

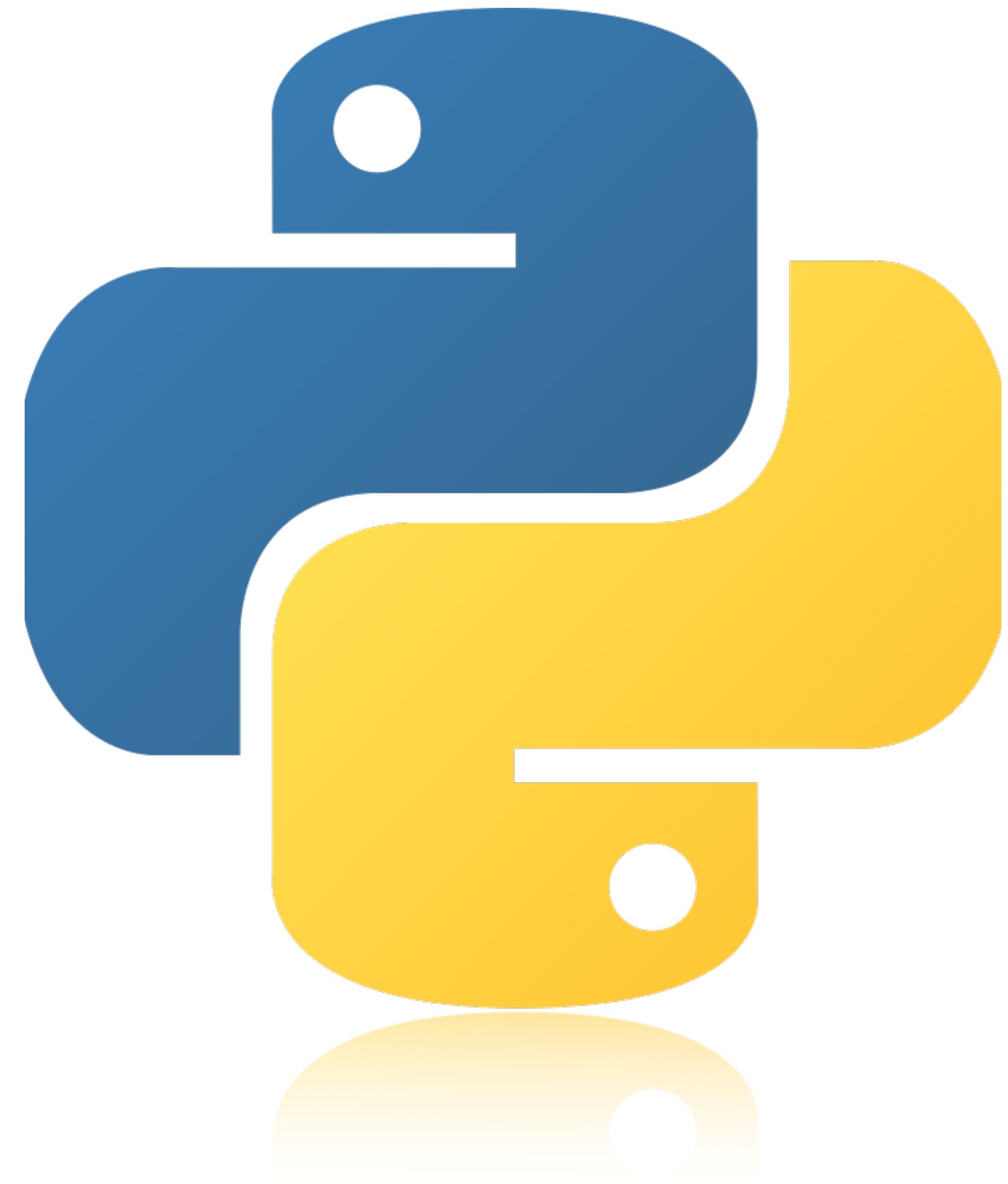
Sudoku lösen mit Python

```
182     for file in files:
183         try:
184             s = Sudoku(file, recording = True)
185             s.solve()
186             if s.solved == True:
187                 solved_grids.append(file)
188             if s.exceptions:
189                 exceptions[file] = s.exceptions
190             h = Heatmap(file)
191             h.plot_data()
192         except:
193             pass
194     return solved_grids, exceptions
```

Abschlusspräsentation - "Programmieren mit Python"
Kurs bei Herrn Dr. Feiler

Inhalt

- Features des Projekts
- Angewendete Techniken
- Grundlagen von Sudokus
- Vorstellung des Algorithmus
- Ausblick
- Live-Demonstration



Features des Projekts

- Sequenzielles Lösen von Sudokus
- Datenaufzeichnung und -visualisierung
- Klassen zum Lösen und Plotten
- User-Interface für die Kommandozeile
- Tool zum Verifizieren des Solvers

```
This is is a solver for sudoks.
```

```
You can use the following commands:
```

- add <filename>	-> add a file to files
- remove <filename>	-> remove a file from files
- files	-> print the files added to files
- print <filename>	-> print a file in the terminal
- list	-> print a file from grids/
- quit	-> quit the program
- solve	-> start the solving sequence
- clear	-> clear the terminal
- help	-> show this help

```
>>> list
```

```
The following grids are available in /grids:
```

```
easiest.csv  
intermediate.csv  
wikipedia.csv  
dummy.csv  
difficult.csv  
not_fun.csv  
unsolvable.csv
```

```
>>> add not_fun
```

```
Adding not_fun.csv to files...
```

```
>>> █
```

```
>>> █
```

```
Adding not_fun.csv to files...  
>>> add not_fun
```

Angewendete Techniken

- Objektorientierte Programmierung
- Interaktion mit dem User per `input()`
- Daten Input / Output mit `csv` und `json`
- Abfangen von Fehlern mit `try / except`
- Plotten mit `matplotlib.pyplot`

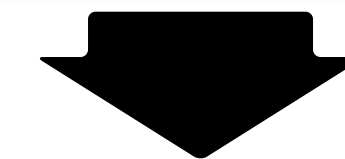


Sudoku (klassisch)

Logisches Zahlenrätsel

- Quadratisches Zahlengitter mit 9 x 9 Zellen
- Es sind mehr oder weniger Ziffern vorgegeben
- Das Gitter muss mit den Ziffern 1 bis 9 befüllt werden
- Pro Zeile, Spalte und "Einheit" (3 x 3 Sub-Zelle) darf jede Ziffer nur ein Mal vorkommen
- Üblicherweise Lösen durch einen Menschen
 - Logisches Denken
 - Kombinieren

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9



5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9



Sudoku

Lösen mit Backtracking: Brute Force

```
1 def step():
2     if ziffer[cursor] < 9:
3         | ziffer += 1
4         | if loesung_moeglich():
5         |     | cursor += 1
6     else:
7         | backtrack()
8
9 def backtrack():
10    ziffer[cursor] = 0
11    cursor -= 1
12
13 while check_solution() == False:
14     try:
15         | step()
16     except IndexError:
17         | break
```

Grober Ablauf des Backtracking-Algorithmus

5	3	1	2	7	2	1		
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Vorwärtsschritte

Sudoku

Lösen mit Backtracking: Brute Force

```
1 def step():
2     if ziffer[cursor] < 9:
3         | ziffer += 1
4         | if loesung_moeglich():
5         |     | cursor += 1
6     else:
7         | backtrack()
8
9 def backtrack():
10    ziffer[cursor] = 0
11    cursor -= 1
12
13 while check_solution() == False:
14     try:
15         | step()
16     except IndexError:
17         | break
```

Grober Ablauf des Backtracking-Algorithmus

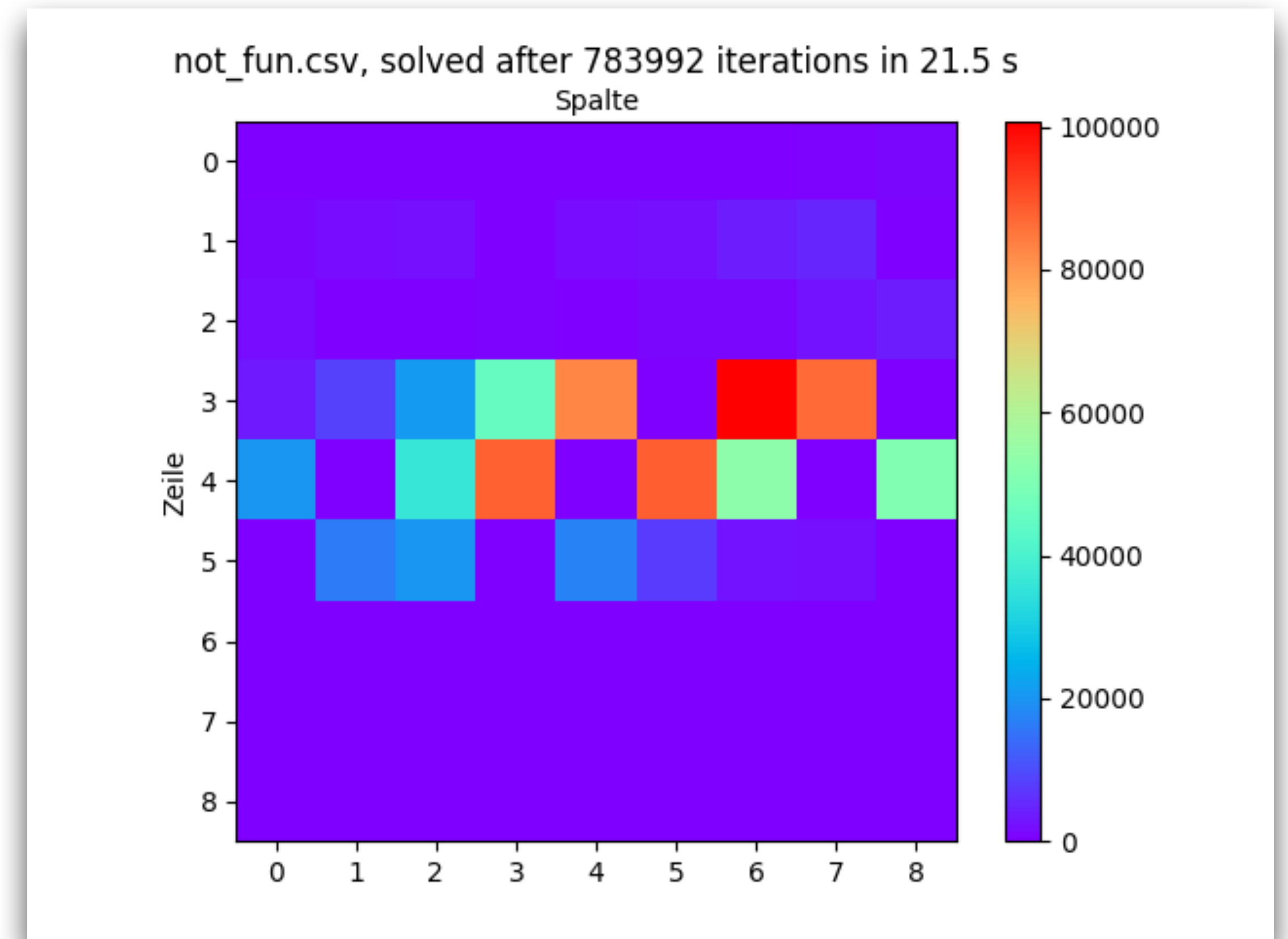
5	3	1	2	7	4	8	9	8
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Backtracking

Visualisierung der Daten

Statistiken und Heatmap

- Daten aus dem Lösungsprozess werden als .csv und .json aufgezeichnet
- In einem Diagramm werden dargestellt:
 - Der Dateiname
 - Die Anzahl der Iterationen
 - Die benötigte Zeitdauer
 - Eine Heatmap zeigt, wie oft welche Zelle iteriert wurde



Exemplarischer Plot

Ausblick

- Refactoring des sehr umfangreichen Codes
- Ausführlichere Kommentare schreiben
- Lösen von Anti-Brute-Force-Rätseln
- Implementierung analytischer Methoden
- Bedienung mit einem GUI
- Schöner Gestaltung der Plots
- Die spielerisch angewendeten Techniken lassen sich auch für "echte" Probleme einsetzen

					3		8	5
		1		2				
			5		7			
		4				1		
	9							
5							7	3
		2		1				
				4				9



Keine Lösung nach 10^9 Iterationen

```
1 if fragen:
2     for frage in fragen:
3         | frage.stellen()
4 else:
5     live_demo()
```