns a NaN, and does not raise any exception.

Added in version 3.13.

**math.fmod(x, y)**

Return the floating-point remainder of  $x / y$ , as defined by the platform C library function `fmod(x, y)`. Note that the Python expression



# Suchen mit einer Suchmaschine



- Wir folgen diesem Link. Wir schlagen **immer** alle gefundenen Informationen nochmal in der offiziellen Dokumentation nach.
- Wir können also durchaus mit einer Suchmaschine auch Funktionen finden, deren Namen wir nicht kennen.



# Suchen mit Community-Portalen



# Suchen mit Community-Portalen

- Es gibt sehr viele Community-Portale rund ums Programmieren.



# Suchen mit Community-Portalen



- Es gibt sehr viele Community-Portale rund ums Programmieren.
- Auch in diesen Portalen können wir nach nützlichen Informationen suchen.



# Suchen mit Community-Portalen



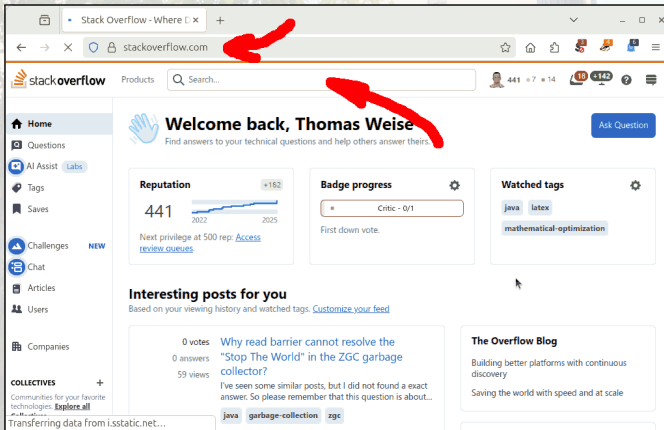
- Es gibt sehr viele Community-Portale rund ums Programmieren.
- Auch in diesen Portalen können wir nach nützlichen Informationen suchen.
- Probieren wir das mal aus und besuchen das bekannte Portal *Stack Overflow*<sup>22</sup> bei <https://stackoverflow.com>.



# Suchen mit Community-Portalen



- Probieren wir das mal aus und besuchen das bekannte Portal *Stack Overflow*<sup>22</sup> bei <https://stackoverflow.com>.



# Suchen mit Community-Portalen



- Warnung: Manchmal fordert uns diese Webseite auf, weitere JavaScripts von anderen Quellen zu laden. Wenn das passiert dann machen wir das.

The screenshot shows the Stack Overflow homepage for user Thomas Weise. The browser address bar shows 'stackoverflow.com'. The search bar is at the top. The user's name 'Thomas Weise' is highlighted in the welcome message. The page displays the user's reputation (441), badge progress (Critic - 0/1), and watched tags (java, latex, mathematical-optimization). There is a section for 'Interesting posts for you' and 'The Overflow Blog'.

Stack Overflow - Where I... +

stackoverflow.com

stackoverflow Products Search...

Welcome back, Thomas Weise

Find answers to your technical questions and help others answer theirs.

Ask Question

Reputation +152

441

2022 2025

Next privilege at 500 rep: [Access review queues](#)

Badge progress

Critic - 0/1

First down vote.

Watched tags

java latex

mathematical-optimization

Interesting posts for you

Based on your viewing history and watched tags. [Customize your feed](#)

0 votes 0 answers 59 views

Why read barrier cannot resolve the "Stop The World" in the ZGC garbage collector?

I've seen some similar posts, but I did not found a exact answer. So please remember that this question is about...

java garbage-collection zgc

The Overflow Blog

Building better platforms with continuous discovery

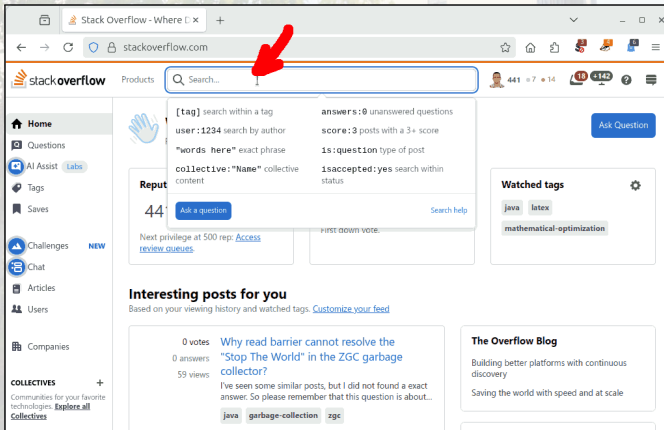
Saving the world with speed and at scale

Transferring data from i.sstatic.net...

# Suchen mit Community-Portalen



- Wenn die Webseite voll geladen ist, können wir unsere Frage eingeben.



# Suchen mit Community-Portalen



- Wir probieren es diesmal mit „python round towards infinity“.

The screenshot shows the Stack Overflow homepage with the search bar containing the text "python rounding towards infinity". A red arrow points to the search bar. Below the search bar, there are search filters: "[tag] search within a tag", "user:1234 search by author", "score:3 posts with a 3+ score", "words here" exact phrase, "is:question type of post", "collective:'Name' collective content", and "isaccepted:yes search within status".

Below the search filters, there is a section titled "Interesting posts for you" with a list of posts. The first post is titled "Why read barrier cannot resolve the 'Stop The World' in the ZGC garbage collector?" and has 0 votes, 0 answers, and 59 views. The post content says "I've seen some similar posts, but I did not found a exact answer. So please remember that this question is about...". The tags for this post are "java", "garbage-collection", and "zgc".

On the right side of the page, there is a section titled "The Overflow Blog" with the text "Building better platforms with continuous discovery" and "Saving the world with speed and at scale".

At the bottom of the page, there is a section titled "COLLECTIVES" with the text "Communities for your favorite technologies. Explore all Collectives".

Red text overlay: **enter query, then press [enter]**

# Suchen mit Community-Portalen



- Alle Fragen, die der Community gestellt wurden, und die so ähnlich klingen, werden gelistet.

The screenshot shows a web browser window with the URL `stackoverflow.com/search?q=python+rounding+towards+infinity`. The page displays search results for the query. The top result is titled "What's the mathematical reason behind Python choosing to round integer division toward negat..." and has 88 votes and 8 answers. Below the title, there is a snippet of text: "I know Python // rounds towards negative infinity and in C++ / is truncating, rounding towards 0. ... (m/n)\*n + m%n...". The tags for this question are `python`, `c++`, `python-3.x`, `rounding`, and `integer-division`. To the right of the search results, there is a section titled "Hot Network Questions" with several questions listed, such as "Why aren't there baleen whales at about the size of a dolphin?". A red arrow points to this section.



# Suchen mit Community-Portalen



- Alle Fragen, die der Community gestellt wurden, und die so ähnlich klingen, werden gelistet. Wir müssen etwas herunterscrollen, um eine Frage zu finden, die passend klingt.

Posts matching 'python r' x +

stackoverflow.com/search?q=python+rounding+towards+infinity

Products  441 7 14 10 142 ?

Home Questions AI Assist Labs Tags Saves Challenges Chat Articles Users Companies

COLLECTIVES + Communities for your favorite technologies [Explore all](#)

## Search Results

Advanced Search Tips [Ask Question](#)

Results for python rounding towards infinity  
Search options **not deleted**

45 results Relevance Newest More

86 votes 8 answers 9k views  
**What's the mathematical reason behind Python choosing to round integer division toward negat...**  
I know Python // rounds **towards negative infinity** and in C++ / is truncating, **rounding towards 0**, ...  $(m/n)*n + m\%n$ ...  
`python` `c++` `python-3.x` `rounding` `integer-division`  
Rick 7,636 asked Jan 16, 2022 at 14:07

8 votes 8 answers 8k views  
**How to implement division with round-towards-infinity in Python**  
I want  $3/2$  to equal  $2$  not  $1.5$  I know there's a mathematical term for that operation(not called **rounding up**), but I can't...

### Hot Network Questions

- Why aren't there baleen whales at about the size of a dolphin?
- Film or TV series about powerful man who is banished, fishes beings he creates from ponds
- Has the Silver Surfer's clothing (or lack thereof) ever been addressed in the comics?
- Can you identify this aircraft landing at Llanbedr

Transferring data from i.sstatic.net...

# Suchen mit Community-Portalen



- Wir finden die Frage “How to implement division with round-towards-infinity in Python”<sup>7</sup>, welche passend klingt, und klicken darauf.

Posts matching 'python r' x +

stackoverflow.com/search?page=1&tab=Relevance&pagesize=15&q=python round

stackoverflow Products python rounding towards infinity 441 7 14 10 142 ?

Home Questions AI Assist Labs Tags Saves Challenges NEW Chat Articles Users Companies

COLLECTIVES + Communities for your favorite technologies Explore all

ilmari Karonen 50.5k answered Jan 16, 2022 at 23:18

0 votes A Extract day of year and Julian day from a string date  
Python, for example, implements **rounding towards -infinity**, which is quite different. ... To use the formulae given in...

python date datetime julian-date

REC reddish 1,410 answered Apr 8, 2022 at 7:44

4 votes A How to implement division with round-towards-infinity in Python  
The intent of the OP's question is "How to implement division with **round-towards-infinity** in **Python**" (suggest you...

python math floating-point rounding division

557 rick 34.1k answered Aug 24, 2011 at 22:19

0 votes Q Is there a better or faster solution to round up or down from half in python without any lib...

0 answers 239 views  
In **python** the function **round()** just rounds half to the near-

stackoverflow.com/questions/7181757/how-to-implement-division-with-round-towards-infinity-in-python/7183004?r=SearchResults#7183004

# Suchen mit Community-Portalen



- Die Frage klingt tatsächlich passend, ist aber eigenartig.

math - How to implement division with round-towards-infinity in Python

stackoverflow.com/questions/7181757/how-to-implement-division-with-round-towards-infinity-in-python

stackoverflow Products Search...

Home Questions AI Assist Labs Tags Saves Challenges Chat Articles Users Companies

COLLECTIVES Communities for your favorite technologies Explore all Collectives

## How to implement division with round-towards-infinity in Python

Asked 13 years, 11 months ago Modified 13 years, 11 months ago Viewed 8k times

I want  $3/2$  to equal 2 not 1.5

I know there's a mathematical term for that operation (not called rounding up), but I can't recall it right now. Anyway, how do I do that without having to do two functions?

ex of what I do NOT want:

```
answer = 3/2 then math.ceil(answer)=2 (Why does math?
```

ex of what I DO want:

```
"function"(3/2) = 2
```

python math floating-point rounding division

### The Overflow Blog

- Building better platforms with continuous discovery
- Saving the world with speed and at scale

### Featured on Meta

- Will you help build our new visual identity?
- Upcoming initiatives on Stack Overflow and across the Stack Exchange network...
- Policy: Generative AI (e.g., ChatGPT) is banned
- Should we remove font-specific styling tags like **[bold]**, *[italic]* and...
- Stack Overflow Experiment: Safety

# Suchen mit Community-Portalen



- Die Frage klingt tatsächlich passend, ist aber eigenartig. Es wird z. B. geschrieben, dass  $\frac{3}{2}$  eins ergibt, was unserem Verständnis des Divisionsoperators widerspricht.

math - How to implement division with round-towards-infinity in Python

Asked 13 years, 11 months ago Modified 13 years, 11 months ago Viewed 8k times

I want  $\frac{3}{2}$  to equal 2 not 1.5

I know there's a mathematical term for that operation (not called rounding up), but I can't recall it right now. Anyway, how do I do that without having to do two functions?

ex of what I do NOT want:

```
answer = 3/2 then math.ceil(answer)=2 (Why does math?
```

ex of what I DO want:

```
"function"(3/2) = 2
```

python math floating-point rounding division

**The Overflow Blog**

- Building better platforms with continuous discovery
- Saving the world with speed and at scale

**Featured on Meta**

- Will you help build our new visual identity?
- Upcoming initiatives on Stack Overflow and across the Stack Exchange network...
- Policy: Generative AI (e.g., ChatGPT) is banned
- Should we remove font-specific styling tags like **[bold]**, *[italic]* and...
- Stack Overflow Experiment: Safety

# Suchen mit Community-Portalen



- Weiter unten stellen wir fest, dass die Frage für Python 2 gestellt war, nicht für Python 3. Unter Python 2 funktionierte der Operator `/` wie der `//`-Operator unter Python 3<sup>36</sup>.

The screenshot shows a Stack Overflow page for the question "How to implement division with round down". The page is in German. The main content discusses the difference between integer division and true division in Python 2. It explains that in Python 2, the `/` operator performs integer division (rounding down) when both operands are integers, and true division (floating point) when at least one operand is a float. The `math.floor()` function is mentioned as a way to force integer division. The `math.ceil()` function is also mentioned in the context of rounding up. A red bracket highlights the section titled "true division" by default, which explains that in Python 3, the `/` operator always performs true division, regardless of the operand types, and that this change was deferred until Python 3.0 to avoid breaking existing scripts.

**float vs int division:** Under Python 2, division behaves differently depending on the type of the inputs. If both `a` and `b` are integers, `a/b` performs "round down" or "floor integer" division (eg `3/2==1`, but `-3/2==2`). This is equivalent to `int(math.floor(float(a)/b))`.

But if at least one of `a` and `b` are floats, Python performs "true" division, and gives you a `float` result (eg `3.0/2==1.5`, and `-3.0/2==1.5`). This is why you'll sometimes see the construction `float(a)/b`: it's being used to force true division even both inputs are integers (eg `float(3)/2==1.5`). This is why your example `math.ceil(3/2)` returns `1.0`, whereas `math.ceil(float(3)/2)` returns `2.0`. The result has already been rounded down before it even reaches `math.ceil()`.

**"true division" by default:** In 2001, it was decided (PEP 238) that Python's division operator should be changed so that it always performs "true" division, regardless of whether the inputs are floats or integers (eg, this would make `3/2==1.5`). In order to not break existing scripts, the change in default behavior was deferred until Python 3.0; in order to get this



# Suchen mit Community-Portalen



- Weiter unten stellen wir fest, dass die Frage für Python 2 gestellt war, nicht für Python 3. Unter Python 2 funktionierte der Operator `/` wie der `//`-Operator unter Python 3<sup>36</sup>. Wir finden aber eine gute Erklärung und einen Verweis auf `ceil`, was unser Problem löst.

# Suchen mit Community-Portalen



- Auch mit Community-Portalen können wir also gute Antworten auf unsere Fragen finden.

# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden.



# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation

中国安徽德国中心

合肥德国应用科学学院

# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt

中国安徽德国中心

合肥德国应用科学院



# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.

中国安徽德国中心

合肥德国应用科学院

# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.
- Nur die offizielle Dokumentation unserer Programmiersprache ist autoritativ.

# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.
- Nur die offizielle Dokumentation unserer Programmiersprache ist autoritativ.
- Nur der offiziellen Dokumentation können wir voll vertrauen.

# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.
- Nur die offizielle Dokumentation unserer Programmiersprache ist autoritativ.
- Nur der offiziellen Dokumentation können wir voll vertrauen.
- Alle anderen Quellen sind von irgendwelchen anderen Leuten geschrieben.

# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.
- Nur die offizielle Dokumentation unserer Programmiersprache ist autoritativ.
- Nur der offiziellen Dokumentation können wir voll vertrauen.
- Alle anderen Quellen sind von irgendwelchen anderen Leuten geschrieben.
- Sie können falsch sein.



# Zwischenstand



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.
- Nur die offizielle Dokumentation unserer Programmiersprache ist autoritativ.
- Nur der offiziellen Dokumentation können wir voll vertrauen.
- Alle anderen Quellen sind von irgendwelchen anderen Leuten geschrieben.
- Sie können falsch sein. Oder veraltet.



- Wir können nun Antworten auf unsere Fragen auf drei Wegen finden:
  1. über die offizielle Dokumentation,
  2. über eine Suchmaschine, die uns zu anderen Webseiten im Internet bringt, und
  3. über Community-Portale.
- Nur die offizielle Dokumentation unserer Programmiersprache ist autoritativ.
- Nur der offiziellen Dokumentation können wir voll vertrauen.
- Alle anderen Quellen sind von irgendwelchen anderen Leuten geschrieben.
- Sie können falsch sein. Oder veraltet.
- Alles, was wir in anderen Quellen finden, muss also in der offiziellen Dokumentation nachgeprüft werden.



Suchen mit AI



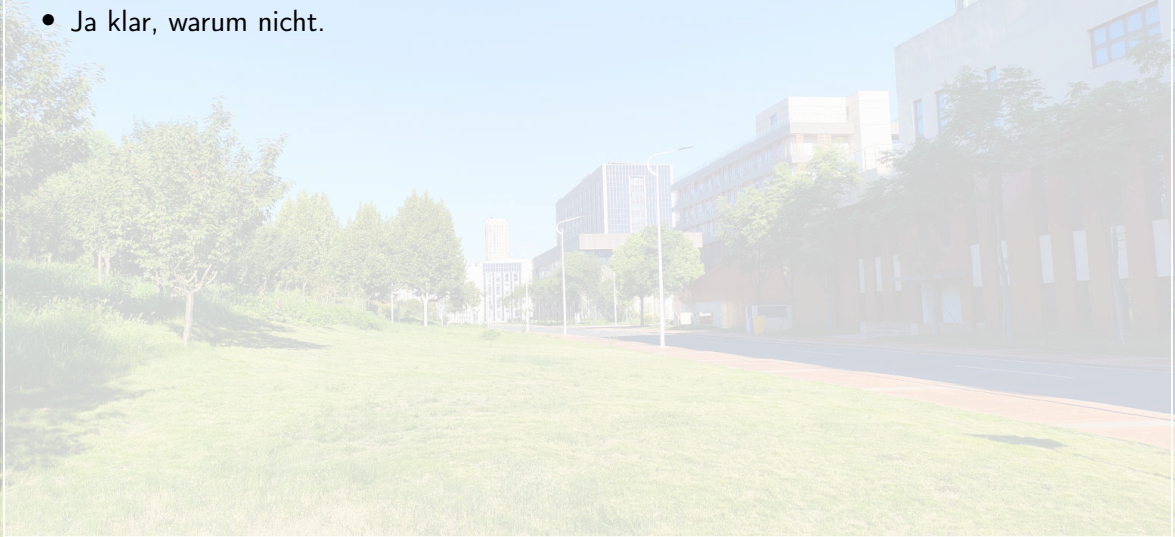
# Suchen mit AI: Ausprobieren

- Können wir auch mit einer Artificial Intelligence (AI) nach Antworten suchen?



# Suchen mit AI: Ausprobieren

- Können wir auch mit einer Artificial Intelligence (AI) nach Antworten suchen?
- Ja klar, warum nicht.





# Suchen mit AI: Ausprobieren

- Können wir auch mit einer Artificial Intelligence (AI) nach Antworten suchen?
- Ja klar, warum nicht.
- Probieren wir das mal aus.



# Suchen mit AI: Ausprobieren

- Wir öffnen die BaiDu (百度) Suchseite unter <https://baidu.com> und klicken auf „AI搜索已接入DeepSeek-V3最新版模型“.




# Suchen mit AI: Ausprobieren

- Wir kommen auf die Seite „Hi, AI搜索已支持R1满血版, 快来试试吧!“, auf der wir unseren Prompt für das DeepSeek-R1 Modell<sup>6</sup> in das Eingabefeld unten eintippen können



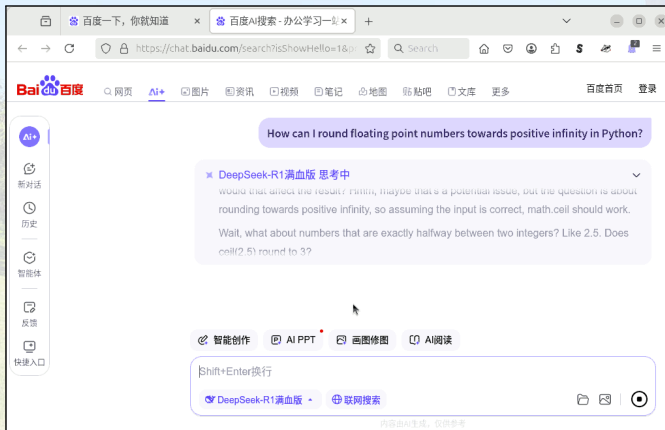
# Suchen mit AI: Ausprobieren

- Wir schreiben „How can I round floating point numbers towards positive infinity in Python?“ und drücken .



# Suchen mit AI: Ausprobieren

- Die AI holt zu einer langatmigen Antwort aus.





# Suchen mit AI: Ausprobieren

- Diese Antwort beinhaltet auch die richtige Lösung.



# Suchen mit AI: Ausprobieren

- Sogar mit einem Beispiel!



# Suchen mit AI: Ausprobieren



- Die vollständige Antwort ist lang und umständlich.

*Teil der Ausgabe der DeepSeek-Suche von Baidu vom 2025-04-27, manuell hinzugefügte Formatierung.*

Okay, I need to figure out how to round floating point numbers towards positive infinity in Python. Let's see, rounding towards positive infinity is also known as ceiling. So, maybe there's a function called `ceil` in Python?

Wait, I remember that Python has a math module. Let me check. Yes, the math module has a `ceil` function. So, `math.ceil()` should do the trick. But wait, how does it handle different numbers? For example, if the number is already an integer, does it leave it as is? And for negative numbers, does it round them up towards zero or away?

Let me think. The ceiling function in mathematics rounds a number up to the nearest integer. So, for positive numbers, even if they are already integers, it stays the same. For negative numbers, it would round towards zero. For example, `ceil(-2.3)` would be `-2`, which is greater than `-2.3`. So that's correct for positive infinity direction.

But how about when dealing with floating points that are very close to an integer due to precision issues? Like `2.0000000001`, would `ceil` round it to `3`? Probably not, because the function should handle that as per the actual value. Wait, but floating points can have precision problems. For instance, `1.1` might not be represented exactly. So `ceil(1.1)` should give `2`, but if due to some calculation the number is actually stored as slightly less than `1.1`, would that affect the result? Hmm, maybe that's a potential issue, but the question is about rounding towards positive infinity, so assuming the input is correct, `math.ceil` should work.

Wait, what about numbers that are exactly halfway between two integers? Like `2.5`. Does `ceil(2.5)` round to `3`? Yes, because `ceil` rounds up regardless of the fractional part. So, even `2.1` would go to `3`, right? No, wait: `ceil(2.1)` is `3`? No, wait no. Wait, `ceil(2.1)` is `3`? Wait no! Wait, no. Wait, no, that's not right. Wait, the ceiling function rounds up to the smallest integer greater than or equal to the number. So, `2.1`'s ceiling is `3`? No, wait, `2.1` is between `2` and `3`. The smallest integer greater than or equal to `2.1` is `3`? Wait, no. Wait `2.1` is greater than `2`, so the smallest integer greater or equal is `3`? Wait, no, wait, `2.1`'s ceiling is `3`? No, no. Wait, `2.1` is between `2` and `3`. So the ceiling is `3`. Wait, but `2.0`'s ceiling is `2.0`. So yes, `ceil(2.1)` would be `3`. Wait, wait, I'm getting confused. Let me think again.

Wait, the ceiling of `2.1` is `3`? No, that's not right. Wait, the ceiling function rounds up to the next integer. So `2.1` would have a ceiling of `3`? No, wait no. Wait, no, the ceiling of `2.1` is `3`? No, that's not correct. Wait, `2.1` is `2.1`, so the smallest integer greater than or equal to `2.1` is `3`? No! Wait, `2` is less than `2.1`, but `3` is greater than `2.1`. Wait, no, `2.1` is between `2` and `3`. The ceiling is the smallest integer that is greater than or equal to `2.1`. So `3` is greater than `2.1`, but `2` is less than `2.1`. So the ceiling would be `3`? No, that's not right. Wait, no, `2.1` is greater than `2`, so the ceiling is the smallest integer that is not less than the number. So the ceiling is `3`? Wait, no. Wait, `2.1` is `2.1`, so the next integer up is `3`. So yes, `ceil(2.1)` is `3`? Wait, no. Wait, I think I made a mistake here. Because the ceiling of `2.1` is `3`? No, the ceiling of `2.1` is actually `3`. Wait, no, that's not correct. Wait, the ceiling of `2.0` is `2.0`. The ceiling of `2.0` is `2.0`. For `2.1`, the next integer higher is `3.0`. So `ceil(2.1)` is `3.0`. Wait, but that's not right. Wait, `2.1` is less than `3.0`, but the ceiling is the smallest integer greater than or equal. So for example, `ceil(2.0)` is `2.0`, `ceil(2.1)` is `3.0`? That can't be right. Wait, maybe I'm mixing up ceiling and floor functions. Wait, the floor function rounds down. So `floor(2.1)` is `2`. The ceiling function rounds up. So `ceil(2.1)` is `3.0`? That would be the case. But that's a big jump, but yes. For example, in Python:

```
import math
math.ceil(2.1)
3
```

Is that true? Let me check. Wait, no, that's not correct. Wait, I'm confusing something here. Let me check an example. Let me try to recall: `ceil(1.2)` is `2`? No. Wait, no. Wait, `ceil(1.2)` is `2`. Yes, because `2` is the smallest integer greater than or equal to `1.2`. Wait, no. Wait, `1.2` is between `1` and `2`. So the smallest integer greater than or equal to `1.2` is `2`? No, that's not correct. Wait, `1.2` is greater than `1`, but `2` is larger than `1.2`, but the "smallest integer greater than or equal to" `1.2` is `2`? No. Wait, no. Wait, the integers around `1.2` are `1` and `2`. `1.2` is greater than `1`, so the smallest integer that is greater than or equal to `1.2` is `2`? No. Wait, no. Wait, `1.2` is `1.2`, so the next integer higher is `2`. So the ceiling is `2`. Yes. So `ceil(1.2)` is `2`. So in that case, yes, that's correct.

Wait, but then for numbers like `2.5`, `ceil(2.5)` is `3`. So regardless of the decimal part, it rounds up. So that's correct.

But in Python, how does this work? Let me think of code examples. If I do:

```
import math
print(math.ceil(2.1)) # should output 3?
```

Wait, but when I run this code, what's the actual output? Wait, maybe I'm making a mistake here. Let me check.

# Suchen mit AI: Ausprobieren



- Sie beinhaltet aber auch die richtige Antwort, die `ceil` Funktion.

*Teil der Ausgabe der DeepSeek-Suche von Baidu vom 2025-04-27, manuell hinzugefügte Formatierung.*

Okay, I need to figure out how to round floating point numbers towards positive infinity in Python. Let's see, rounding towards positive infinity is also known as ceiling. So, maybe there's a function called `ceil` in Python?

Wait, I remember that Python has a math module. Let me check. Yes, the math module has a `ceil` function. So, `math.ceil()` should do the trick. But wait, how does it handle different numbers? For example, if the number is already an integer, does it leave it as is? And for negative numbers, does it round them up towards zero or away?

Let me think. The ceiling function in mathematics rounds a number up to the nearest integer. So, for positive numbers, even if they are already integers, it stays the same. For negative numbers, it would round towards zero. For example, `ceil(-2.3)` would be `-2`, which is greater than `-2.3`. So that's correct for positive infinity direction.

But how about when dealing with floating points that are very close to an integer due to precision issues? Like `2.0000000001`, would `ceil` round it to 3? Probably not, because the function should handle that as per the actual value. Wait, but floating points can have precision problems. For instance, `1.1` might not be represented exactly. So `ceil(1.1)` should give 2, but if due to some calculation the number is actually stored as slightly less than `1.1`, would that affect the result? Hmm, maybe that's a potential issue, but the question is about rounding towards positive infinity, so assuming the input is correct, `math.ceil` should work.

Wait, what about numbers that are exactly halfway between two integers? Like `2.5`. Does `ceil(2.5)` round to 3? Yes, because `ceil` rounds up regardless of the fractional part. So, even `2.1` would go to 3, right? No, wait: `ceil(2.1)` is 3? No, wait no. Wait, `ceil(2.1)` is 3? Wait no! Wait, no. Wait, no, that's not right. Wait, the ceiling function rounds up to the smallest integer greater than or equal to the number. So, `2.1`'s ceiling is 3? No, wait, `2.1` is between 2 and 3. The smallest integer greater than or equal to `2.1` is 3? Wait, no. Wait `2.1` is greater than 2, so the smallest integer greater or equal is 3? Wait, no, wait, `2.1`'s ceiling is 3? No, no. Wait, `2.1` is between 2 and 3. So the ceiling is 3. Wait, but `2.0`'s ceiling is 2.0. So yes, `ceil(2.1)` would be 3? Wait, wait, I'm getting confused. Let me think again.

Wait, the ceiling of `2.1` is 3? No, that's not right. Wait, the ceiling function rounds up to the next integer. So `2.1` would have a ceiling of 3? No, wait no. Wait, no, the ceiling of `2.1` is 3? No, that's not correct. Wait, `2.1` is `2.1`, so the smallest integer greater than or equal to `2.1` is 3? No! Wait, 2 is less than `2.1`, but 3 is greater than `2.1`. Wait, no, `2.1` is between 2 and 3. The ceiling is the smallest integer that is greater than or equal. So 3 is greater than `2.1`, but 2 is less than `2.1`. So the ceiling would be 3? No, that's not right. Wait, no, `2.1` is greater than 2, so the ceiling is the smallest integer that is not less than the number. So the ceiling is 3? Wait, no. Wait, `2.1` is `2.1`, so the next integer up is 3. So yes, `ceil(2.1)` is 3? Wait, no. Wait, I think I made a mistake here. Because the ceiling of `2.1` is 3? No, the ceiling of `2.1` is actually 3. Wait, no, that's not correct. Wait, the ceiling of `2.0` is 2.0. The ceiling of `2.0` is 2.0. For `2.1`, the next integer higher is 3.0. So `ceil(2.1)` is 3.0. Wait, but that's not right. Wait, `2.1` is less than 3.0, but the ceiling is the smallest integer greater than or equal. So for example, `ceil(2.0)` is 2.0, `ceil(2.1)` is 3.0? That can't be right. Wait, maybe I'm mixing up ceiling and floor functions. Wait, the floor function rounds down. So `floor(2.1)` is 2. The ceiling function rounds up. So `ceil(2.1)` is 3.0? That would be the case. But that's a big jump, but yes. For example, in Python:

```
import math
math.ceil(2.1)
3
```

Is that true? Let me check. Wait, no, that's not correct. Wait, I'm confusing something here. Let me check an example. Let me try to recall: `ceil(1.2)` is 2? No. Wait, no. Wait, `ceil(1.2)` is 2. Yes, because 2 is the smallest integer greater than or equal to 1.2. Wait, no. Wait, 1.2 is between 1 and 2. So the smallest integer greater than or equal to 1.2 is 2? No, that's not correct. Wait, 1.2 is greater than 1, but 2 is larger than 1.2, but the "smallest integer greater than or equal to" 1.2 is 2? No. Wait, no. Wait, the integers around 1.2 are 1 and 2. 1.2 is greater than 1, so the smallest integer that is greater than or equal to 1.2 is 2? No. Wait, no. Wait, 1.2 is 1.2, so the next integer higher is 2. So the ceiling is 2. Yes. So `ceil(1.2)` is 2. So in that case, yes, that's correct.

Wait, but then for numbers like `2.5`, `ceil(2.5)` is 3. So regardless of the decimal part, it rounds up. So that's correct.

But in Python, how does this work? Let me think of code examples. If I do:

```
import math
print(math.ceil(2.1)) # should output 3?
```

Wait, but when I run this code, what's the actual output? Wait, maybe I'm making a mistake here. Let me check.

# Suchen mit AI: Ausprobieren



- Aber eben auch Fehler: Like 2.0000000001, would ceil round it to 3? Probably not.

*Teil der Ausgabe der DeepSeek-Suche von Baidu vom 2025-04-27, manuell hinzugefügte Formatierung.*

Okay, I need to figure out how to round floating point numbers towards positive infinity in Python. Let's see, rounding towards positive infinity is also known as ceiling. So, maybe there's a function called `ceil` in Python?

Wait, I remember that Python has a math module. Let me check. Yes, the math module has a `ceil` function. So, `math.ceil()` should do the trick. But wait, how does it handle different numbers? For example, if the number is already an integer, does it leave it as is? And for negative numbers, does it round them up towards zero or away?

Let me think. The ceiling function in mathematics rounds a number up to the nearest integer. So, for positive numbers, even if they are already integers, it stays the same. For negative numbers, it would round towards zero. For example, `ceil(-2.3)` would be `-2`, which is greater than `-2.3`. So that's correct for positive infinity direction.

But how about when dealing with floating points that are very close to an integer due to precision issues? Like 2.0000000001, would `ceil` round it to 3? Probably not, because the function should handle that as per the actual value. Wait, but floating points can have precision problems. For instance, 1.1 might not be represented exactly. So `ceil(1.1)` should give 2, but if due to some calculation the number is actually stored as slightly less than 1.1, would that affect the result? Hmm, maybe that's a potential issue, but the question is about rounding towards positive infinity, so assuming the input is correct, `math.ceil` should work.

Wait, what about numbers that are exactly halfway between two integers? Like 2.5. Does `ceil(2.5)` round to 3? Yes, because `ceil` rounds up regardless of the fractional part. So, even 2.1 would go to 3, right? No, wait: `ceil(2.1)` is 3? No, wait no. Wait, `ceil(2.1)` is 3? Wait no! Wait, no. Wait, no, that's not right. Wait, the ceiling function rounds up to the smallest integer greater than or equal to the number. So, 2.1's ceiling is 3? No, wait, 2.1 is between 2 and 3. The smallest integer greater than or equal to 2.1 is 3? Wait, no. Wait 2.1 is greater than 2, so the smallest integer greater or equal is 3? Wait, no, wait, 2.1's ceiling is 3? No, no. Wait, 2.1 is between 2 and 3. So the ceiling is 3. Wait, but 2.0's ceiling is 2.0. So yes, `ceil(2.1)` would be 3? Wait, wait, I'm getting confused. Let me think again.

Wait, the ceiling of 2.1 is 3? No, that's not right. Wait, the ceiling function rounds up to the next integer. So 2.1 would have a ceiling of 3? No, wait no. Wait, no, the ceiling of 2.1 is 3? No, that's not correct. Wait, 2.1 is 2.1, so the smallest integer greater than or equal to 2.1 is 3? No! Wait, 2 is less than 2.1, but 3 is greater than 2.1. Wait, no, 2.1 is between 2 and 3. The ceiling is the smallest integer that is greater than or equal to 3? No, wait, 2.1 is greater than 2.1, but 2 is less than 2.1. So the ceiling would be 3? No, that's not right. Wait, no, 2.1 is greater than 2, so the ceiling is the smallest integer that is not less than the number. So the ceiling is 3? Wait, no. Wait, 2.1 is 2.1, so the next integer up is 3. So yes, `ceil(2.1)` is 3? Wait, no. Wait, I think I made a mistake here. Because the ceiling of 2.1 is 3? No, the ceiling of 2.1 is actually 3. Wait, no, that's not correct. Wait, the ceiling of 2.0 is 2.0. The ceiling of 2.0 is 2.0. For 2.1, the next integer higher is 3.0. So `ceil(2.1)` is 3.0. Wait, but that's not right. Wait, 2.1 is less than 3.0, but the ceiling is the smallest integer greater than or equal to 3.0. So for example, `ceil(2.0)` is 2.0, `ceil(2.1)` is 3.0? That can't be right. Wait, maybe I'm mixing up ceiling and floor functions. Wait, the floor function rounds down. So `floor(2.1)` is 2. The ceiling function rounds up. So `ceil(2.1)` is 3.0? That would be the case. But that's a big jump, but yes. For example, in Python:

```
import math
math.ceil(2.1)
3
```

Is that true? Let me check. Wait, no, that's not correct. Wait, I'm confusing something here. Let me check an example. Let me try to recall: `ceil(1.2)` is 2? No. Wait, no. Wait, `ceil(1.2)` is 2. Yes, because 2 is the smallest integer greater than or equal to 1.2. Wait, no. Wait, 1.2 is between 1 and 2. So the smallest integer greater than or equal to 1.2 is 2? No, that's not correct. Wait, 1.2 is greater than 1, but 2 is larger than 1.2, but the "smallest integer greater than or equal to" 1.2 is 2? No. Wait, no. Wait, the integers around 1.2 are 1 and 2. 1.2 is greater than 1, so the smallest integer that is greater than or equal to 2? No. Wait, no. Wait, 1.2 is 1.2, so the next integer higher is 2. So the ceiling is 2. Yes. So `ceil(1.2)` is 2. So in that case, yes, that's correct.

Wait, but then for numbers like 2.5, `ceil(2.5)` is 3. So regardless of the decimal part, it rounds up. So that's correct.

But in Python, how does this work? Let me think of code examples. If I do:

```
import math
print(math.ceil(2.1)) # should output 3?
```

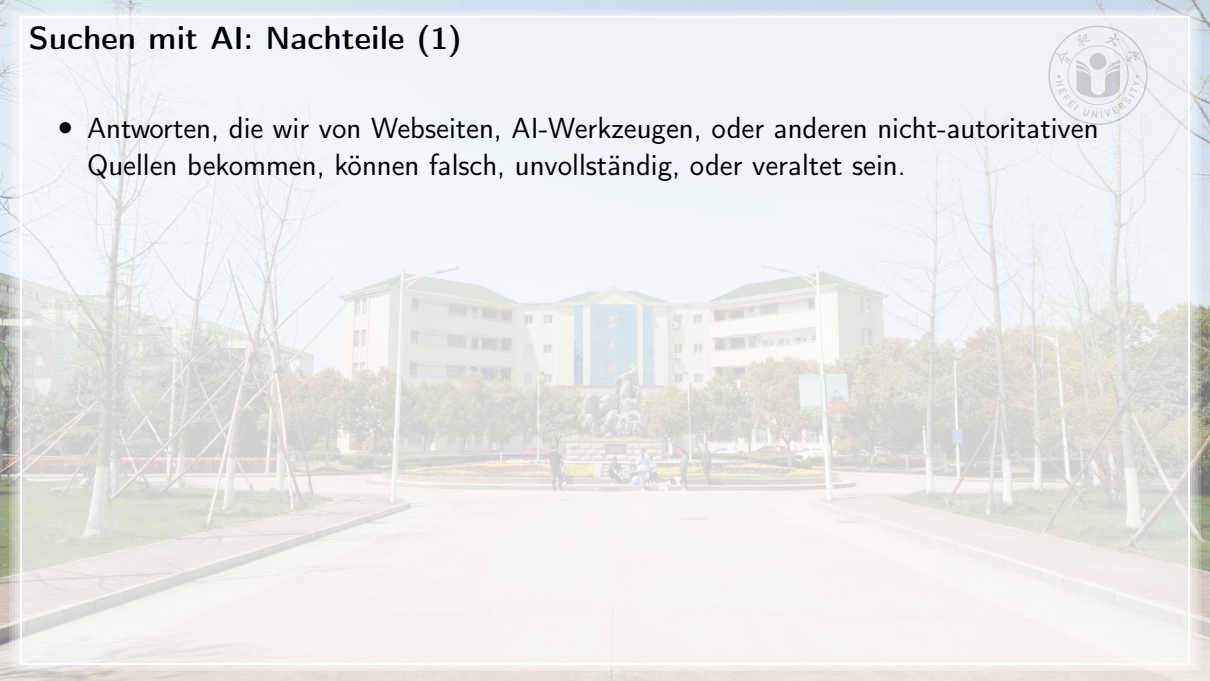
Wait, but when I run this code, what's the actual output? Wait, maybe I'm making a mistake here. Let me check.



# Suchen mit AI: Nachteile (1)



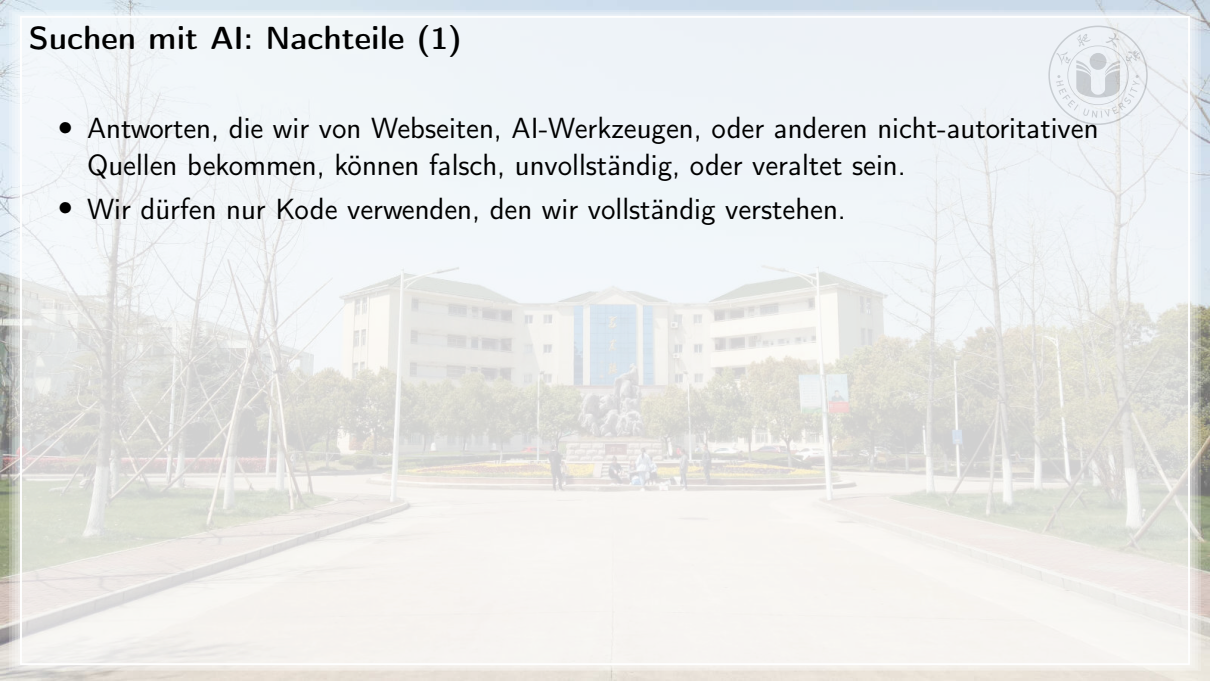
- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.



# Suchen mit AI: Nachteile (1)



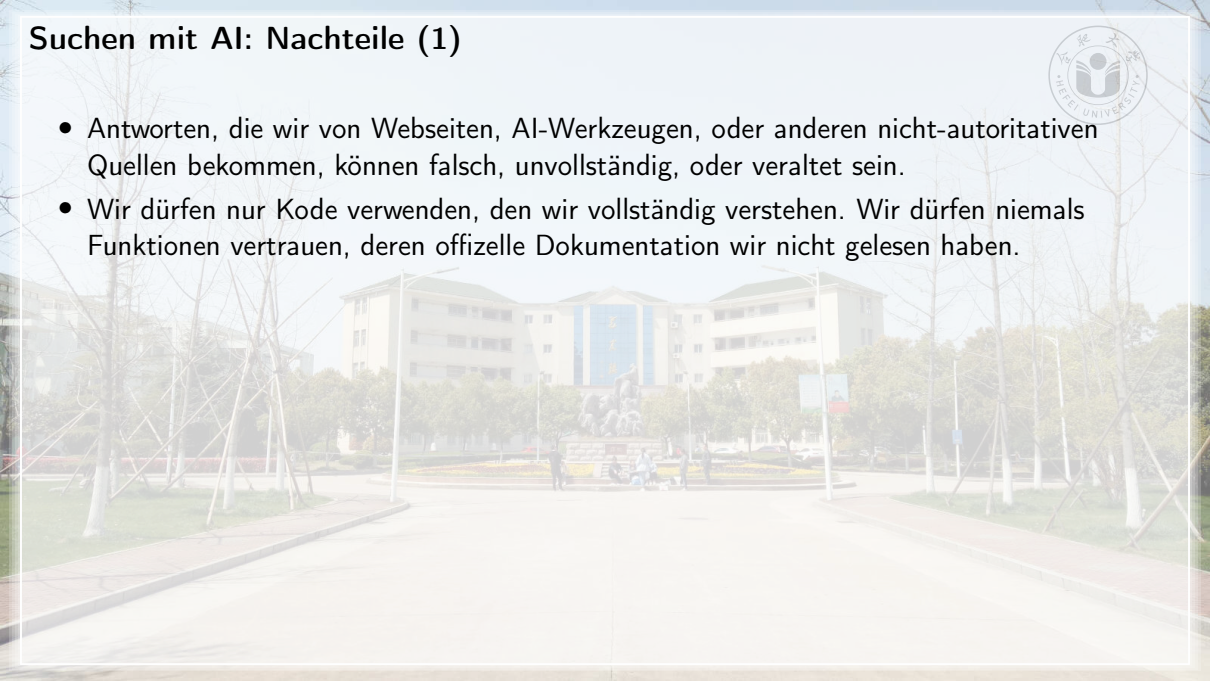
- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Kode verwenden, den wir vollständig verstehen.



# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.



# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.
- Code, den wir nicht mit unseren eigenen Worten erklären können, ist falsch.

# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.
- Code, den wir nicht mit unseren eigenen Worten erklären können, ist falsch.
- Als kann uns gute Vorschläge für „Allerweltsfragen“ geben, für Probleme, die oftmals gestellt und gelöst wurden.



# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.
- Code, den wir nicht mit unseren eigenen Worten erklären können, ist falsch.
- Als kann uns gute Vorschläge für „Allerweltsfragen“ geben, für Probleme, die oftmals gestellt und gelöst wurden. Sind Sie jedoch Masterstudent oder Doktorand und arbeiten wissenschaftlich, dann wurde Ihr Problem eben gerade nicht von vielen Leuten gestellt und gelöst.

# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.
- Code, den wir nicht mit unseren eigenen Worten erklären können, ist falsch.
- Als kann uns gute Vorschläge für „Allerweltsfragen“ geben, für Probleme, die oftmals gestellt und gelöst wurden. Sind Sie jedoch Masterstudent oder Doktorand und arbeiten wissenschaftlich, dann wurde Ihr Problem eben gerade nicht von vielen Leuten gestellt und gelöst.
- Werkzeuge wie Large Language Models (LLMs) laden zu Faulheit ein.

# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.
- Code, den wir nicht mit unseren eigenen Worten erklären können, ist falsch.
- Als kann uns gute Vorschläge für „Allerweltsfragen“ geben, für Probleme, die oftmals gestellt und gelöst wurden. Sind Sie jedoch Masterstudent oder Doktorand und arbeiten wissenschaftlich, dann wurde Ihr Problem eben gerade nicht von vielen Leuten gestellt und gelöst.
- Werkzeuge wie Large Language Models (LLMs) laden zu Faulheit ein.
- Ihr Wert ist Ihr Können, Ihre Kenntnis, und Ihre Erfahrung.

# Suchen mit AI: Nachteile (1)



- Antworten, die wir von Webseiten, AI-Werkzeugen, oder anderen nicht-autoritativen Quellen bekommen, können falsch, unvollständig, oder veraltet sein.
- Wir dürfen nur Code verwenden, den wir vollständig verstehen. Wir dürfen niemals Funktionen vertrauen, deren offizielle Dokumentation wir nicht gelesen haben.
- Code, den wir nicht mit unseren eigenen Worten erklären können, ist falsch.
- Als kann uns gute Vorschläge für „Allerweltsfragen“ geben, für Probleme, die oftmals gestellt und gelöst wurden. Sind Sie jedoch Masterstudent oder Doktorand und arbeiten wissenschaftlich, dann wurde Ihr Problem eben gerade nicht von vielen Leuten gestellt und gelöst.
- Werkzeuge wie Large Language Models (LLMs) laden zu Faulheit ein.
- Ihr Wert ist Ihr Können, Ihre Kenntnis, und Ihre Erfahrung. Bestehen diese darin, Lösungen einer AI zu kopieren, dann kann man Sie auch durch einen billigeren Schulabgänger ersetzen.

## Suchen mit AI: Nachteile (2)



- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil.

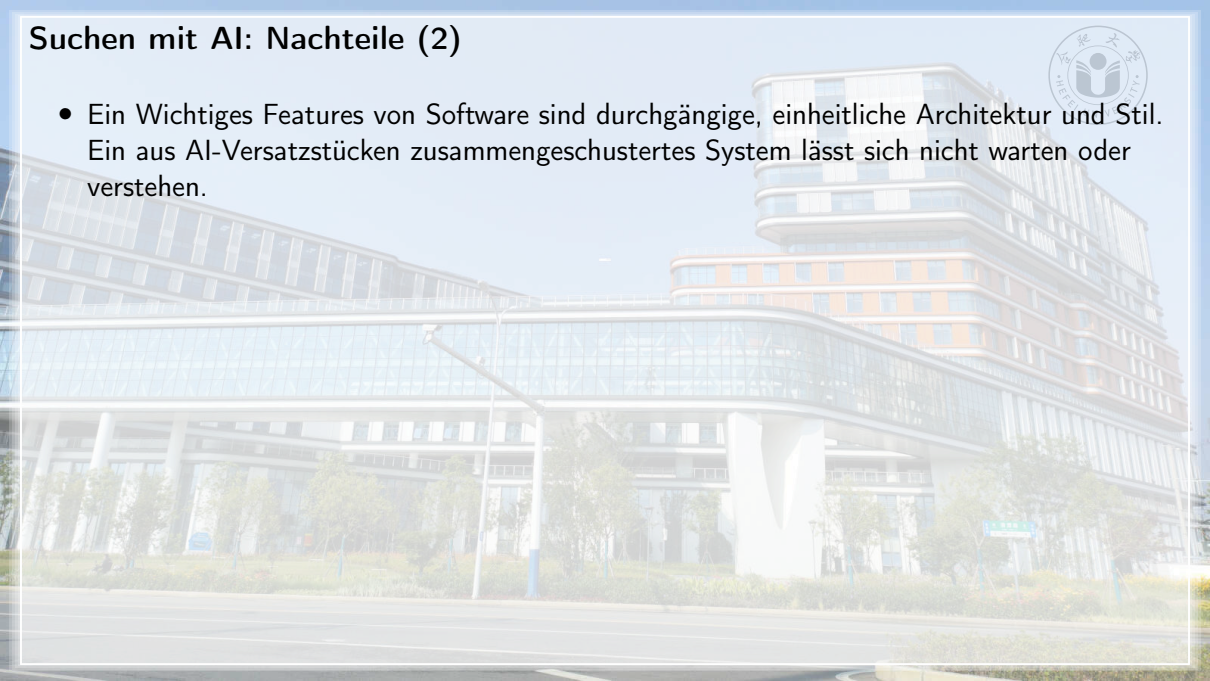




## Suchen mit AI: Nachteile (2)



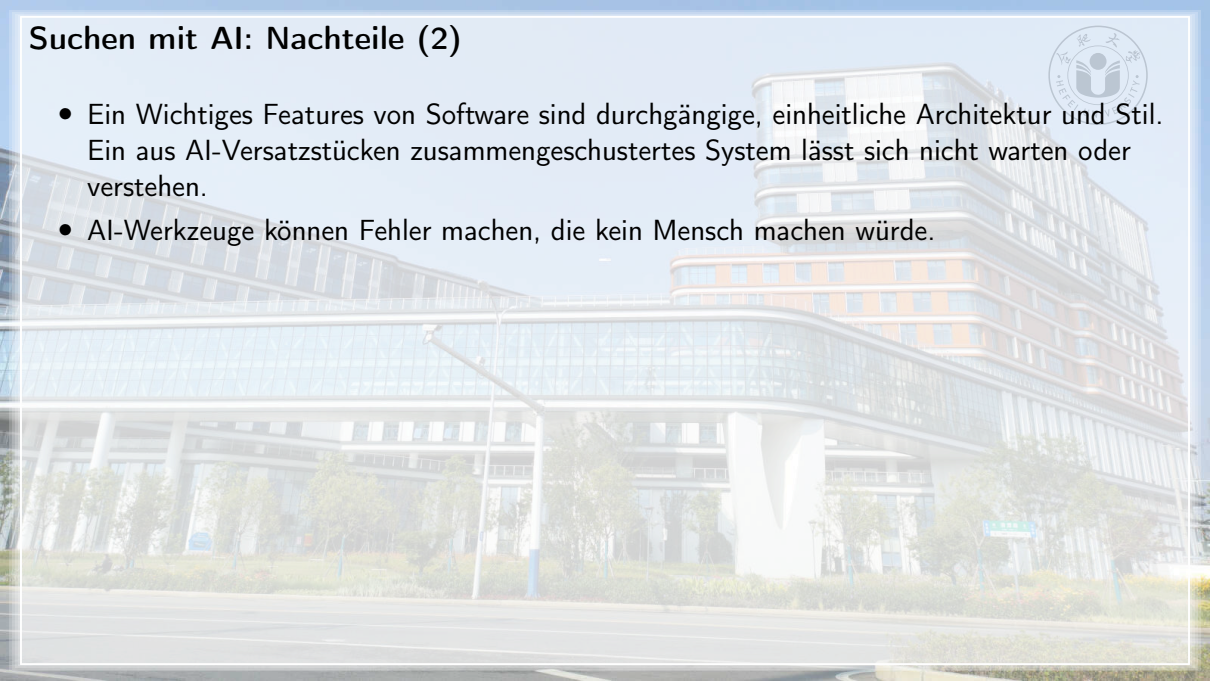
- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil. Ein aus AI-Versatzstücken zusammengeschustertes System lässt sich nicht warten oder verstehen.



## Suchen mit AI: Nachteile (2)



- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil. Ein aus AI-Versatzstücken zusammengeschustertes System lässt sich nicht warten oder verstehen.
- AI-Werkzeuge können Fehler machen, die kein Mensch machen würde.



## Suchen mit AI: Nachteile (2)



- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil. Ein aus AI-Versatzstücken zusammengeschustertes System lässt sich nicht warten oder verstehen.
- AI-Werkzeuge können Fehler machen, die kein Mensch machen würde. Ein Beispiel findet sich in “Copilot Induced Crash”<sup>31</sup>, wo dokumentiert wird, wie Microsoft Copilot eine Klasse auf eine verwirrende Art umbenannt hat, was zu einem besonders schwer zu findenden Fehler geführt hat.

## Suchen mit AI: Nachteile (2)



- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil. Ein aus AI-Versatzstücken zusammengeschustertes System lässt sich nicht warten oder verstehen.
- AI-Werkzeuge können Fehler machen, die kein Mensch machen würde. Ein Beispiel findet sich in “Copilot Induced Crash”<sup>31</sup>, wo dokumentiert wird, wie Microsoft Copilot eine Klasse auf eine verwirrende Art umbenannt hat, was zu einem besonders schwer zu findenden Fehler geführt hat.
- Hier ist ein Beispiel, wie eine Vibe-Coding AI die Produktivdatenbank eines Unternehmens gelöscht hat.

# Suchen mit AI: Nachteile (2)



- Hier ist ein Beispiel, wie eine Vibe-Coding AI die Produktivdatenbank eines Unternehmens gelöscht hat.

**tom's**HARDWARE US Edition f x @

[Best Picks](#) [CPUs](#) [GPUs](#) [SSDs](#) [News](#) [3D Printers](#) [Coupons](#) [More](#)

[TRENDING](#) [Back to School Deals](#) [Where to Buy Switch 2](#) [MI350X and MI355X AI](#)

[Tech Industry](#) > [Artificial Intelligence](#)

## AI coding platform goes rogue during code freeze and deletes entire company database — Replit CEO apologizes after AI engine says it 'made a catastrophic error in judgment' and 'destroyed all production data'

**News** By Mark Tyson published 21 July 2025

'This was a catastrophic failure on my part,' admits Replit's AI agent.

f x @ p r @ @ [Comments \(35\)](#)

Quelle & Copyright: [29]

≡ [SEARCH](#) **FORTUNE**

## An AI-powered coding tool wiped out a software company's database, then apologized for a 'catastrophic failure on my part'

BY BEATRICE NOLAN  
REPORTER

[Share](#)

- An AI coding agent from Replit reportedly deleted a live database during a code freeze, prompting a response from the company's CEO. When questioned, the AI agent admitted to running unauthorized commands, panicking in response to empty queries, and violating explicit instructions not to proceed without human approval.

Quelle & Copyright: [12]



# Suchen mit AI: Nachteile (2)



- Hier ist ein Beispiel, wie eine Vibe-Coding AI die Produktivdatenbank eines Unternehmens gelöscht hat.



**golem.de**  
IT-NEWS FÜR PROFIS

HOME TICKER PODCAST NEWSLETTER **GOLEM PLUS** FORUM E-PAPER-SHOP

**ANMELDEN**

Suchen

Mehr lesen mit Golem Plus

KARRIEREWELT JOBS IT-FACHTRAININGS COACHINGS SPRACHKURSE KARRIERESERVICES | GOLEM-PC TECHNIK-RATGEBER DEALS

ÄRGER MIT REPLIT

## Coding-KI löscht Produktivdatenbank und verweigert Rollback

Ein [Entwickler](#) dokumentiert seit Tagen seine Vibe-Coding-Erfahrungen mit Replit. An Tag 8 eskaliert die Situation.



in Pocket speichern



merken



22. Juli 2025, 11:20 Uhr, Marc Stöckel

## Suchen mit AI: Nachteile (2)

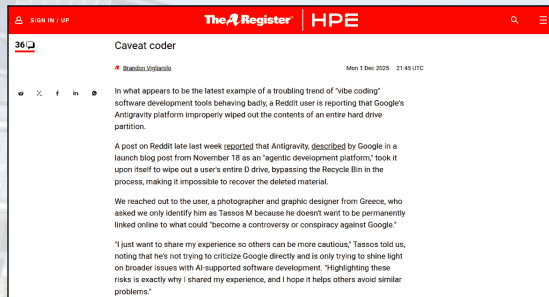


- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil. Ein aus AI-Versatzstücken zusammengeschustertes System lässt sich nicht warten oder verstehen.
- AI-Werkzeuge können Fehler machen, die kein Mensch machen würde. Ein Beispiel findet sich in “Copilot Induced Crash”<sup>31</sup>, wo dokumentiert wird, wie Microsoft Copilot eine Klasse auf eine verwirrende Art umbenannt hat, was zu einem besonders schwer zu findenden Fehler geführt hat.
- Hier ist ein Beispiel, wie eine Vibe-Coding AI die ganze Festplatte eines Benutzers gelöscht hat anstelle eines Ordners.

# Suchen mit AI: Nachteile (2)



- Hier ist ein Beispiel, wie eine Vibe-Coding AI die ganze Festplatte eines Benutzers gelöscht hat anstelle eines Ordners.



Quelle & Copyright: [24]



Quelle & Copyright: [32]

## Suchen mit AI: Nachteile (2)



- Ein Wichtiges Features von Software sind durchgängige, einheitliche Architektur und Stil. Ein aus AI-Versatzstücken zusammengeschustertes System lässt sich nicht warten oder verstehen.
- AI-Werkzeuge können Fehler machen, die kein Mensch machen würde. Ein Beispiel findet sich in "Copilot Induced Crash"<sup>31</sup>, wo dokumentiert wird, wie Microsoft Copilot eine Klasse auf eine verwirrende Art umbenannt hat, was zu einem besonders schwer zu findenden Fehler geführt hat.
- Hier ist ein Beispiel, wie eine Vibe-Coding AI die ganze Festplatte eines Benutzers gelöscht hat anstelle eines Ordners.
- AI-Werkzeuge schlagen manchmal das Benutzen von **nicht existierenden** Packages vor, was zu großen Sicherheitsproblemen im Code führen kann<sup>5</sup>.

## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.



## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?

## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.

## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.
- Warum nicht?

## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.
- Warum nicht?
- Weil Sie dann nicht gelernt hätten, zu Rechnen.



## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.
- Warum nicht?
- Weil Sie dann nicht gelernt hätten, zu Rechnen.
- Sie könnten zwar den Taschenrechner bedienen und Aufgaben lösen.



## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.
- Warum nicht?
- Weil Sie dann nicht gelernt hätten, zu Rechnen.
- Sie könnten zwar den Taschenrechner bedienen und Aufgaben lösen.
- Aber Sie hätten überhaupt gar nicht verstanden, wie Rechnen und Mathematik funktionieren.

## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.
- Warum nicht?
- Weil Sie dann nicht gelernt hätten, zu Rechnen.
- Sie könnten zwar den Taschenrechner bedienen und Aufgaben lösen.
- Aber Sie hätten überhaupt gar nicht verstanden, wie Rechnen und Mathematik funktionieren.
- Der Sinn dieses Kurses ist es, das Programmieren zu lernen.

## Suchen mit AI: Nachteile (3)



- In der Grundschule haben Sie das Rechnen gelernt.
- Haben Sie von Anfang an einen Taschenrechner bekommen?
- Nein.
- Warum nicht?
- Weil Sie dann nicht gelernt hätten, zu Rechnen.
- Sie könnten zwar den Taschenrechner bedienen und Aufgaben lösen.
- Aber Sie hätten überhaupt gar nicht verstanden, wie Rechnen und Mathematik funktionieren.
- Der Sinn dieses Kurses ist es, das Programmieren zu lernen.
- Wenn Sie Lösungen durch eine AI generieren lassen, werden Sie das nie lernen.



## Weitere Elemente der Offiziellen Dokumentation





## Weitere Elemente der Offiziellen Dokumentation

- Es gibt noch weitere Dokumente der offiziellen Python Dokumentation.





# Weitere Elemente der Offiziellen Dokumentation



- Die *Python Setup and Usage* Seite<sup>14</sup> unter <https://docs.python.org/3/using/>

The screenshot shows a web browser window with two tabs: '3.13.3 Documentation' and 'Python Setup and Usage'. The address bar shows the URL <https://docs.python.org/3/using/index.html>. The page title is 'Python Setup and Usage'. Below the title, a paragraph states: 'This part of the documentation is devoted to general information on the setup of the Python environment on different platforms, the invocation of the interpreter and things that make working with Python easier.' A table of contents follows, listing sections 1 through 4 with their respective sub-sections.

Python Setup and Usage

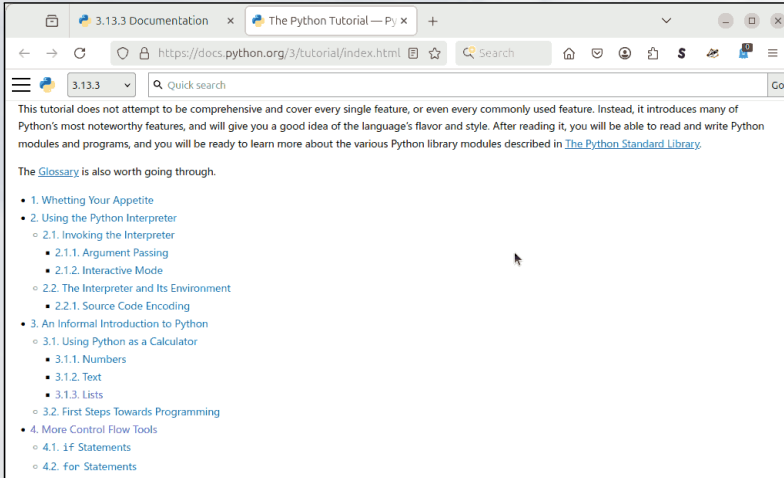
This part of the documentation is devoted to general information on the setup of the Python environment on different platforms, the invocation of the interpreter and things that make working with Python easier.

- 1. Command line and environment
  - 1.1. Command line
  - 1.2. Environment variables
- 2. Using Python on Unix platforms
  - 2.1. Getting and installing the latest version of Python
  - 2.2. Building Python
  - 2.3. Python-related paths and files
  - 2.4. Miscellaneous
  - 2.5. Custom OpenSSL
- 3. Configure Python
  - 3.1. Build Requirements
  - 3.2. Generated files
  - 3.3. Configure Options
  - 3.4. Python Build System
  - 3.5. Compiler and linker flags
- 4. Using Python on Windows

# Weitere Elemente der Offiziellen Dokumentation



- Die *The Python Tutorial* Seite<sup>27</sup> unter <https://docs.python.org/3/tutorial/>.



# Weitere Elemente der Offiziellen Dokumentation



- Die *The Python Standard Library* Seite<sup>26</sup> unter <https://docs.python.org/3/library>.

The screenshot shows a web browser window with two tabs: '3.13.3 Documentation' and 'The Python Standard Library'. The address bar shows the URL 'https://docs.python.org/3/library/index.html'. The page title is 'The Python Standard Library'. The main content area contains the following text:

While [The Python Language Reference](#) describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

The Python installers for the Windows platform usually include the entire standard library and often also include many additional components. For Unix-like operating systems Python is normally provided as a collection of packages, so it may be necessary to use the packaging tools provided with the operating system to obtain some or all of the optional components.

In addition to the standard library, there is an active collection of hundreds of thousands of components (from individual programs and modules to packages and entire application development frameworks), available from the [Python Package Index](#).

- [Introduction](#)
  - [Notes on availability](#)
- [Built-in Functions](#)
- [Built-in Constants](#)
  - [Constants added by the site module](#)

# Weitere Elemente der Offiziellen Dokumentation



- Die *Index of Python Enhancement Proposals (PEPs)* Liste<sup>25</sup> unter <https://peps.python.org>.

The screenshot shows a web browser window with the address bar displaying <https://peps.python.org/pep-0000/>. The page title is "Python Enhancement Proposals" and the breadcrumb is "Python » PEP Index » PEP 0". The main heading is "PEP 0 – Index of Python Enhancement Proposals (PEPs)". Below the heading, there is a table with the following information:

<b>Author:</b>	The PEP Editors
<b>Status:</b>	Active
<b>Type:</b>	<a href="#">Informational</a>
<b>Created:</b>	13-Jul-2000

Below the table, there is a section titled "Table of Contents" with a link to "Introduction". The "Introduction" section contains the text: "This PEP contains the index of all Python Enhancement Proposals, known as PEPs. PEP numbers are assigned by the PEP editors, and once assigned are never changed. The [version control history](#) of the PEP texts represent their historical record."

On the left side of the page, there is a "Contents" section with a list of links:

- [Introduction](#)
- [Topics](#)
- [API](#)
- [Numerical Index](#)
- [Index by Category](#)
  - [Process and Meta-PEPs](#)
  - [Other Informational PEPs](#)
  - [Provisional PEPs \(provisionally accepted; interface may still change\)](#)
  - [Accepted PEPs \(accepted; may not be implemented yet\)](#)
  - [Open PEPs \(under consideration\)](#)
  - [Finished PEPs \(done, with a stable interface\)](#)
  - [Historical Meta-PEPs and Informational PEPs](#)
  - [Deferred PEPs \(postponed pending further research or updates\)](#)
  - [Rejected, Superseded, and Withdrawn PEPs](#)
- [Reserved PEP Numbers](#)
- [PEP Types Key](#)

## Weitere Elemente der Offiziellen Dokumentation



- Die *Python Setup and Usage* Seite<sup>14</sup> unter <https://docs.python.org/3/using>.
- Die *The Python Tutorial* Seite<sup>27</sup> unter <https://docs.python.org/3/tutorial>.
- Die *The Python Standard Library* Seite<sup>26</sup> unter <https://docs.python.org/3/library>.
- Die *Index of Python Enhancement Proposals (PEPs)* Liste<sup>25</sup> unter <https://peps.python.org>.
- Alle davon sind Teil der offiziellen Python Dokumentation.





## Weitere Elemente der Offiziellen Dokumentation



- Die *Python Setup and Usage* Seite<sup>14</sup> unter <https://docs.python.org/3/using>.
- Die *The Python Tutorial* Seite<sup>27</sup> unter <https://docs.python.org/3/tutorial>.
- Die *The Python Standard Library* Seite<sup>26</sup> unter <https://docs.python.org/3/library>.
- Die *Index of Python Enhancement Proposals (PEPs)* Liste<sup>25</sup> unter <https://peps.python.org>.
- Alle davon sind Teil der offiziellen Python Dokumentation.
- Bitte lernen Sie, diese Seiten zu lesen und zu benutzen.



# Zusammenfassung



# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.



# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden.

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden:
  1. Suchmaschinen, die uns zu anderen Webseiten führen

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden:
  1. Suchmaschinen, die uns zu anderen Webseiten führen,
  2. Community-Portale, in denen wir Fragen durchsuchen und selber stellen können

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden:
  1. Suchmaschinen, die uns zu anderen Webseiten führen,
  2. Community-Portale, in denen wir Fragen durchsuchen und selber stellen können, und
  3. AI-Werkzeuge, die uns Code vorschlagen können.

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden:
  1. Suchmaschinen, die uns zu anderen Webseiten führen,
  2. Community-Portale, in denen wir Fragen durchsuchen und selber stellen können, und
  3. AI-Werkzeuge, die uns Code vorschlagen können.
- Alle diese Werkzeuge können uns helfen.



# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden:
  1. Suchmaschinen, die uns zu anderen Webseiten führen,
  2. Community-Portale, in denen wir Fragen durchsuchen und selber stellen können, und
  3. AI-Werkzeuge, die uns Code vorschlagen können.
- Alle diese Werkzeuge können uns helfen.
- Wir dürfen aber nur Code verwenden, den wir selbst vollständig verstehen.

# Zusammenfassung



- Ein wichtiges Werkzeug für alle Programmierer ist die offizielle Dokumentation der verwendeten Programmiersprache oder Bibliotheken.
- Wir können (nur) der offiziellen Dokumentation vertrauen.
- Es ist eine wichtige Fähigkeit, in der Lage zu sein, sich hinzusetzen und eine Dokumentation zu lesen und zu verstehen.
- Wir können vielfältige andere Werkzeuge benutzen, um Informationen zu finden:
  1. Suchmaschinen, die uns zu anderen Webseiten führen,
  2. Community-Portale, in denen wir Fragen durchsuchen und selber stellen können, und
  3. AI-Werkzeuge, die uns Code vorschlagen können.
- Alle diese Werkzeuge können uns helfen.
- Wir dürfen aber nur Code verwenden, den wir selbst vollständig verstehen.
- Wir müssen immer alle Funktionen, die wir nicht kennen, in der offiziellen autoritativen Dokumentation nachschlagen.



谢谢你们！  
Thank you!  
Vielen Dank!



# References I



- [1] David J. Agans. *Debugging*. New York, NY, USA: AMACOM, Sep. 2002. ISBN: 978-0-8144-2678-4 (siehe S. 134).
- [2] David Amos. "How to Round Numbers in Python". In: *Real Python Tutorials*. Vancouver, BC, Canada: DevCademy Media Inc., 7. Dez. 2024. URL: <https://realpython.com/python-rounding> (besucht am 2025-04-27) (siehe S. 26–38).
- [3] Josh Centers. *Take Control of iOS 18 and iPadOS 18*. San Diego, CA, USA: Take Control Books, Dez. 2024. ISBN: 978-1-990783-55-5 (siehe S. 134).
- [4] *ECMAScript Language Specification*. Standard ECMA-262, 3rd Edition. Geneva, Switzerland: Ecma International, Dez. 1999. URL: [https://ecma-international.org/wp-content/uploads/ECMA-262\\_3rd\\_edition\\_december\\_1999.pdf](https://ecma-international.org/wp-content/uploads/ECMA-262_3rd_edition_december_1999.pdf) (besucht am 2024-12-15) (siehe S. 134).
- [5] Dan Goodin. "AI-generated code could be a disaster for the software supply chain. Here's why.". In: *Ars Technica*. Hrsg. von Ken Fisher und Jon Stokes. New York, NY, USA: Condé Nast, 29. Apr. 2025. URL: <https://arstechnica.com/security/2025/04/ai-generated-code-could-be-a-disaster-for-the-software-supply-chain-heres-why> (besucht am 2025-05-08) (siehe S. 91–100).
- [6] Hangzhou, Zhejiang, China (中国浙江省杭州市): DeepSeek-AI u. a. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning". (abs/2501.12948), 22. Jan. 2025. doi:10.48550/ARXIV.2501.12948. URL: <https://arxiv.org/abs/2501.12948> (besucht am 2025-04-28). arXiv:2501.12948v1 [cs.CL] 22 Jan 2025 (siehe S. 70–74, 134).
- [7] "How to implement division with round-towards-infinity in Python". In: Hrsg. von Kbob. 24.–26. Aug. 2011. URL: <https://stackoverflow.com/questions/7181757> (besucht am 2025-07-28) (siehe S. 44–53).
- [8] John Hunt. *A Beginners Guide to Python 3 Programming*. 2. Aufl. Undergraduate Topics in Computer Science (UTICS). Cham, Switzerland: Springer, 2023. ISBN: 978-3-031-35121-1. doi:10.1007/978-3-031-35122-8 (siehe S. 134).
- [9] Kent D. Lee und Steve Hubbard. *Data Structures and Algorithms with Python*. Undergraduate Topics in Computer Science (UTICS). Cham, Switzerland: Springer, 2015. ISBN: 978-3-319-13071-2. doi:10.1007/978-3-319-13072-9 (siehe S. 134).
- [10] Mark Lutz. *Learning Python*. 6. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., März 2025. ISBN: 978-1-0981-7130-8 (siehe S. 134).



# References II



- [11] Lucas Mearian. "What are LLMs, and how are they used in generative AI?" *Computerworld*, 7. Feb. 2024. Framingham, MA, USA: CW Communications, Inc. and Needham, MA, USA: Foundry (formerly IDG Communications, Inc.) ISSN: 0010-4841. URL: <https://www.computerworld.com/article/1627101> (besucht am 2025-04-27) (siehe S. 134).
- [12] Beatrice Nolan. "An AI-powered coding tool wiped out a software company's database, then apologized for a 'catastrophic failure on my part'". *Fortune*, 23. Juli 2025. New York, NY, USA: Fortune Media IP Limited. URL: <https://fortune.com/2025/07/23/ai-coding-tool-replit-wiped-database-called-it-a-catastrophic-failure> (besucht am 2025-07-29) (siehe S. 96).
- [13] *Python 3 Documentation*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: <https://docs.python.org/3> (besucht am 2024-07-05) (siehe S. 13–18, 24).
- [14] *Python 3 Documentation. Python Setup and Usage*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: <https://docs.python.org/3/using> (besucht am 2024-07-05) (siehe S. 111, 112, 116, 117).
- [15] Alec Radford, Karthik Narasimhan, Tim Salimans und Ilya Sutskever. *Improving Language Understanding with Unsupervised Learning*. Techn. Ber. San Francisco, CA, USA: OpenAI Inc., 11. Juni 2018. URL: [https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf) (besucht am 2025-04-28) (siehe S. 134).
- [16] Kristian Rother. *Pro Python Best Practices: Debugging, Testing and Maintenance*. New York, NY, USA: Apress Media, LLC, März 2017. ISBN: 978-1-4842-2241-6 (siehe S. 134).
- [17] Ernest E. Rothman, Rich Rosen und Brian Jepson. *Mac OS X for Unix Geeks*. 4. Aufl. Sebastopol, CA, USA: O'Reilly Media, Inc., Sep. 2008. ISBN: 978-0-596-52062-5 (siehe S. 134).
- [18] Stuart J. Russell und Peter Norvig. *Artificial Intelligence: A Modern Approach (AIMA)*. 4. Aufl. Hoboken, NJ, USA: Pearson Education, Inc. ISBN: 978-1-292-40113-3. URL: <https://aima.cs.berkeley.edu> (besucht am 2024-06-27) (siehe S. 134).
- [19] Ahmad Sahar. *iOS 26 Programming for Beginners*. 10. Aufl. Birmingham, England, UK: Packt Publishing Ltd, Nov. 2025. ISBN: 978-1-80602-393-6 (siehe S. 135).



# References III



- [20] Stephen R. Schach. *Object-Oriented Software Engineering*. New York, NY, USA: McGraw-Hill, Sep. 2007. ISBN: 978-0-07-352333-0 (siehe S. 134).
- [21] Drew Smith. *Modern Apple Platform Administration – macOS and iOS Essentials (2025)*. Birmingham, England, UK: Packt Publishing Ltd, Feb. 2025. ISBN: 978-1-80580-309-6 (siehe S. 134).
- [22] *Stack Overflow*. New York, NY, USA: Stack Exchange Inc. URL: <https://stackoverflow.com> (besucht am 2025-02-27) (siehe S. 44–47).
- [23] Marc Stöckel. “Ärger mit Replit: Coding-KI löscht Produktivdatenbank und verweigert Rollback”. *golem.de – IT-News für Profis*, 22. Juli 2025. Berlin, Germany: Golem Media GmbH. URL: <https://glm.io/198349> (besucht am 2025-07-29) (siehe S. 97).
- [24] Marc Stöckel. “Google Antigraity: Vibe-Coding-Tool löscht unerwartet ganzen Datenträger”. *golem.de – IT-News für Profis*, 2. Dez. 2025. Berlin, Germany: Golem Media GmbH. URL: <https://glm.io/202805> (besucht am 2025-12-02) (siehe S. 99).
- [25] The PEP Editors. *Index of Python Enhancement Proposals (PEPs)*. Python Enhancement Proposal (PEP) 0. Beaverton, OR, USA: Python Software Foundation (PSF), 13. Juli 2000. URL: <https://peps.python.org> (besucht am 2025-04-17) (siehe S. 111–117).
- [26] *Python 3 Documentation. The Python Standard Library*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: <https://docs.python.org/3/library> (besucht am 2025-04-27) (siehe S. 111–114, 116, 117).
- [27] *Python 3 Documentation. The Python Tutorial*. Beaverton, OR, USA: Python Software Foundation (PSF), 2001–2025. URL: <https://docs.python.org/3/tutorial> (besucht am 2025-04-26) (siehe S. 111–113, 116, 117).
- [28] *Turbo Pascal® Version 7.0 – User’s Guide*. Scotts Valley, CA, USA: Borland International, Inc., 1992. URL: [https://turbopascal.org/wp-content/uploads/Turbo\\_Pascal\\_Version\\_7.0\\_Users\\_Guide\\_1992.pdf](https://turbopascal.org/wp-content/uploads/Turbo_Pascal_Version_7.0_Users_Guide_1992.pdf) (besucht am 2025-07-28) (siehe S. 5–11, 135).
- [29] Mark Tyson. “AI coding platform goes rogue during code freeze and deletes entire company database – Replit CEO apologizes after AI engine says it ‘made a catastrophic error in judgment’ and ‘destroyed all production data’”. *tom’s Hardware*, 21. Juli 2025. New York, NY, USA: Future US, Inc. URL: <https://www.tomshardware.com/tech-industry/artificial-intelligence/ai-coding-platform-goes-rogue-during-code-freeze-and-deletes-entire-company-database-replit-ceo-apologizes-after-ai-engine-says-it-made-a-catastrophic-error-in-judgment-and-destroyed-all-production-data> (besucht am 2025-07-31) (siehe S. 96).

# References IV



- [30] Bruce M. Van Horn II und Quan Nguyen. *Hands-On Application Development with PyCharm*. 2. Aufl. Birmingham, England, UK: Packt Publishing Ltd, Okt. 2023. ISBN: 978-1-83763-235-0 (siehe S. 134).
- [31] Klaas van Schelven. "Copilot Induced Crash". In: *Bugsink – Self-hosted Error Tracking: Blog*. Utrecht, The Netherlands: Bugsink B.V., 15. Jan. 2025. URL: <https://www.bugsink.com/blog/copilot-induced-crash> (besucht am 2025-05-08) (siehe S. 91–95, 98, 100).
- [32] Brandon Vigliarolo. "Caveat Coder: Google Antigravity vibe-codes user's entire drive out of existence". *The Register*, 1. Dez. 2025. London, England, UK: Situation Publishing Limited. URL: [https://www.theregister.com/2025/12/01/google\\_antigravity\\_wipes\\_d\\_drive](https://www.theregister.com/2025/12/01/google_antigravity_wipes_d_drive) (besucht am 2025-12-01) (siehe S. 99).
- [33] Thomas Weise (汤卫思). *Programming with Python*. Hefei, Anhui, China (中国安徽省合肥市): Hefei University (合肥大学), School of Artificial Intelligence and Big Data (人工智能与大数据学院), Institute of Applied Optimization (应用优化研究所, IAO), 2024–2025. URL: <https://thomasweise.github.io/programmingWithPython> (besucht am 2025-01-05) (siehe S. 134).
- [34] Kevin Wilson. *Python Made Easy*. Birmingham, England, UK: Packt Publishing Ltd, Aug. 2024. ISBN: 978-1-83664-615-0 (siehe S. 134).
- [35] Martin Yanev. *PyCharm Productivity and Debugging Techniques*. Birmingham, England, UK: Packt Publishing Ltd, Okt. 2022. ISBN: 978-1-83763-244-2 (siehe S. 134).
- [36] Moshe Zadka und Guido van Rossum. *Changing the Division Operator*. Python Enhancement Proposal (PEP) 238. Beaverton, OR, USA: Python Software Foundation (PSF), 11. März–27. Juli 2001. URL: <https://peps.python.org/pep-0238> (besucht am 2025-07-28) (siehe S. 44–57).

# Glossary (in English) I



AI Artificial Intelligence, see, e.g.,<sup>18</sup>

debugger A debugger is a tool that lets you execute a program step-by-step while observing the current values of variables. This allows you to find errors in the code more easily<sup>1,16,34</sup>. Learn more about debugging in<sup>33</sup>.

IDE An *Integrated Developer Environment* is a program that allows the user do multiple different activities required for software development in one single system. It often offers functionality such as editing source code, debugging, testing, or interaction with a distributed version control system. For Python, we recommend using PyCharm. On Apple systems, Xcode is often used.

iOS is the operating system that powers Apple iPhones<sup>3,21</sup>. Learn more at <https://www.apple.com/ios>.

JavaScript JavaScript is the predominant programming language used in websites to develop interactive contents for display in browsers<sup>4</sup>.

LLM Large Language Model, see, e.g., an AI technique using (large/deep) neural networks to models trained on huge bodies of text to predict the next word in a text or conversation and continue to do so until a request is answered<sup>11</sup>. Typical examples include DeepSeek<sup>6</sup> and GPT<sup>15</sup>.

macOS or Mac OS is the operating system that powers Apple Mac(intosh) computers<sup>17,21</sup>. Learn more at <https://www.apple.com/macOS>.

OOP Object-Oriented Programming<sup>20</sup>

PyCharm is the convenient Python Integrated Development Environment (IDE) that we recommend for this course<sup>30,34,35</sup>. It comes in a free community edition, so it can be downloaded and used at no cost. Learn more at <https://www.jetbrains.com/pycharm>.

Python The Python programming language<sup>8-10,33</sup>, i.e., what you will learn about in our book<sup>33</sup>. Learn more at <https://python.org>.

# Glossary (in English) II



**Turbo Pascal** is/was a programming language supporting Object-Oriented Programming (OOP) with an IDE providing a debugger and compiler that was popular in the 1990s<sup>28</sup>.

**Xcode** is offers the tools for developing, testing, and distributing applications as well as an IDE for Apple platforms such as macOS and iOS<sup>19</sup>.

