11. April 2025

FDS2 - Übung 6

SS 2025

Tim Peko

1. Beispiel 1: Sudoku

1.1. Lösungsansatz

- **`sudoku::read`**:
  Liest ein Sudoku $n^2 \times n^2$ der Ordnung n aus einem `std::istream`. Das Format beginnt mit der Ordnung n, gefolgt von n^2 Zeilen mit jeweils n^2 Zahlen (0 für leere Zellen). Es wird ein Grid gemäß der Ordnung initialisiert und dann die Zellen zuerst der Spalte nach, dann Zeile für Zeile eingelesen.
- **`sudoku::is_valid`**:
  Überprüft, ob das Setzen einer Zahl in einer Zelle gemäß den Sudoku-Regeln (Zeile, Spalte, Block) gültig ist, indem über die querenden Zellen iteriert und nach einem Duplikat der zu setzenden Zahl gesucht wird.
- **`sudoku::simplify`**:
  Reduziert den Suchraum vor der Exhaustion durch Constraint Propagation. Dazu werden zwei Strategien angewendet:
  1. `check_single_possibility`: Füllt Zellen, für die nur noch eine einzige Zahl möglich ist.
  2. `check_unique_in_unit`: Füllt Zellen, wenn eine bestimmte Zahl nur an einer einzigen Stelle in einer Zeile, Spalte oder einem Block platziert werden kann.

  Die `apply_constraints` Funktion wendet diese Strategien iterativ solange an, bis keine weiteren Vereinfachungen mehr möglich sind.
- **`sudoku::solve`**:
  Implementiert einen rekursiven Backtracking-Algorithmus (Exhaustion).
  1. Sucht die nächste leere Zelle.
  2. Wenn keine leere Zelle gefunden wird, ist das Sudoku gelöst.
  3. Probiert für die leere Zelle alle Zahlen von 1 bis n.
  4. Wenn eine Zahl gültig ist, wird sie gesetzt und `solve` rekursiv aufgerufen.
  5. Wenn der rekursive Aufruf erfolgreich ist, wird `true` zurückgegeben. Das Sudoku ist gelöst.
  6. Wenn nicht, wird die Zahl zurückgesetzt (Backtracking) und die nächste Zahl probiert.
  7. Wenn keine Zahl funktioniert, wird `false` zurückgegeben. Das Sudoku ist nicht lösbar.

Die `soduku` Klasse wurde in einer eigenen Datei implementiert. Die `main01.cpp` Datei definiert die `main` Funktion und liest die Sudoku-Datei ein. Der Pfad zu dieser Datei kann optional über die Kommandozeile angegeben werden. Die Ausgabe erfolgt auf die Standardausgabe.

Referenzierte Dateien können unter `assets/` gefunden werden.

1.2. Testfälle

1.2.1. Testfall 1: Standard Sudoku

Input:
datei: `sudoku-I-3.txt`

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

Output:

```
------------------
|  1  |  6 5|4    |
|     |  8 4|1    |
|4    |     |  7  |
------------------
|  5  |1 9  |     |
|    3|     |7    |
|     |  3 7|  5  |
------------------
|  8  |     |    3|
|    2|6 5  |     |
|    9|8 1  |  2  |
------------------

------------------
|9 1 7|3 6 5|4 8 2|
|2 3 5|7 8 4|1 9 6|
|4 6 8|9 2 1|3 7 5|
------------------
|7 5 6|1 9 8|2 3 4|
|8 2 3|5 4 6|7 1 9|
|1 9 4|2 3 7|6 5 8|
------------------
|5 8 1|4 7 2|9 6 3|
|3 7 2|6 5 9|8 4 1|
|6 4 9|8 1 3|5 2 7|
------------------
```

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

Ergebnis: success

1.2.2. Testfall 2: Ungültige Eingabe (Zu wenig Reihen)

Input: datei: `sudoku-testcase-2.txt`

```
3
|0|1|0|0|6|5|4|0|0|
|0|0|0|0|8|4|1|0|0|
```

Output:

```
error: not enough rows in sudoku grid. expected 9 rows.

Could not solve the Sudoku.
```

Ergebnis: success

1.2.3. Testfall 3: Ungültige Eingabe (Falsche Anzahl Zahlen pro Zeile)

Input: datei: `sudoku-testcase-3.txt`

```
2
|1|0|3|0|
|0|2|0|
|3|0|0|0|
|0|4|0|0|1|
```

Output:

```
error: not enough numbers in row |0|2|0|. expected 4 numbers, got 3.

Could not solve the Sudoku.
```

Ergebnis: success

1.2.4. Testfall 4: Ungültige Eingabe (Ungültiger Wert)

Input: datei: `sudoku-testcase-4.txt`

```
2
|1|0|3|0|
|0|4|0|5|
|2|3|4|0|
|0|0|1|0|
```

Output:

```
error: invalid value 5 at row |0|4|0|5|, col 3. must be between 0 and 4.

Could not solve the Sudoku.
```

Ergebnis: success

1.2.5. Testfall 5: Unlösbares Sudoku

Input: datei: `sudoku-testcase-5.txt`

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

Output:

```
------------------
|2    |9    |     |
|     |     |  6  |
|     |    1|     |
------------------
|5   2|6    |4   7|
|     |    4|1    |
|     |  9 8|  2 3|
------------------
|     |    3|  8  |
|    5|  1  |     |
|    7|     |     |
------------------
```

```
failure: failed to solve the Sudoku.
```

```
Could not solve the Sudoku.
```

Ergebnis: success

1.2.6. Testfall 6: Falsches Sudoku

Input: datei: `sudoku-testcase-6.txt`

1	2	3	4
3	4	1	2
2	1	4	3
4	3	1	2

Output:

```
error: invalid initial sudoku state. duplicate number 1 found at row 1, col 2.
```

```
Could not solve the Sudoku.
```

Aufwand in h: 9