

AI hallucinations

- [Intro](#)
- [Nushell "fill"](#)

Intro

For learning, I like to use LLMs to ask questions, get recommendations or review what I could do better with code I've written. Usually it is very straight forward and helping, but there are some occurances, which are just wrong, or maybe even dangerous. For example I had some recommendations from LLMs how to write something, what would create security issues.

In this book I want to document those hallucinations. The purpose is neither bashing LLMs, nor glorifying it. It is a tool, like a knife. It can be helpful and harmful. It depends how you use it, and you should use it always with care.

The purpose is for entertainment purpose, to see what errors some LLMs make, and on the other side, for educational purpose, how wrong LLMs can be, and how to handle situations without LLMs.

Nushell "fill"

I'm currently learning Nushell and I wanted to know, how I can fill a number with leading zeros. ChatGPT answered this:

Dabei bedeutet:

- `-a 0` → mit `0` auffüllen
- `-w 5` → auf eine Gesamtbreite von 5 Zeichen

Zum Beispiel:

```
nu
let x = 42
print ($x | fill -a 0 -w 5)
```

This translates to:

- `-a 0` -> fill with `0`
- `-w 5` -> to a width of 5 characters

Okay, I found out, that the command `fill` exists. But the code just did nothing. Why? because the flags are wrong.

In this scenario I learned about fill, but I had to check the flags. Usually tools bring manuals with, like the unix manual pages, a help command, documentation website, and so on. I would recommend at this point not to ask AI, but instead improving your search skills by searching the help for the command. In my case, using nushell, I just typed `help fill`.

The help page for fill shows this:

```
Fill and align text in columns.

Search terms: display, render, format, pad, align, repeat

Usage:
> fill {flags}

Flags:
-h, --help: Display the help message for this command
-w, --width <int>: The width of the output. Defaults to 1.
-a, --alignment <string>: The alignment of the output. Defaults to Left (Left(l), Right(r), Center(c/m), MiddleRight(cr/mr)).
-c, --character <string>: The character to fill with. Defaults to ' ' (space).
```

So `-a` is not the fill character, it is the alignment, which should be `r`. But there is also `-c`, which is the character, which should be `0`. Thus creating the correct answer:

```
let x = 42
```

```
print ($x | fill -a r -c 0 -w 4)
```