Sudoku Skyscraper - Funktionsaufrufe

Diese Liste zeigt die Reihenfolge, in der die Funktionen aufgerufen werden, um das Sudoku zu lösen. Der Ablauf folgt dem rekursiven Backtracking-Ansatz.

Chronologische Reihenfolge der Funktionsaufrufe:

1. main
2. copy_matrix
3. solve
4. recursion
5. next_position
6. is_available
7. column_available
8. row_available
9. box_available
10. reverse_recursion
11. reverse_next_position
12. compare_solutions
13. print_sudoku

Funktionen und ihre Erklärungen:

main:

Dies ist der Startpunkt des Programms. Es überprüft die Eingabe und bereitet die Daten für die Sudoku-Lösung vor.
Wenn die Eingabe gültig ist, ruft es die Funktion 'solve' auf.
Die Hauptfunktion ist der Einstiegspunkt des Programms. Sie überprüft,
ob die richtige Anzahl an Argumenten übergeben wurde. Danach kopiert sie die Eingabe in zwei separate 9x9-Matrizen (tab1, tab2) und ruft die 'solve'-Funktion auf, die das Sudoku löst.

copy_matrix:

Diese Funktion kopiert die eingegebenen Sudoku-Daten in eine 9x9-Matrix. Sie stellt sicher, dass das Sudoku-Grid korrekt ausgelesen wird.

Diese Funktion kopiert die Eingabedaten (die aus den Befehlszeilenargumenten kommen) in eine 9x9-Sudoku-Matrix. Sie überprüft auch,
ob die Eingabe gültig ist (9 Zeichen pro Zeile, nur zulässige Zeichen wie Zahlen oder '.').

solve:

Die Hauptfunktion zur Lösung des Sudokus. Sie ruft 'recursion' auf, um das Sudoku mit einer Vorwärtslösung zu versuchen.
Dann ruft sie 'reverse_recursion' auf, um zu prüfen, ob die Lösung eindeutig ist.

Diese Funktion übernimmt das eigentliche Lösen des Sudokus. Sie versucht zuerst eine normale Lösung über die 'recursion'-Funktion zu finden.
Dann überprüft sie mit 'reverse_recursion', ob die Lösung eindeutig ist. Falls die Lösungen nicht übereinstimmen, wird ein Fehler ausgegeben.

recursion:

Diese Funktion versucht, das Sudoku mit Backtracking zu lösen. Sie sucht eine freie Position, testet Ziffern von 1 bis 9 und ruft sich selbst rekursiv auf, bis eine Lösung gefunden ist.

Diese Funktion implementiert das rekursive Backtracking, um das Sudoku zu lösen. Sie sucht mit 'next_position' nach der nächsten freien Zelle und testet alle Zahlen von '1' bis '9'.
Falls eine Zahl passt ('is_available'), wird sie eingesetzt und die Funktion ruft sich selbst auf. Falls keine Lösung gefunden wird, wird die Zelle zurückgesetzt (Backtracking).

next_position:

Findet die nächste freie Position im Sudoku (von links oben nach rechts unten). Wird von 'recursion' genutzt.

Findet die nächste leere Position ('.') im Sudoku, indem sie das Feld von links nach rechts, oben nach unten durchsucht. Diese Funktion ist essenziell für das rekursive Lösen.

is_available:

Prüft, ob eine Zahl an einer bestimmten Position eingefügt werden kann. Dafür ruft sie 'column_available', 'row_available' und 'box_available' auf.

Diese Funktion überprüft, ob eine Zahl an einer bestimmten Position erlaubt ist. Dafür ruft sie drei Hilfsfunktionen auf: 'column_available' (Spaltenprüfung), 'row_available' (Zeilenprüfung) und 'box_available' (3x3 Block-Prüfung).

column_available:

Überprüft, ob eine Zahl bereits in der aktuellen Spalte vorkommt.

Überprüft, ob eine bestimmte Zahl bereits in der Spalte vorkommt. Falls ja, gibt die Funktion '0' zurück (nicht erlaubt), andernfalls '1'. Sie geht von oben nach unten durch die Spalte.

row_available:

Überprüft, ob eine Zahl bereits in der aktuellen Zeile vorkommt.

Überprüft, ob eine bestimmte Zahl bereits in der Zeile vorkommt. Falls ja, gibt die Funktion '0' zurück (nicht erlaubt), andernfalls '1'. Sie prüft von links nach rechts.

box_available:

Überprüft, ob eine Zahl bereits im 3x3-Block vorkommt.

Diese Funktion überprüft, ob eine Zahl bereits im 3x3-Block vorkommt. Die Koordinaten der aktuellen Zelle werden genutzt, um den entsprechenden Block zu finden und dann durch ihn zu iterieren.

reverse_recursion:

Löst das Sudoku von unten rechts nach oben links, um zu prüfen, ob eine eindeutige Lösung existiert.

Funktioniert ähnlich wie 'recursion', aber geht das Sudoku in umgekehrter Richtung (von unten rechts nach oben links) durch. Dadurch wird überprüft, ob das Sudoku eine eindeutige Lösung hat.

reverse_next_position:

Findet die nächste freie Position von unten rechts nach oben links.

Findet die nächste leere Position ('.'), jedoch in umgekehrter Richtung von unten rechts nach oben links. Dies wird in der 'reverse_recursion'-Funktion genutzt.

compare_solutions:

Vergleicht die vorwärts und rückwärts gelösten Sudoku-Gitter, um zu sehen, ob sie übereinstimmen.

Vergleicht zwei Sudoku-Lösungen (eine vorwärts und eine rückwärts gelöst). Falls beide übereinstimmen, ist die Lösung eindeutig. Falls nicht, wird ein Fehler ausgegeben.

print_sudoku:

Gibt das gelöste Sudoku im richtigen Format auf dem Bildschirm aus.

Diese Funktion gibt das gelöste Sudoku auf der Konsole aus. Sie iteriert durch das 9x9-Grid und gibt es in einem lesbaren Format aus.