Sudoku - Projektdokumentation

**2018**

Jan Peter Hamm, Maksim Scheierman, Lena Willenbrock

10.04.2018



Inhalt

[1 Ausgangssituation 1](#_Toc511124507)

[1.1 Projektziele und Teilaufgaben 1](#_Toc511124508)

[1.2 Kundenanforderungen 1](#_Toc511124509)

[1.3 Projektumfeld 2](#_Toc511124510)

[1.4 Prozessschnittstellen 2](#_Toc511124511)

[2 Ressourcen und Ablaufplanung 3](#_Toc511124512)

[2.1 Projektmitglieder 3](#_Toc511124513)

[2.2 Aufgabenverteilung 3](#_Toc511124514)

[3 Durchführung und Auftragsbearbeitung 4](#_Toc511124515)

[3.1 Programmablaufplan 4](#_Toc511124516)

[3.2 Prozessschritte, Vorgehensweise, Qualitätssicherung 4](#_Toc511124517)

[3.3 Abweichungen, Anpassungen, Entscheidungen 4](#_Toc511124518)

[4 Projektergebnisse 5](#_Toc511124519)

[4.1 Soll-Ist-Vergleich 5](#_Toc511124520)

[4.2 Qualitätskontrolle 5](#_Toc511124521)

[4.3 Abweichungen, Anpassungen 5](#_Toc511124522)

[4.4 Fazit 5](#_Toc511124523)

[5 Anlagen 6](#_Toc511124524)

[5.1 Kundendokumentation 6](#_Toc511124525)

[5.2 Quellverzeichnis 6](#_Toc511124526)

[5.4 Abbildungsverzeichnis 6](#_Toc511124527)

# 1 Ausgangssituation

## Projektziele und Teilaufgaben

Das Ziel dieses Projektes ist es, ein fertiges und fehlerfreies Sudoku-Spiel zu erstellen. Der Kunde soll später fehlerfrei damit spielen können und das Sudokufeld fehlerfrei ausfüllen können.

## 1.2 Kundenanforderungen

Die folgenden Anforderungen sollen erfüllt werden. Dabei ist zwischen den Pflichtanforderungen (gekennzeichnet mit „Muss“) und Erweiterungen (gekennzeichnet mit „Kann“) für das Programm zu unterscheiden:

* Anzeige des Gitters (9x9) -> Muss

Programm zeigt auf dem Bildschirm ein Gitter aus 9x9 Feldern an, die mit den passenden Zahlen gefüllt werden

* Auswahl des aktuellen Feldes auf dem Spielfeld -> Muss

Spieler kann auswählen, welches Feld im Gitter bearbeitet werden soll

* Eingabe bzw. Veränderung des Wertes eines Feldes -> Muss

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 -> Muss

Programm reagiert nur auf gültige Eingaben des Benutzers.

* Auswahl eines zu spielenden Sudoku -> Muss

Spieler kann ein zu spielendes Sudoku auswählen. Eine Auswahl von Sudoku verschiedener Schwierigkeit kann fest vorgegeben werden. Alternativ können neue Sudoku automatisch erzeugt oder aus einer Datei eingelesen werden (siehe unten).

* Erkennung einer korrekten Lösung des Sudoku -> Muss

Programm erkennt automatisch oder nach Aufforderung ein korrekt geöstes Sudoku.

* Zeitmessung während des Spiels -> Muss

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.

* Automatische Erstellung neuer Sudoku -> Kann

Programm generiert automatisch neue (lösbare) Sudokurätsel verschiedener Schwierigkeit.

* Einlesen eines Sudoku aus einer Datei -> Kann

Programm liest ein neues Sudoku aus einer Datei ein.

* Abspeichern und Laden eines Spielstandes -> Kann

Spieler kann den aktuelle Spielstand in einer Datei abspeichern und zu einem späteren Zeitpunkt wieder laden, um das Spiel fortzusetzen.

* Lösungshinweise anzeigen -> Kann

Programm gibt auf Nachfrage einen gültigen Tipp zur Lösung des Rätsels.

## 1.3 Projektumfeld

Neben den Anforderungen haben wir außerdem noch einige Rahmenbedingungen bekommen.

Zum einen soll das Programm durch Funktionen und Dateien strukturiert werden.

Alle Dateien, Funktionen und wichtige Quellcodeabschnitte sollen kommentiert werden.

Das Programm soll in der Programmiersprache C mit Hilfe des Programms „CodeBlocks“ und dem damit einhergehenden „MinGW“- Compiler programmiert werden.

Es sollen keine Compiler- oder Linker- Warnungen bzw. –Fehler bei Abgabe vorhanden sein.

Die Benutzung von Bibliotheken ist Erlaubt.

Wenn man einen kleinen Codeabschnitt aus dem Internet zieht, muss die Quelle hinzugefügt werden.

Wir haben uns dazu entschlossen, einen Programmablaufplan zur grafischen Darstellung des Programmes zu erstellen. Außerdem wollen wir die Quellcodeverwaltung mittels Github erleichtern.

Das Abgabedatum ist der 22.05.2018.

## 1.4 Prozessschnittstellen

Als Ansprechpartner steht uns Herr Wichmann zur Verfügung.

# 2 Ressourcen und Ablaufplanung

## 2.1 Projektmitglieder

Unsere Projektgruppe besteht aus Jan Peter Hamm, Maksim Scheierman und Lena Willenbrock.

## 2.2 Aufgabenverteilung

Wir haben uns darauf geeinigt, dass Jan Peter Hamm und Maksim Scheierman den größeren Teil der Programmierung vornehmen. Lena Willenbrock wird zeitgleich an der Dokumentation arbeiten und ggf. bei der Programmierung unterstützen. Wir haben uns für diese Aufgabenverteilung entschieden, da es in unseren Augen am effektivsten erschien. Jeder kann so selbstständig arbeiten.

Wir haben uns in diesem Zuge für eine Quellcodeverwaltung mit GitHub entschieden, da man dort nach Einarbeitung relativ einfach den Quellcode Verwalten kann. Somit kann jeder auf die aktuelle Version des Quellcodes zugreifen.

# 3 Durchführung und Auftragsbearbeitung

## 3.1 Programmablaufplan

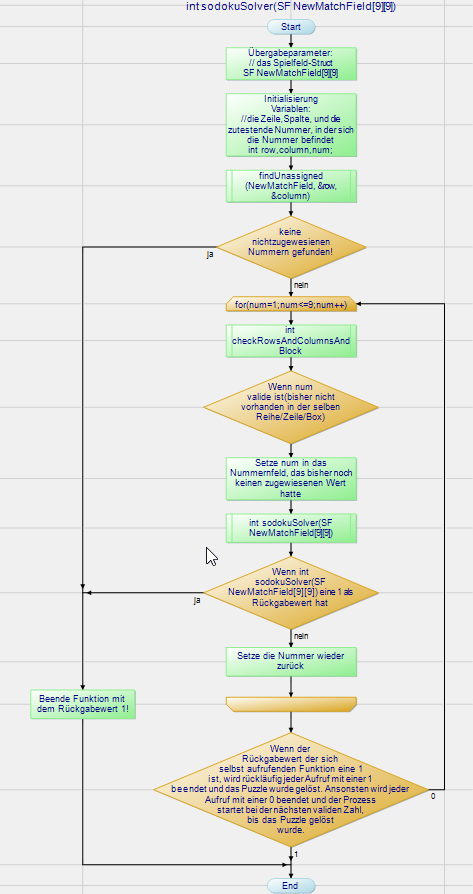
Wir haben uns entschieden nur die wichtigsten Kernfunktionen als PAP darzustellen.

Dazu haben wir das Programm PapDesigner verwendet. Die Abbildungen der Paps können ebenfalls mit dem genannten Programm geöffnet werden, die Dateien befinden sich im Dokumentationsordner „Paps“ in der Projektmappe.

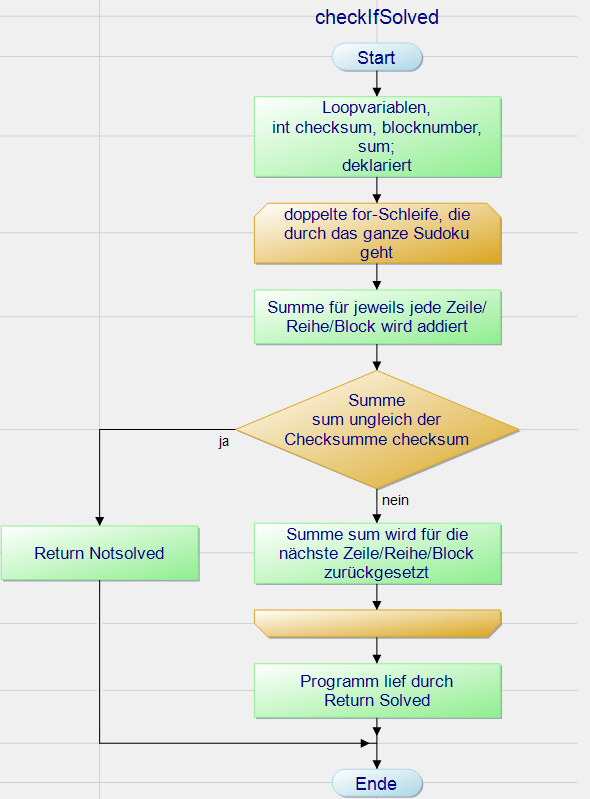
Unsere 5 wichtigsten Funktionen:

1. Speichern,
2. Laden des Sudokus
3. Lösen des Sudokus
4. Überprüfung der Lösung
5. Spielloop

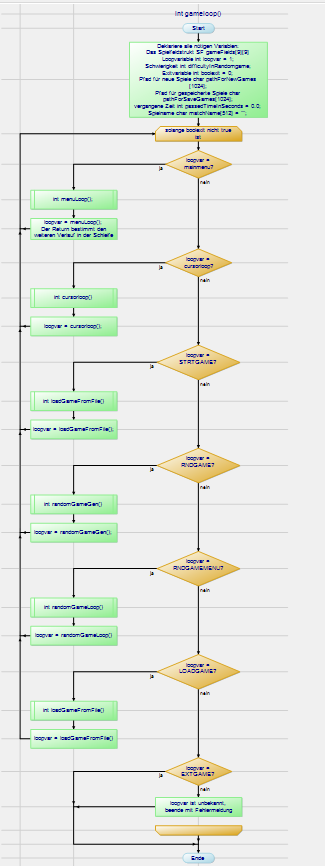
Die Funktion int sodokuSolver(SF NewMatchField[9][9]) sorgt dafür, dass das ein beliebiges unfertiges Sudoku gelöst wird.(Unterfunktionen als PAP in der Datei)



Die Funktion checkIFSolved, überprüft ob das Sudoku gelöst ist.



Die Funktion gameLoop() ist die Funktion die dafür sorgt, dass der Benutzer im Spiel bleibt und abhängig von seiner Auswahl sich der nächste Loop öffnet.



## 3.2 Prozessschritte, Vorgehensweise, Qualitätssicherung

Begonnen wurde zuerst mit der grafischen Darstellung des Sudokus. Es war die generateField-Funktion. Parallel dazu wurde die Navigation in dem Sudoku-Feld in der Funktion cursorLoop entwickelt. Nachdem der Grundbaustein geschaffen wurde, waren die nächsten Schritte das Entwickeln der Ladefunktion und aufbauend auf der cursorLoop-Funktion die Navigation in den Menüs.

Nach einem Abgleich und Austausch konnten wir die Funktionen im Team verwenden.

Als nächstes wurden noch einige Menügraphiken erstellt und damit alle benötigten Menüs fertiggestellt.

Nachdem die Lade-Funktionen implementiert wurde, konnte einerseits mit der Entwicklung des Überprüfungsalgorithmus und andrerseits die Arbeit an der Speicher-Funktion begonnen werden.

Im letzten großen Sprint wurde dann noch der Überprüfung- und der Lösungshinweisalgorithmus fertiggestellt.

## 3.3 Abweichungen, Anpassungen, Entscheidungen

Eine wichtige Entscheidung war es, dass wir für viele Sachen als Rückgabewert Defines festlegen, damit wir aussagekräftige Rückgabewerte besitzen. Besonders im gameLoop war die Übersicht um einiges klarer als dort die definierten Werte benutzt wurden.

# 4 Projektergebnisse

## 4.1 Soll-Ist-Vergleich

dkdtuk

## 4.2 Qualitätskontrolle

fzkdtj

## 4.3 Abweichungen, Anpassungen

gjfcz

## 4.4 Fazit

Djht

# 5 Anlagen

## 5.1 Kundendokumentation

## 5.2 Quellverzeichnis

## 5.4 Abbildungsverzeichnis

|  |  |  |
| --- | --- | --- |
| Nummer der Abbildung | Name | Kapitel |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |