



# BESONDERE LERNLEISTUNG INFORMATIK

Der CaRP-Assigner

Fabius Mettner  
Fabius1705@live.de

## Inhaltsverzeichnis

1. Einleitung .....	2
1.1. Motivation.....	2
1.2. Vorgehensweise.....	4
2. Der Entwicklungsprozess .....	5
2.1. Die Neustrukturierung .....	5
2.2. Die Ausprogrammierung.....	6
2.3. Die größten Fehler und Bugs .....	7
2.3.1. Das Verschwinden der Projektkurs-Schüler .....	7
2.3.2. Fehler beim Sortieren der Berechnungen nach der Güte .....	8
3. Bedienungshandbuch .....	9
3.1. Importieren .....	9
3.1.1. Kriterien, die eine Eingabedatei erfüllen muss .....	10
3.2. Der Verteilungsprozess .....	11
3.3. Das Exportieren der Daten .....	11
3.4. Das Bearbeiten der Eingabedaten im CaRP-Assigner.....	12
3.5. Das Bearbeiten der Verteilung im CaRP-Assigner.....	13
3.6. Die Einstellungen .....	14

## 1. Einleitung

Der Course and Research Paper Assinger (zu Deutsch: Kurs und Facharbeit Zuweiser (KuFa-Zuweiser)) ist ein Programm, zum Zuweisen von Schülern nach Wunschkursen, welche nach einer Priorität geordnet sind. Dabei wird versucht eine Zuweisung zu erstellen, welche einer Verteilung mit der bestmöglichen Priorität entspricht und dabei das Limit eines jeden Kurses nicht überschreitet.

### 1.1. Motivation

Betrachtet man die alte Version, den KuFa-Zuweisers (Abbildung 1), so entdeckt man zunächst eine im Swing-Format angelegte Benutzeroberfläche. Schnell ist auch zu erkennen, dass diese graphische Oberfläche (GUI) sich während des Berechnungsprozess aufhängt, wodurch ein

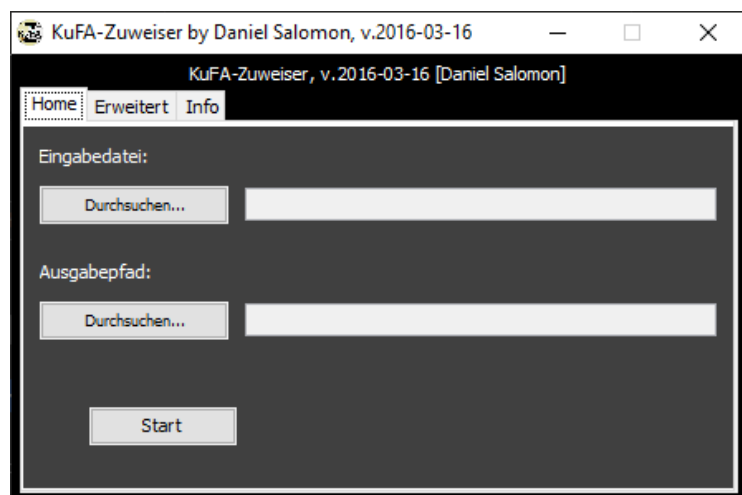


Abbildung 1: Alte KuFA-Anwendung

Beobachten des Fortschrittes nicht mehr möglich ist. Beides hat die Grundfunktionen der Applikation nicht beeinträchtigt, konnte aber beim Anwenden der Applikation als störend empfunden werden. Aus diesem Grund ließ sich zunächst eine vollständige Überarbeitung der graphischen Oberfläche als Ziel setzen. Diese sollte zum einen Anwenderfreundlicher, variabler und moderner werden.

Bei der ersten Betrachtung des hinter stehenden Quellcodes ist bereits der nächste Verbesserungspunkt zu erkennen. Der alte KuFa-Zuweiser verfügt lediglich über wenige besonders lange Klassen, die die einzelnen Funktionen grob unterteilen. Um eine übersichtliche Struktur in das Projekt zu bringen wurde nun mit sogenannten „packets“, zu Deutsch Paketen, gearbeitet, die das Projekt hierarchisch strukturieren sollten. Über diesen Weg wurden die einzelnen Funktionen leichter

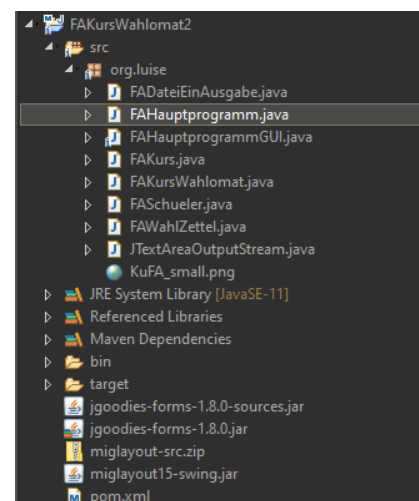


Abbildung 2: Klassenstruktur des alten KuFA-Zuweisers

unterscheidbar gemacht. Gleichzeitig bietet die in Abbildung 3 zu erkennende Struktur einen weiteren Vorteil: Jeder Bereich kann einfacher getrennt werden und so in einem anderen Programm einfach wiederverwertet werden. Gleichzeitig erleichtert es auch das Testen der einzelnen Funktionen, durch den Fokus auf die unterschiedlichen Funktionen.

In Folge dieser Neustrukturierung ist der gesamte Vorangegangene Quellcode aus mehreren Gründen verworfen worden:

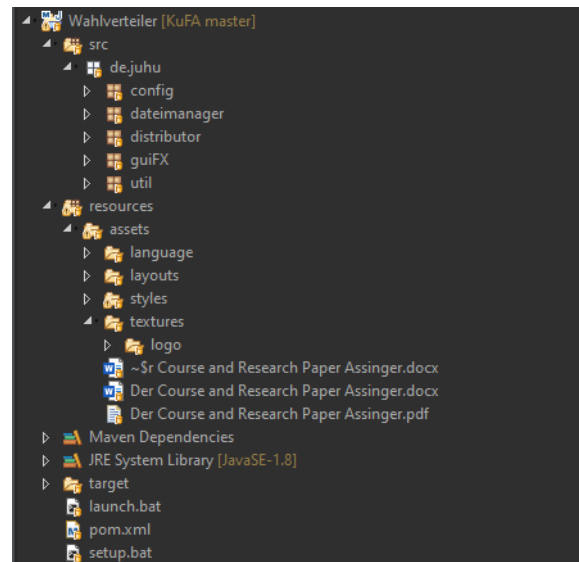


Abbildung 3: Klassenstruktur des CaRP-Assigners

1. Zum einen war es nicht möglich mit ihm die einzigen Funktionen entsprechend zu trennen.
2. Teilweise erschien der Quellcode sehr unübersichtlich und das Einarbeiten in diesen hätte ähnlich viel Zeit in Anspruch genommen, wie das strukturierte Neuerstellen des Quellcodes.
3. Durch diese komplette Neuauflage des Projektes war es zudem möglich einfacher neue Funktionen zu implementieren.

Im Verlauf des Erstellens der besonderen Lernleistung ist zudem ein neuer Punkt hinzugekommen. Auf der Suche nach einer neuen Herausforderung fällt

CapS Assign (1.1.7)

☒ Aufgaben ☒ Musterklausuren ☒ HfA  
☒ Aufgabenkategorie ☒ Klausuren ☒ 16 Verabreichung ☒ Einordnungen ☒ Log

**Klausur**

Vorname		Nachname		Kurs 1		Kurs 2		Kurs 3	
		Fach	Lehrer	Fach	Lehrer	Fach	Lehrer	Fach	Lehrer
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	D	C	D	F	F		
Max	Mustermann	A	C	D	D	F	F		
Max	Mustermann	QPA1	-	-	-	-	-		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	QPA2	-	-	-	-	-		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	QPA3	-	-	-	-	-		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	QPA2	-	-	-	-	-		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	QPA3	-	-	-	-	-		
Max	Mustermann	A	S	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	A	J	C	D	F	F		
Max	Mustermann	QPA3	-	-	-	-	-		
Max	Mustermann	A	S	C	D	F	F		

☒ Technische Verabreichung ☒ Spätklausur-Verabreichung ☒ Besondere und Spätklausur-Verabreichung

Abbildung 4: Eingabevorschau der Schüler mit Musterdaten

einem ins Auge, dass eine Voransicht der Daten im Programm, sowohl vor dem Zuweisungsprozess als auch nach diesem, für den Endnutzer von Bedeutung sein kann. So ist eine Vorschau für die Eingabe, als auch für die Ausgabe eingefügt worden. Um die flexible Nutzung des Programms hierbei zu steigern wurde im Anschluss noch das Bearbeiten der Daten in den Voransichten implementiert.

## 1.2. Vorgehensweise

Mit diesen Verbesserungspunkten habe ich mich an die Erstellung eines komplett neuen Kurs- und Facharbeit Zuweisers gesetzt. Zunächst einmal mussten die bereits vorhandenen Funktionen des alten Programms gesichtet und neu implementiert werden, um im Folgenden dann nach neuen Ansätzen zur Überarbeitung des Programmes und neuen zu implementierenden Funktionen suchen zu können. Nach einiger Überlegungszeit und vielem ausprobieren wurde das alte Programm zur Seite gelegt und ein neues Java-Projekt in Eclipse erstellt.

Hier wurden schon die ersten großen Änderungen im Grundaufbau vorgenommen. Anstatt der Flachen Hierarchie des Klassenbaumes wurde nun auf einen hierarchisch strukturierten Klassenbaumaufbau, wie er auch in Abbildung 3 zu sehen ist, gesetzt. Dies ermöglicht nicht nur einen einfacheren Überblick über die einzelnen Bereiche, sondern erweist sich auch als sinnvoll, wenn man nur Teile des Programmes verändern, oder in eine neue Umgebung (beispielsweise in ein neues Projekt) überführen will.

Des Weiteren wurde nun auch die Einbindung von Libraries erneuert. Statt die einzelnen benötigten Libraries manuell hinzuzufügen wurde hier auf das Build Management Tool „Maven“ gesetzt. Neben einer vereinfachten Möglichkeit zum Exportieren des Projektes in das „.exe“ und „.jar“-Format ist es so auch möglich geworden einfacher neue Referenzen, sogenannte „Dependencies“, im Programm einzufügen. Diese Referenzen werden von Maven selbstständig verwaltet und es ist nicht mehr nötig manuell Libraries in das Projekt zu importieren.

Auch bei der graphischen Oberfläche wurden einige grundlegende Veränderungen vorgenommen. Die genutzte „Swing“ Oberfläche wurde verworfen und durch eine neue „JavaFX“ Oberfläche ersetzt. Diese gibt der Anwendung nicht nur einen „modernerer Look“, sondern ermöglichte auch das erleichterte Einbinden neuer Funktionen. Um eine passende Oberfläche zu schaffen, wurde mit dem für diesen Zweck entwickelten

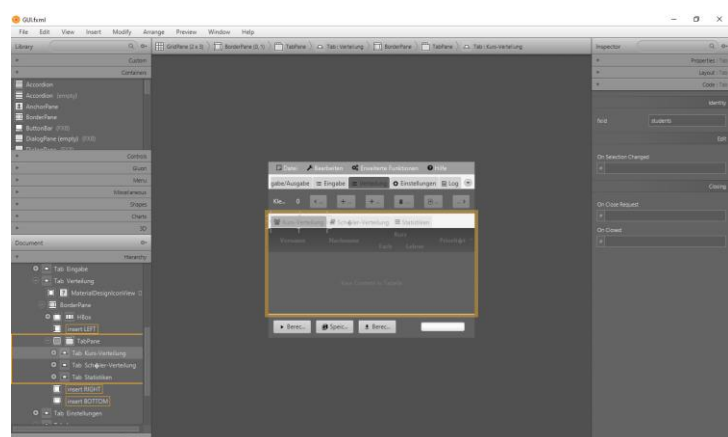


Abbildung 5: Ansicht des GUIs im „Scene Builder“

sondern ermöglichte auch das erleichterte Einbinden neuer Funktionen. Um eine passende Oberfläche zu schaffen, wurde mit dem für diesen Zweck entwickelten

„Scene Builder“ gearbeitet, welcher die Erstellung neuer Benutzeroberflächen durch eine bearbeitbare Voransicht weitgehend erleichtert.

Im weiteren Entwicklungsprozess wurden neben neuen Features auch weitere Referenzen und Designs eingefügt. So ist das Wechseln zwischen einem hellen und einem dunklen Oberflächendesign, sowie das Wechseln der Sprache zwischen Deutsch und Englisch ermöglicht worden. Dies konnte durch das Einbinden von „css“, zum Verändern des Oberflächendesigns, sowie durch das Einbinden von Sprachdateien, über sogenannte „Internationalized Strings“, ermöglicht werden.

## 2. Der Entwicklungsprozess

Nachdem die grundlegenden Ideen strukturiert und der Aufbau des Projekts geklärt worden war, konnte nun der eigentliche Entwicklungsprozess gestartet werden. Hierbei wurde neben stundenlanger Arbeit des Testens und

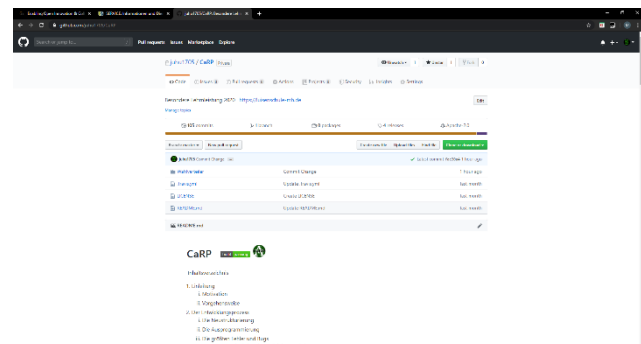


Abbildung 6: „Github Repository“ des CaRP-Assigners

Verbesserns auch viel Zeit in die Implementierung der einzelnen Ideen mithilfe der vorangegangenen Vorgehensweisen investiert. Um einen Datenverlust, oder einen andersgearteten Rückschlag zu vermeiden wurde nun ein „GitHub-Repository“ erstellt, auf dem die letzte funktionierende Version stets hinterlegt werden konnte.

## 2.1. Die Neustrukturierung

Zuerst ist durch die neu aufgesetzte Struktur auch eine vollständige Erneuerung des alten „KuFA-Zuweisers“ geschehen. Dabei wurde folgende Baumpfade erstellt: Unter dem Pfad „distributor“ ist alles für den Zuweisungsprozess Relevante implementiert, während das Einlesen und Schreiben von Dateien in den Reiter „dateimanager“ eingefügt wurde. Der Pfad „config“ enthält alle relevanten Dateien zum Erstellen der Konfiguration und unter „guiFX“ wurden die Klassen der graphischen Oberfläche gesammelt. Jegliche weiteren Klassen wurden schließlich im Reiter „util“ zusammengefasst. Um eine weitreichende Struktur zu wahren wurde zudem ein eigener Ordner für alle nicht Java-Dateien erstellt. Hier finden sich neben der „Hilfe“ Datei auch die „Stylesheets“ (css Dateien zum Bearbeiten der Ansicht der graphischen Oberfläche) und die Sprachdateien, sowie die „fxml“ Dateien, welche die Struktur der einzelnen GUIs beinhalten.

## 2.2. Die Ausprogrammierung

Begonnen hat die Ausprogrammierung zunächst mit dem Im- und Exporter für Dateien. Im gleichen Zug ist auch die Klasse „References“ mit einem den Umständen angepassten Logger entstanden. Um die Exportdatei graphisch aufzuwerten ist die Klasse „CellStyles“ entstanden, welche eine kleine Auswahl an unterschiedlichen Aussehmöglichkeiten für die Tabellenzellen von Excel bereitstellt.

Nachfolgend wurden der Berechnungsprozess und alle zugehörigen Klassen in einem Schwung implementiert. Um diese Funktionen zu Testen wurde zunächst ein einfaches GUI auf „Swing“-Basis benutzt, bevor schließlich im nächsten Schritt unter zu Hilfenahme des „SceneBuilders“ ein JavaFX GUI aufgesetzt wurde. Nach erstellen des neuen GUIs kamen nun vor allem in dem Bereich der Berechnung und in der graphischen Darstellung immer wieder kleinere und größere Neuerungen hinzu. So entstand auch die im Benutzerverzeichnis hinterlegte speicherbare Konfiguration, der speicherbare Log und die Möglichkeit eine einlesbare „carp“-Datei zu exportieren. Über einen längeren experimentellen Zeitraum mit vielen Änderungen im graphischen Design entstanden auch die neuen Vorschauen für die Berechnung und die eingelesenen Daten. Hier wurde die Möglichkeit zur Bearbeitung der Daten mit allen notwendigen Funktionen immer weiter optimiert. Schließlich wurden die Schortcuts, sowie die Icons in das Programm eingeführt und ein verbessertes „Über“ Fenster erstellt. Zuletzt wurde, die in Abbildung 7 zu erkennende, Darstellung der Einstellungen überarbeitet und damit war das eigentliche Programmierprozess fertig und es mussten nur noch bestehende Bugs behoben und Klassen ausdokumentiert werden.

	A	B	C	D	E
1	Name	Vorname	Kurs	Lehrer	Priorität
2					
3	Mustermann	Max	C	D	2
4	Mustermann	Max	C	D	2
5	Mustermann	Max	@PJK G		1
6	Mustermann	Max	C	D	2
7	Mustermann	Max	@PJK G		1
8	Mustermann	Max	C	D	2
9	Mustermann	Max	C	D	2
10	Mustermann	Max	C	D	2
11	Mustermann	Max	C	D	2
12	Mustermann	Max	@PJK G		1
13	Mustermann	Max	C	D	2
14	Mustermann	Max	C	D	2
15	Mustermann	Max	C	D	2
16	Mustermann	Max	C	D	2
17	Mustermann	Max	C	D	2
18	Mustermann	Max	C	D	2
19	Mustermann	Max	C	D	2
20	Mustermann	Max	C	D	2
21	Mustermann	Max	C	D	2
22	Mustermann	Max	C	D	2
23	Mustermann	Max	C	D	2
24	Mustermann	Max	C	D	2
25	Mustermann	Max	C	D	2
26	Mustermann	Max	C	D	2
27	Mustermann	Max	C	D	2
28	Mustermann	Max	C	D	2
29	Mustermann	Max	C	D	2
30	Mustermann	Max	C	D	2
31	Mustermann	Max	A	B	1
32	Mustermann	Max	A	B	1
33	Mustermann	Max	A	B	1
34	Mustermann	Max	A	B	1
35	Mustermann	Max	@PJK G		1
36	Mustermann	Max	A	B	1
37	Mustermann	Max	A	B	1
38	Mustermann	Max	A	B	1

Abbildung 8: Exporttabelle der Kurs-Verteilung mit Beispieldaten

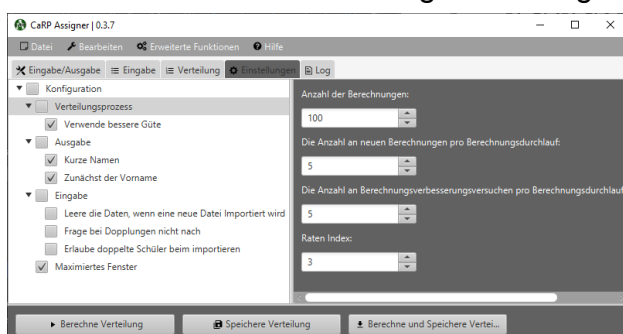


Abbildung 7: Überarbeitete Einstellungsansicht

## 2.3. Die größten Fehler und Bugs

Über die Zeit haben sich in den Programmcode immer wieder kleinere und größere Fehler eingeschlichen. Während einige Fehler sofort auffallen, ist bei anderen ein genaues Verfolgen der vorgehenden Prozesse von Nöten. Sobald ein solcher Fehler auffällt ist es an der Zeit diesen im Code selbst ausfindig zu machen und zu beheben. Da viele Fehler auf kleinen Programmabschnitten beruhen, die dann im Verlauf zu weitreichenden Problemen führen, ist zunächst ein kleinschrittiges Auffinden der Fehlerquelle erforderlich. Mithilfe von hunderten „Print“-Messages und unter Nutzung des in Eclipse verfügbaren Debug-Modus, können so viele der Fehlerquellen identifiziert werden. Nach dem Auffinden der Fehlerquelle ist das eigentliche Fehlerbeheben nicht mehr sonderlich schwer, da die meisten Fehler lediglich kleinen logischen Aussetzern, oder dem Übersehen von Zusammenhängen zu verdanken sind. Bei komplexeren Fehlern war es sinnvoll, testweise die Wirkung einer Veränderung einzelner Programmabschnitte zu versuchen, um so schließlich doch den Fehler zu beheben. Im Folgenden sind einige dieser Fehler Beispielhaft aufgeführt.

### 2.3.1. Das Verschwinden der Projektkurs-Schüler

Dieser erste Fehler hat sehr viel Zeit beim Beheben gekostet, ist aber, wie man im Folgenden sehen wird, relativ banal gewesen. Aufgefallen ist er dadurch, dass die Zahl der Projektkursschüler sich durch den Berechnungsprozess verkleinert hat. Immer wieder habe ich den Fehler bei den unterschiedlichen Kopiervorgängen gesucht, bis ich ihn schließlich in folgender for-Schleife gefunden habe:

```
For (int i = 0; i < this.allStudents.size(); i++) {  
    If ((this.allStudents.get(i).getActiveCourse() != null &&  
        this.allStudents.get(i).getActiveCourse().equals(this.ignore())) ||  
        this.allStudents.get(i).getCoursesAsList().contains(this.ignore()))  
        this.ignoredStudents.add(this.allStudents.remove(i));  
}
```

Wie zu sehen ist, wird jedes Mal beim Durchlaufen der Schleife der Index *i* eine Position weitergezählt. Wird nun ein Schüler aus der Liste *allStudents* entfernt, rücken die nachfolgenden Schüler eine Position auf. Steht nun zwei Projektkurs Schüler hintereinander in der Liste, so wird der erste erkannt und richtig verschoben, der dahinterstehende Schüler rückt dadurch einen Listenplatz auf, das heißt, er nimmt nun den gerade behandelten Index an. Da der Index nun aber eine Position weitergezählt wird, kann dieser Schüler nicht behandelt werden und kann nicht mehr in die Liste



„ignoredStudents“ eingefügt werden. Was ihn aus dem Verteilungsprozess schließlich löscht.

Beheben kann man diesen Fehler, in dem man den Index nur erhöht, wenn die if-Bedingung nicht zutrifft. Also ist die Lösung dieses Fehlers folgende:

```
For (int i = 0; i < this.allStudents.size();) {  
    If ((this.allStudents.get(i).getActiveCourse() != null &&  
        this.allStudents.get(i).getActiveCourse().equals(this.ignore()))  
        ||  
        this.allStudents.get(i).getCoursesAsList().contains(this.ignore()))  
        this.ignoredStudents.add(this.allStudents.remove(i));  
    else  
        i++;  
}
```

### 2.3.2. Fehler beim Sortieren der Berechnungen nach der Güte

Dieser zweite Fehler den ich erläutern will entstand durch das falsche Interpretieren von den Rückgabewerten der „compareTo(Object o) Methode. Durch Ausprobieren habe ich diesen Fehler schließlich behoben. Da ich bei diesem Fehler den Ursprung kannte, nämlich die selbst geschriebene „Save.compareTo(Save s)“ Methode, konnte ich durch Ausprobieren relativ unkompliziert eine Lösung finden. Hierzu habe ich jeweils das Vorzeichen des zurückgegebenen Wertes vertauscht und if-Bedingungen für Randfälle eingeführt, um so die Speicher in der richtigen Reihenfolge anordnen zu können. Dabei bewirkte der entsprechende Vorzeichenwechsel, entweder, dass die beste, oder schlechteste Berechnung vorne in der Liste der Speicher stand und zuerst angezeigt wurde. Dieser Fehler wird hier vor allem aus dem Grund aufgeführt, weil er durch zwischenzeitliches Umstellen von den Berechnungen der Güte und früherer Berücksichtigter Werte immer wieder auftrat.

### 3. Bedienungshandbuch

Um den Einstieg in die Nutzung des CaRP-Assigners zu erleichtern werde ich im Folgenden die wichtigsten Funktionen erklären und Schritt für Schritt den Umgang mit dem Programm erläutern.

#### 3.1. Importieren

Gestartet wird zunächst mit dem Import einer Datei in den CaRP-Assigner. Der CaRP-Assigner unterstützt sowohl .csv, .xls und .xlsx Dateien. Um eine Datei zu laden begeben sie sich in den Reiter „Eingabe/Ausgabe“. Das Importieren einer Datei kann nun über drei Unterschiedliche Wege gestaltet werden:

- Über Drag and Drop: Nehmen sie ihre Datei und ziehen sie diese über das weiße Textfeld neben dem oberen „Suche“ Knopf. Wenn sie die Datei nun loslassen, wird sie automatisch in den Assigner geladen.
- Über die „Suche“ Funktion: Betätigen sie den oberen „Suche“ Knopf und wählen sie ihre Datei aus dem Dateisystem aus. Mit „Öffnen“ können sie Datei in das Programm laden.
- Sie können auch über den Menü-Reiter „Datei“ und dann über die Funktion „Suche Eingabedatei“ den Dateieexplorer zum Importieren der gewünschten Tabellendatei aufrufen.

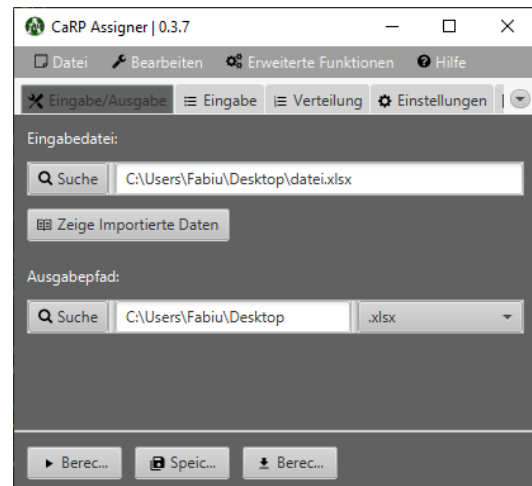


Abbildung 9: Der Reiter „Eingabe/Ausgabe“

Nachdem sie ihre Datei importiert haben, können sie sich über den Knopf „Zeige Importierte Daten“, die von dem Programm eingelesenen Daten anzeigen lassen.

**Hinweis:** Es ist möglich Daten zu bereits importierten Daten zu importieren. Sollten sich bei diesem Vorgang Daten doppeln, wird ihnen eine Warnung angezeigt. Die Optionen in diesem Warnfenster können auch in der Konfiguration geändert werden. Die Warnung wird nach jedem Neustart jedoch wieder angezeigt werden!

### 3.1.1. Kriterien, die eine Eingabedatei erfüllen muss

Damit eine Datei von dem neuen CaRP-Assigner eingelesen werden kann muss sie folgende Kriterien erfüllen:

- Zum Einfügen eines Schülers
  - Die Datei muss die unter den Konfigurationseinstellungen zu findende Config „Schüler Kennzeichnung“ beinhalten, um einen Schüler hinzuzufügen. Diese „Schüler Kennzeichnung“ kann entweder in der Spalte „A“ vor den Angaben zum Schüler stehen, sodass dessen Werte erst in der darauffolgenden Spalte „B“ beginnen, oder sie kann als Tabellen Benennung angegeben werden. Dann steht das der Nachname des Schülers schon in der Spalte „A“.
  - Dabei ist die maximale Kursanzahl zu beachten. Der Schüler wird wie folgt in der Tabelle aufgeschrieben: {Nachname, Vorname, Kurs 1: Fach, Kurs 1: Lehrer, Kurs 2: Fach, Kurs 2: Lehrer, ..., Kurs („maximale Kursanzahl“): Fach, Kurs („maximale Kursanzahl“): Lehrer}
  - Wenn mehr als „maximale Kursanzahl“ Fächer angegeben werden, werden sie auch eingelesen, aber sie sind in der Berechnung nicht verfügbar. Daher ist es zu empfehlen, dass die „maximale Kursanzahl“ auf die Kursanzahl des Schülers mit den meisten Kursangaben gesetzt wird!
- Zum Einfügen eines Kurses:
  - Der Kurs kann beim Schüler mit angegeben sein, wodurch er mit dem Schülermaximum aus der Config Datei hinzugefügt wird.
  - Die Datei muss die unter den Konfigurationseinstellungen zu findende Config „Kurs Kennzeichnung“ beinhalten, um einen Schüler hinzuzufügen. Diese „Kurs Kennzeichnung“ kann entweder in der Spalte „A“ vor den Angaben zum Kurs stehen, sodass dessen Werte erst in der darauffolgenden Spalte „B“ beginnen, oder sie kann als Tabellen Benennung angegeben werden. Dann steht beginnen die Angaben zum Kurs in Spalte „A“.
  - Der Kurs muss wie folgt in der Tabelle stehen: {Fach, Lehrer, Maximale Schülerzahl}

- Um einen Kommentar zu kennzeichnen muss in der Spalte „A“, als erstes das Zeichen aus der Konfiguration, welches unter „Kommentar Kennzeichnung“ zu finden ist, angegeben werden.

**Hinweis:** Alle Einstellungen, die hier erwähnt wurden, sind unter „Konfiguration>Eingabe“ zu finden.

### 3.2. Der Verteilungsprozess

Um eine Verteilung zu berechnen müssen bereits Daten eingelesen worden sein. Sollte dass der Fall sein, reicht ein Einfaches betätigen des Knopfes „Berechne Verteilung“ aus, um den Verteilungsprozess zu starten. Den Knopf

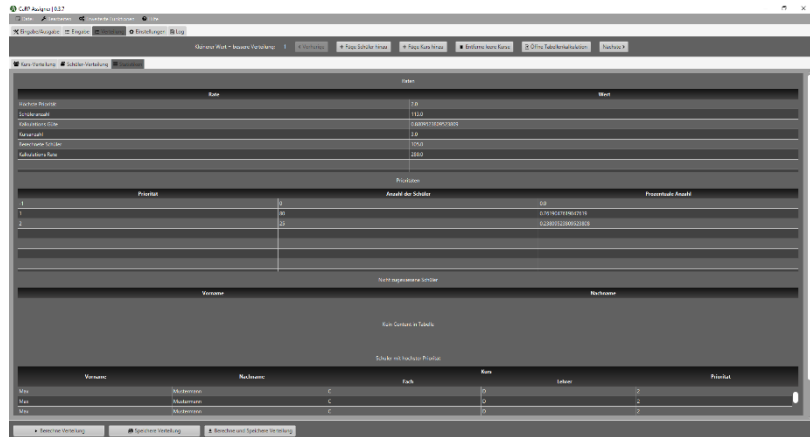


Abbildung 10: Vorschau der Statistik einer Verteilung

„Berechne Verteilung“ finden sie unten links in jedem Fenster. Wenn unten rechts eine Prozessanzeige erscheint, welche ihnen anzeigt, wie weit das Programm mit dem Verteilungsprozess ist, haben sie den Prozess erfolgreich gestartet. Sobald die Zuweisung abgeschlossen ist, wird eine Vorschau der Daten angezeigt.

### 3.3. Das Exportieren der Daten

Das Exportieren ist wieder, sowohl nach .xls, .xlsx oder .csv Datei möglich. Um den Ort auszuwählen, an den sie ihre Daten exportiert haben wollen, gehen sie wieder in den Reiter „Eingabe/Ausgabe“ aus Abbildung 1. Hier betätigen sie den unteren „Suchen“ Knopf, um einen Ausgabepfad auszuwählen. Wählen sie nun den gewünschten Ausgabeordner und bestätigen sie mit „Ordner Auswählen“. Um den Dateityp festzulegen, in den Exportiert wird, kann der rechts in der gleichen Reihe zu findendem Knopf verwendet werden, mit dem zwischen den drei Möglichkeiten gewählt werden kann. Die Ausgewählte Möglichkeit wird dann als Text des Knopfes angezeigt. Nun kann die Verteilung über den Knopf „Speichere Verteilung“ gespeichert werden. In dem gewünschten Ordner finden sie nun zum einem eine „.log“-Datei, in der sich der Log befindet, dann eine „.carp“-Datei, die sie mit dem KuFA-Zuweiser wieder öffnen können, und zuletzt die exportierte Datei im gewünschten Format.

### 3.4. Das Bearbeiten der Eingabedaten im CaRP-Assigner

Unter dem Reiter Eingabe finden sie eine Vorschau alle der von Ihnen importierten Daten. Um diese Daten zu bearbeiten, wählen sie einfach den Eintrag in der Liste der Schüler oder der Liste der Kurse aus, den sie gerne bearbeiten möchten. Mit einem Rechtsklick werden Ihnen nun die Bearbeitungsfunktionen angezeigt. Alternativ finden Sie im Menü unter dem Reiter „Bearbeiten“ ebenfalls für den

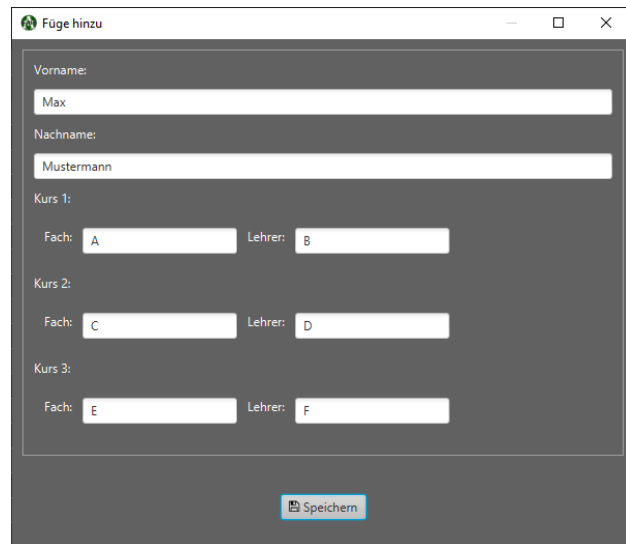


Abbildung 11: Bearbeitungsfenster für einen Schüler mit Daten

ausgewählten Kurs oder Schüler die folgenden Bearbeitungsfunktionen:

1. „Füge hinzu“: Mit dieser Funktion kann ein neuer Schüler oder Kurs in das Programm eingefügt werden. Dabei erscheint eine grafische Oberfläche, in die Sie die gewünschten Daten eintragen können. Wenn Sie fertig sind, betätigen Sie einfach den Knopf „Speichern“ und der neue Schüler wird automatisch eingefügt.
2. „Bearbeite“: Mit dieser Funktion ist es möglich einen Schüler oder Kurs zu bearbeiten. Hier erscheint dasselbe GUI wie beim Einfügen des gewünschten Schülers oder Kurses. Die Felder sind nun dem Kurs oder Schüler entsprechend ausgefüllt und können daher beliebig verändert werden. Indem Sie „Speichern“ drücken, werden die Daten überschrieben.
3. „Entferne“: Mit dieser Funktion ist es möglich den aktuell ausgewählten Kurs oder Schüler zu löschen.

**Hinweis:** Groß und Kleinschreibung wird bei den Kursen nicht berücksichtigt.

### 3.5. Das Bearbeiten der Verteilung im CaRP-Assigner

Auch an den Verteilten Daten können noch Veränderungen vorgenommen werden. Dazu kann der Schüler, oder Kurs, der bearbeitet werden soll, ausgewählt werden und dann die gewünschte Aktion ausgeführt werden:

Fach	Lehrer	Schüler 1		Schüler 2		Schüler 3	
		Vorname	Nachname	Vorname	Nachname	Vorname	Nachname
B	A	Max	Mustermann	Max	Mustermann	Max	Mustermann
D	C	Max	Mustermann	Max	Mustermann	Max	Mustermann
F	E	Max	Mustermann	Max	Mustermann	Max	Mustermann

Abbildung 12: Fenster zum Bearbeiten eines Schülers in einer Verteilung

1. „Bearbeite“: Ermöglicht es bei Schülern diesen in einen anderen seiner Gewählten Kurse zu setzen oder seinen Namen zu verändern und bei Kursen deren Namen zu verändern. Dazu erscheint, wie auch beim Bearbeiten der Eingabedaten eine graphische Oberfläche, an der die Veränderungen vorgenommen werden können.
2. „Entferne“: Entfernt den Schüler oder Kurs aus der Berechnung. Dabei können lediglich leere Kurse entfernt werden!

Um einen Kurs oder Schüler hinzuzufügen können die über den Tabellen zu findenden Knöpfe „Füge Schüler hinzu“ und „Füge Kurs hinzu“ genutzt werden.

Da bei einem Verteilungsprozess mehrere Verteilungen erstellt werden ist es mit Hilfe der Knöpfe „Vorherige“ und „Nächste“ auch möglich zwischen den besten fünf Verteilungen zu wechseln.

Um nur die Kurse mit Schülern in der Verteilung zu haben können auch direkt alle Leeren Kurse über den Knopf „Entferne leere Kurse“ gelöscht werden.

Wenn sie eine Vorschau der Datei in Excel sehen möchten, ist dies über den Knopf „Öffne Tabellenkalkulation“ möglich.

### 3.6. Die Einstellungen

Abschließend wird nun der Reiter „Einstellungen“ erklärt. Hier sind alle Konfigurationsmöglichkeiten aufzufinden. Dabei können sie im Baum (links), die gewünschten Pfade ausgewählt werden wodurch auf der rechten Hälfte die

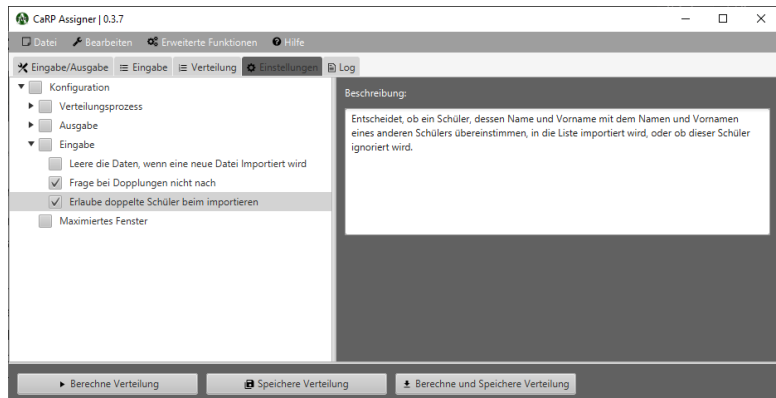


Abbildung 13: Konfigurationsfenster mit rechts eingeblendetem Beschreibungstext für die links ausgewählte Einstellung

zugeordneten Konfigurationsmöglichkeiten angezeigt werden. Einstellungen, die nur an- und ausgeschaltet werden können, sind im Baum mitaufgeführt. Werden diese Dort angeklickt, erscheint rechts eine Beschreibung über deren Funktion. Allen anderen Einstellungen sind Hovertexte, also Texte, die beim Drüberfahren mit der Maus erscheinen, beigefügt, die deren Funktion erläutern.