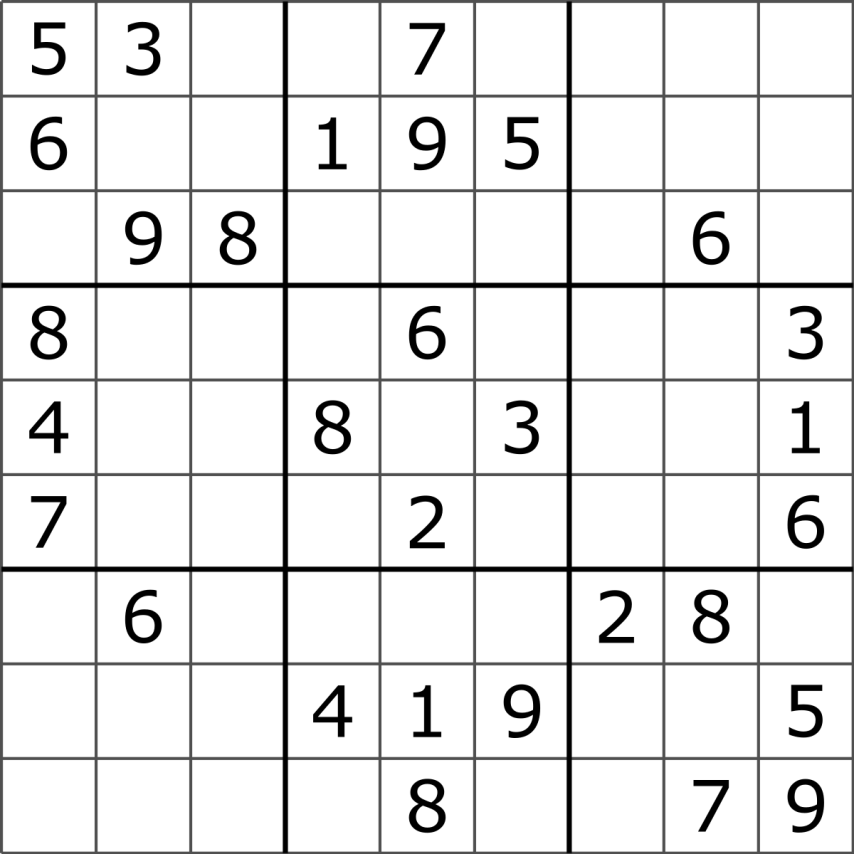
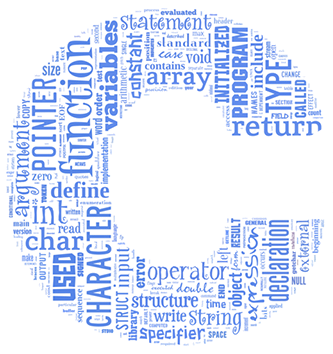
**2018**

Robin Winkler, Fabian Gutendorf, Moritz Bulthaup

Abgabetermin: 22.05.2018



Dokumentation - Sudoku Projekt

Inhaltsverzeichnis

[1 Ausgangssituation 2](#_Toc514788230)

[1.1 Projektziele und Teilaufgaben 2](#_Toc514788231)

[1.2 Kundenanforderungen 2](#_Toc514788232)

[1.3 Projektumfeld 2](#_Toc514788233)

[1.4 Prozessschnittstellen 3](#_Toc514788234)

[2 Ressourcen und Ablaufplanung 4](#_Toc514788235)

[2.1 Personalplanung 4](#_Toc514788236)

[2.2 Terminplanung, Ablaufplan 4](#_Toc514788237)

[3 Durchführung und Auftragsbearbeitung 6](#_Toc514788238)

[3.1 Prozessschritte, Vorgehensweise, Qualitätssicherung 6](#_Toc514788239)

[3.2 Abweichungen, Anpassungen, Entscheidungen 6](#_Toc514788240)

[4 Projektergebnisse 7](#_Toc514788241)

[4.1 Soll-Ist-Vergleich 7](#_Toc514788242)

[4.2 Abweichungen, Anpassungen 7](#_Toc514788243)

[5 Anlagen 8](#_Toc514788244)

[5.1 Kundendokumentation 8](#_Toc514788245)

[5.2 Relevante technische Dokumente/Anmerkungen 8](#_Toc514788246)

[5.3 Technische Hilfestellungen 9](#_Toc514788247)

[5.4 Literaturverzeichnis 9](#_Toc514788248)

# Ausgangssituation

## Projektziele und Teilaufgaben

Die folgende Projektarbeit stellt den Ablauf des Sudoku – Projektes dar.   
Ziel des Projektes ist eine spielbare Version des Logikrätsels Sudoku in der Programmiersprache C.

„Sudoku ist eine Gattung von Logikrätseln, die aus den lateinischen Quadraten entstand. In der üblichen Version ist es das Ziel, ein 9×9-Gitter mit den Ziffern 1 bis 9 so zu füllen, dass jede Ziffer  
in jeder Spalte, in jeder Zeile und in jedem Block (3×3-Unterquadrat) genau einmal vorkommt. Ausgangspunkt ist ein Gitter, in dem bereits mehrere Ziffern vorgegeben sind. In Zeitungen und  
Zeitschriften werden heute regelmäßig Sudoku Rätsel veröffentlicht.“[[1]](#footnote-1)

## Kundenanforderungen

Zur Lösung des Projektes sind einige Anforderungen getroffen worden:

* Anzeige des Gitters (9x9)  
  *Programm zeigt auf dem Bildschirm ein Gitter aus 9x9 Feldern an, die mit den passenden  
  Zahlen gefüllt werden müssen.*
* Auswahl des aktuellen Feldes auf dem Spielfeld  
  *Spieler kann auswählen, welches Feld im Gitter bearbeitet werden soll.*
* Eingabe bzw. Veränderung des Wertes eines Feldes  
  *Spieler kann den Inhalt des aktuellen Feldes im Gitter bearbeiten. Ein Feld darf entweder  
  leer sein oder muss eine Zahl zwischen 1 und 9 enthalten.*
* Sichere Verarbeitung der Benutzereingaben  
  *Programm reagiert nur auf gültige Eingaben des Benutzers.*
* Auswahl eines zu spielenden Sudoku  
  *Spieler kann ein zu spielendes Sudoku auswählen. Eine Auswahl von Sudoku verschiedener  
  Schwierigkeit kann fest vorgegeben. Alternativ können neue Sudoku automatisch erzeugt  
  oder aus einer Datei eingelesen werden.*
* Erkennung einer korrekten Lösung des Sudoku  
  *Programm erkennt automatisch oder nach Aufforderung ein korrekt gelöstes Sudoku.*
* Zeitmessung während des Spiels  
  *Programm zeigt die bereits vergangene Zeit zur Lösung eines Sudoku an. Die Zeit wird  
  entweder am Ende des Spiels, nach jedem Zug oder laufend angezeigt.*

## Projektumfeld

Das Sudoku Projekt wurde im Rahmen des Programmierprojektes im Lernfeld 6: Entwickeln und Bereitstellen von Anwendungssystemen erstellt.

Dieses sollte in der Programmiersprache C in der Entwicklungsumgebung CodeBlocks[[2]](#footnote-2) geschrieben und implementiert. Außerdem war die Einbindung von Bibliotheken erlaubt.

Wichtige Stellen im Quellcode, müssen kommentiert werden, um die dahinterstehende Logik zu verstehen. Außerdem muss das Sudoku Projekt in Funktionen und Dateien strukturiert werden, damit dieses übersichtlich bleibt.

## Prozessschnittstellen

In diesem Projekt sind mehrere Entwickler beteiligt gewesen, sodass eine Möglichkeit der Versionsverwaltung nötig war, damit alle immer auf den aktuellen Stand und Änderungen nachverfolgbar waren.

Hierzu wurde Git verwendet. Eine solche Form der Versionsverwaltung ist in größeren Projekten, die mehrere Entwickler erfordern, nahezu unerlässlich.

Bei einer Versionsverwaltung besitzt jeder Entwickler eine Lokale Arbeitskopie, die er verändern kann. Sind seine Änderungen vollständig, kann er diese auf den Server laden. Dabei wird dieser Version ein Zeit- und ein Benutzerstempel gesetzt, sodass andere Entwickler die Änderungen in Ihre Arbeitskopie einbinden können.

Diese Verwaltung übernimmt Git. Mit einfachen Kommandos in der Git – Konsole oder Sourcetree[[3]](#footnote-3) (einem grafischen Interface für Git) wie **Push** (Dateien auf den Server schieben) und **Pull** (Dateien von dem Server ziehen) werden die geänderten Dateien in die Lokale Arbeitskopie eingebunden.

Die weiteren Entwickler können somit gleichzeitig an verschiedenen Stellen des Quellcodes arbeiten, ohne sich gegenseitig zu hindern.

Wurden einzelne Aufgaben fertiggestellt, werden diese auf den Server hochgeladen. Anschließend laden die Entwickler sich diese Änderungen in Ihre lokale Arbeitskopie und arbeiten weiter.

Wenn alle Aufgaben abgeschlossen sind, wird das gesamte Projekt nochmal aufgeräumt und hochgeladen.

# Ressourcen und Ablaufplanung

## Personalplanung

Die Aufteilung der Aufgaben erfolgte anhand des Kanban – Boards.  
Zu Beginn des Projektes wurden die Kundenanforderungen aufgeteilt, sodass einzelne Aufgaben entstanden. Diese wurden anschließend einer groben Reihenfolge zugewiesen, um einen reibungslosen Verlauf zu gewährleisten.

Die grobe Reihenfolge bestand z.B. aus:  
1. Anlegen des Git Repository  
2. Projekt in Codeblocks erstellen  
3. Sudoku 9x9 Raster anzeigen  
4. Eingaben des Nutzers  
5. Die weiteren Kundenanforderungen

Die oben genannte Reihenfolge wurde gewählt, da diese Punkte Voraussetzung für weitere sind. Um z.B. die Eingaben des Nutzers zu verarbeiten, muss das Raster vorher angezeigt werden.

Diese Aufgaben wurden nun in von den drei Entwicklern abgearbeitet.

Im weiteren Projektverlauf wurden die Kundenanforderungen ebenfalls in Unterpunkte gegliedert.

Die neu aufgeteilten Aufgaben wurden somit abgearbeitet und im Kanbanboard auf den aktuellen Stand gehalten.

Zeitgleich wurde die Dokumentation begonnen. Während zwei Entwickler die ersten Programmieraufgaben bearbeitet haben, hat ein anderer Entwickler bereits mit der Dokumentation angefangen, sodass später nur noch ergänzt werden muss.  
Dieses sorgte vor allem dafür, dass nie Leerlauf entstand, da die Aufgaben, die aufeinander aufbauen so abgearbeitet wurden, dass ein Entwickler schon die Dokumentation schrieb, während die anderen die Aufgabe im Quellcode umsetzten.

## Terminplanung, Ablaufplan

Um einen reibungslosen Ablauf zu gewährleisten, wurde für dieses Projekt die Softwareplanungsmethode KanbanFlow benutzt.

Kanban ist eine agile Methode für evolutionäres Change Management.[[4]](#footnote-4)  
Anstatt wenigen großen Änderungen, werden viele kleine Änderungen durchgeführt, welches die Fehleranfälligkeit reduziert. Das Projekt wird in viele Aufgaben aufgeteilt, diese werden anschießend abgearbeitet und zusammengefügt.

Bevor dieses jedoch durchgeführt werden kann, sind ebenfalls Vorbereitungen nötig.

Die Vorhandene Arbeit sowie die Anforderungen des Kunden werden in einem Kanban – Board visualisiert.

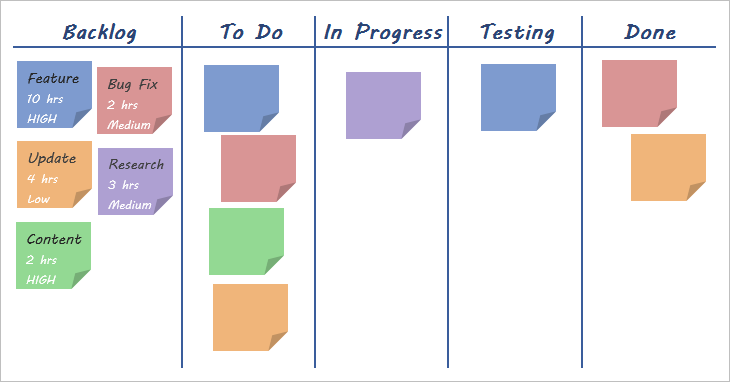


Abbildung Kanban – Board

Die einzelnen Spalten stellen die Schritte da, die die Prozesse durchlaufen müssen. Ist ein Prozess abgeschlossen, kann dieser in den nächsten Status verschoben werden.

Vorteile dieser Planungsmethode:

* Gute Übersichtlichkeit über Prozesse und Probleme
* Hohe Anpassbarkeit
* Teamintegrität
* Intuitiv
* Fehlerreduzierung

Die gewählte Methode weist somit enorme Vorteile auf, um die Umsetzung des Projekts einfach zu halten. Jedoch muss vorher genau geplant werden, in welcher Reihenfolge die einzelnen Anforderungen abgearbeitet werden und wie diese aufgeteilt werden.

# Durchführung und Auftragsbearbeitung

## Prozessschritte, Vorgehensweise, Qualitätssicherung

Die Qualität der Software wird durch ausgiebige Kontrollen schon während des Entwicklungsprozesses gewährleistet. Ein Entwickler schreibt den Quellcode und setzt seine Aufgabe im Kanban – Board auf Kontrolle. Nun schauen die beiden anderen Entwickler erst auf die Funktion und anschießend auf den Quellcode. So wird eine hundertprozentige Funktionalität gewährleistet. Dabei wird der Aufgabe eine Nachricht angefügt, die den aktuellen Stand eineindeutig zuzuordnen ist. Somit ist sichergestellt, dass spätere Änderungen auch wieder kontrolliert werden können. Nachdem beide Entwickler mit der Kontrolle fertig sind wird die Aufgabe im Kanban – Board entweder auf fertig oder zurück in die Bearbeitung geschoben. Am Ende des Projektes werden die Arbeitsergebnisse nochmal in der Funktion getestet.

## Abweichungen, Anpassungen, Entscheidungen

Um keine Aufgaben doppelt zu bearbeiten, wurden die einzelnen Anforderungen aufgeteilt, sodass jeder Entwickler seine Aufgabe abarbeitet. Anschließend werden die Funktionen von beiden anderen Entwicklern ausgiebig analysiert und die Funktionalität getestet.

Um eine funktionierende Software zu entwickeln, mussten diese Funktionen in das Projekt integriert werden.

Nachdem die Kundenanforderungen abgearbeitet waren, wurden weitere Features implementiert.

Eine Auswahl von Sudoku verschiedener Schwierigkeitsgraden ist fest vorgegeben. Um diese Funktion zu erweitern, bestand die Möglichkeit neue Sudokus automatisch zu generieren oder aus einer Datei auszulesen.

Hierbei entschieden wir uns für letzteres. Auslesen aus einer Datei ist übersichtlicher und kann auch von Außerhalb bearbeitet werden. So kann jeder der dieses Programm nutzt, seine eigenen Sudokus erstellen.

Des Weiteren kann der aktuelle Spielstand gespeichert und im Hauptmenu geladen werden. Somit kann das Spiel unterbrochen und später fortgesetzt werden. Hierbei greifen wir ebenfalls auf die Dateien zu, die wir für die Auswahl der Sudokus nehmen.

Anschließend wurde das Lösen eines Sudokus implementiert. Dieser ermöglicht den Nutzer einen Lösungshinweis anzuzeigen.

# Projektergebnisse

## Soll-Ist-Vergleich

In dem Sudoku Projekt wurden alle Grundfunktionen sowie alle Erweiterungen implementiert.

Die Kundenanforderungen wurden hierbei vollständig abgearbeitet. Nachdem diese Anforderungen zu hundert Prozent implementiert wurden, ging es an die Erweiterungen.

Diese wurden ebenfalls zu hundert Prozent in das Projekt integriert.

## Abweichungen, Anpassungen

Während das Projekt umgesetzt wurde, wurden allerdings auch einige Anpassungen gemacht.

Zu Beginn des Projektes wurde bei jedem Funktionsaufruf das Sudoku Array übergeben. Dieses sorgte jedoch für Unübersichtlichkeit, welches Array bereits bearbeitet wurde und welches noch bearbeitet werden muss. Um dieses zu beheben, wurde aus dem Sudoku Array ein Struct, indem alle wesentlichen Informationen vorhanden sind. So z.B. die Anzahl an Zügen des Nutzers, das Original Sudoku, um bei dem Speichern und Laden die eingegebenen von den festen Zahlen unterscheidet und vieles mehr.

Des Weiteren wurde zu Beginn beschlossen, dass das Sudoku ein 9x3x3 Array darstellen soll. Im Verlauf wurden verschiedene Umwandlungsfunktionen geschrieben, die aus den 9x3x3 Array ein 9x9 Array erstellt haben. Zum Ende wurde dieses wieder verbessert, indem nur manche Funktionen das Sudoku als 9x3x3 Array erwarten. Für diese wurden kleine Konverter gebaut.

Schließlich sind verschiedene Befehle als Constants definiert worden, da diese auf verschiedenen Plattformen unterschiedlich dargestellt werden. Des Weiteren sind dort verschiedene Einstellungen zu finden, wie z.B. die Farben zu finden, die das Sudoku anpassbar machen.

# Anlagen

## Kundendokumentation

Nach Programmstart erscheint das Hauptmenü. Dort kann man mit Betätigung der [e]-Taste das Programm beenden. Im selbigen Menü kann man mit druck auf der [l]-Taste ein zuvor abgespeichertes Spiel laden. Mit Betätigung der [n]-Taste im Hauptmenü kann man ein neues Sudoku starten. Man wird aufgefordert einen Schwierigkeitsgrad zu wählen, wobei 1 besonders leicht, 2 mittel schwer und 3 besonders schwer ist. Sie können sich an diesem Zeitpunkt mit der [z]-Taste wieder zurück zum Hauptmenü kommen. Nachdem man entweder ein Spiel geladen oder ein Schwierigkeitsgrad ausgewählt hat, gelangt man zur Spieloberfläche. Auf dieser kann man mit den Pfeiltasten den Cursor bewegen. An Stelle des Cursors kann man die Zahlen 1-9 mit Eingabe dieser setzen. Mit einem Druck auf die 0 wird die unter dem Cursor befindliche Zahl gelöscht. Mit einem Druck auf die [s]-Taste kann man das Spiel abspeichern und beenden. Des weiterem kann man auch hier wieder zurück zum Hauptmenü mit der [m]-Taste. Um das Spiel ohne zu speichern zu beenden kann man die [e]-Taste drücken. Mit der [h]-Taste erhält man einen kleinen Lösungshinweis an der Stelle des Cursors.

## Relevante technische Dokumente/Anmerkungen

Um den Quellcode übersichtlich zu gestalten, wurde ein Code-Stil definiert. Dieser ist von allen Beteiligten in der Entwicklung einzuhalten.

Die verschiedenen Variablen wurden im camelCase geschrieben. Die Variablennamen fangen somit immer klein an, wenn ein neues Wort vorkommt, wird der erste Buchstabe hiervon großgeschrieben.

Bei Arrays und Structs gibt es noch eine kleine Besonderheit. Bevor diese in camelCase geschrieben werden, bekommen Sie noch ein kleines Kürzel.

Arrays beginnen mit „a\_“ gefolgt von dem Array Namen, um Sie von restlichen Variablen zu unterscheiden.  
Bei Structs ist das vorgehen identisch. Diese beginnen mit einem „s\_“ ebenfalls gefolgt von dem Struct Namen.

Außerdem wurde das gesamte Projekt in Dateien und Funktionen gekapselt und gegliedert, sodass es Übersichtlich bleibt.

Damit das Programm Plattformunabhängig funktioniert, wurden Konstanten definiert. Diese stellen ebenfalls die Aktuelle Plattform (Windows und Linux) ein.

## Technische Hilfestellungen

Um verschiedene Algorithmen umzusetzen, wurde in Funktionen die Hilfe von diversen Onlineplattformen genutzt.

Datei einzulesen:  
<http://www.zentut.com/c-tutorial/c-read-text-file/> [20.05.2018]

Dateipfade:  
<https://cboard.cprogramming.com/c-programming/88690-path-file-relative-program.html> [19.05.2018]

Standarteingaben:  
<https://www.c-plusplus.net/forum/39320-full> [22.05.2018]

Backtracking:  
<https://gist.github.com/bopbi/c9f94b1b981b84bcb02b> [15.05.2018]

## Literaturverzeichnis

[Abbildung 1 Kanban - Board 3](#_Toc511126566)

1. Wikipedia: Sudoku. Online unter   
   https://de.wikipedia.org/wiki/Sudoku. [10.04.2018] [↑](#footnote-ref-1)
2. CodeBlocks – Open Source IDE  
   <http://www.codeblocks.org/> [08.05.2018] [↑](#footnote-ref-2)
3. Sourcetree – Git GUI

   <https://www.sourcetreeapp.com/> [08.05.2018] [↑](#footnote-ref-3)
4. IT-Agile: Was ist Kanban? – Online unter:

   https://www.it-agile.de/wissen/einstieg-und-ueberblick/kanban/ [10. April 2018] [↑](#footnote-ref-4)