**Aufzugssteuerung - Ausarbeitung**

von: Maryam Baabaa, Leon Schinnen, Lasse Sönksen

Patrick Stowasser, Sebastian Dengler, Sebastian Klose

￼

￼

￼

￼

￼

# **Beschreibung des Problems**

Gesucht ist ein objektorientiertes Java Programm, welches verschiedene Arten von Aufzügen steuern können soll. Dabei hat jeder dieser Aufzüge besondere Eigenschaften und Voraussetzungen. Das Projekt soll mindestens einen Aspekt der Realität in einer Klassen-Hierarchie sichtbar machen. Das gegebene "Gebäude", in welchem unser Programm agieren soll, hat 100 Stockwerke und insgesamt 45 Aufzüge.

Diese Aufzüge sind unterteilt in fünf Kategorien mit speziellen Attributen unterteilt:

1. Der **kleine Personenaufzug** besitzt eine maximal zulässige Personenanzahl von 15, mit einem zulässigen Gesamtgewicht von 1200 kg. Von diesen gibt es 20 Exemplare, wobei das besondere Attribut dieser Klasse die Aufzugsmelodie ist.

2. Der **große Personenaufzug** im Vergleich hat eine maximale Personenanzahl von 30 und ein zulässiges Maximalgewicht von 2400 kg. Von diesem gibt es 10 Exemplare, genau wie bei dem kleinen Personenaufzug, besitzt dieser die Aufzugsmelodie als besonderes Attribut.

3. Die nächste Aufzugsart ist der **kleine Lastenaufzug**, dieser lässt keinen Personentransport zu und hat damit eine maximale Personenanzahl von null. Das zulässige Gesamtgewicht beträgt hierbei allerdings 5000 kg. Das besondere Attribut dieser Klasse ist, dass die Quadratmeter-Zahl erfasst werden soll. Von diesem Typ gibt es 10.

4. Wie bei dem Personenaufzug gibt es auch von dem Lastenaufzug eine **große Variante**. Diese hat 10 000 kg zulässiges Gesamtgewicht und lässt auch keinen Personentransport zu. Von diesem gibt es fünf und es wird zusätzlich die Quadratmeter-Zahl erfasst.

5. Die letzte und besondere Aufzugstyp ist der **VIP-Aufzug:** Dieser kann nur durch ein Passwort gerufen werden und transportiert maximal fünf Personen mit einem Maximalgewicht von 400 kg. Dieser Aufzug ist auf fünf Exemplare limitiert. Das besondere Attribut dieser Klasse ist die Höchstgeschwindigkeit des Aufzugs.

Allgemein soll der Nutzer an der Steuerung eingeben können, wo er sich momentan befindet und zu welchem Stockwerk er Personen oder Güter transportieren möchte. Dabei muss er allerdings angeben, wie viele Personen transportiert werden sollen oder alternativ wie viel Last transportiert werden soll. Dabei soll das Programm allerdings automatisch Aufträge ablehnen, welche mehr als 30 Personen oder mehr als 10 000 kg transportieren sollen, da diese die maximale Last jeglicher Aufzüge überschreiten.

Durch diese Angaben kann das Programm anhand der angegebenen Voraussetzungen automatisch nach einem passenden Aufzug in der Nähe des Nutzers suchen. Der gefundene Auszug wird dann auf das Stockwerk des Nutzers gesendet. Der Nutzer soll zudem auch zu jedem Zeitpunkt die Möglichkeit haben, die Position aller Aufzüge abfragen zu können. Diese Abfrage soll durch den Nutzer nach Eigenschaften filterbar sein, um somit effizienter nach einer Abfrage suchen zu können. Im Hintergrund soll das Programm alle transportierten Personen und Lasten erfassen und in der Lage sein diese in einer Datei auf einer Festplatte zu speichern, welche auch bei einem Programmneustart wiederherstellbar sein sollen.

# **Darstellung der Alternativen**

//Welche Optionen wir hatten diese Aufgabe zu lösen. Alternative Lösungen kurz erwähnen. Dabei auch auf Codepassagen eingehen, die anders hätten gelöst werden können

- Alles in GUI packen

- Statt CSV in Datenbank

- Auf Server statt Lokal

# **Die gewählte Methode**

//Warum wir uns für die gewählte Methode entschieden haben, welche Vorteile hat sie gegenüber der anderen, bzw. warum eignet diese sich am besten.

- Verschiedene packages

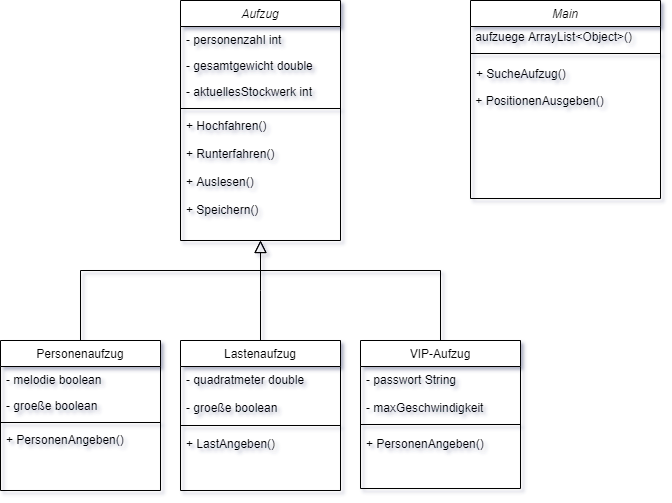
- fahrstuehle

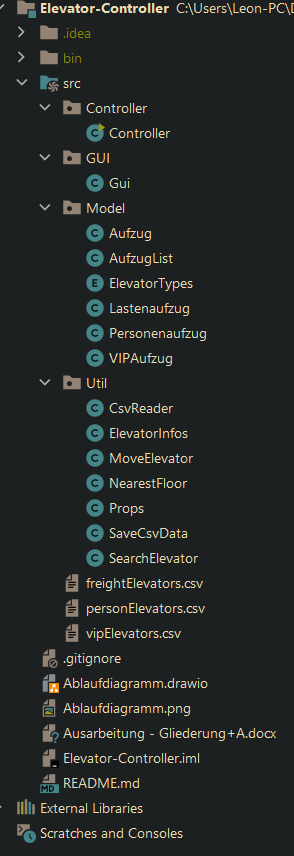
- GUI

- Model

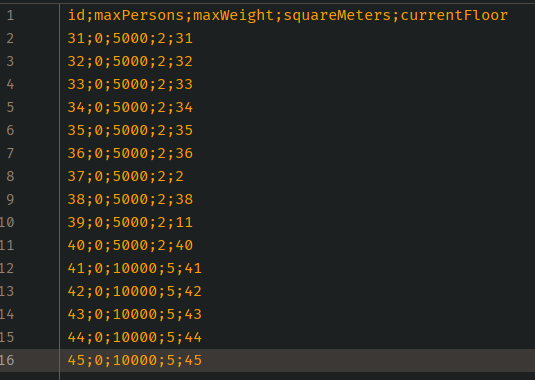
- Util

- CSV Datei

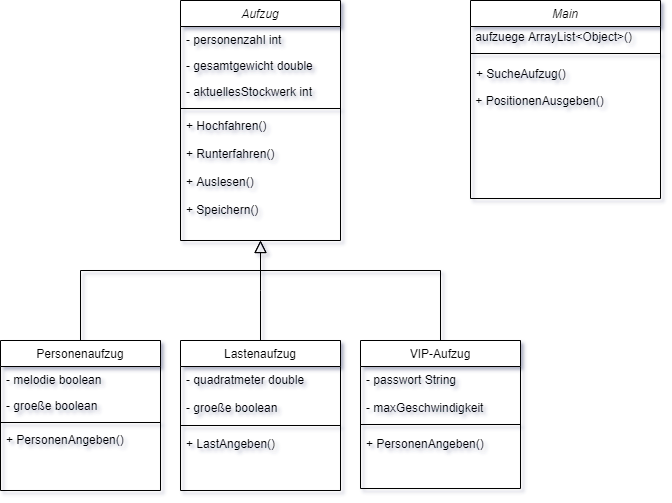
//Nicht aktuell

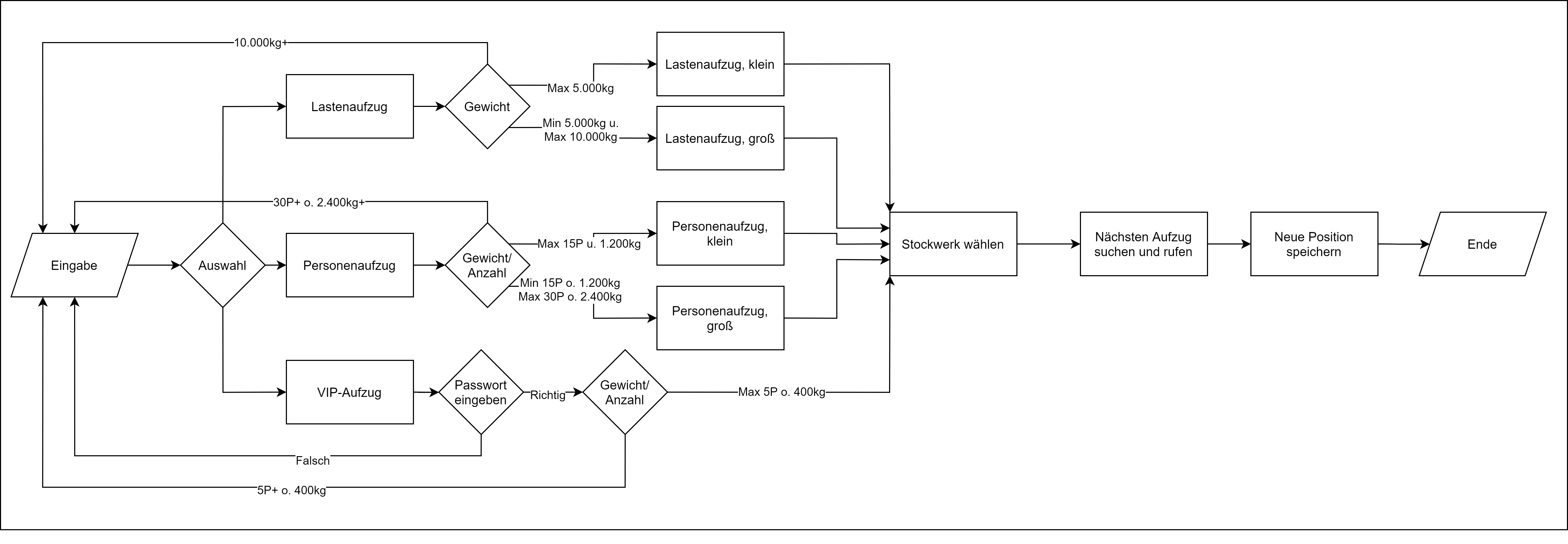


CSV Tabellen Beispiel



# **Erläuterung des Programms**

// Kurzer Überblick, wie das Programm aufgebaut ist inkl. Codepassagen als Beispiele.



# **Fazit und Zukunft des Programms**

//Fazit, schlüssiges Ende. Welche Perspektiven hat das Programm, was könnte in der Zukunft noch implementiert werden.

- Auf Server laufen lassen

- Neue Attribute