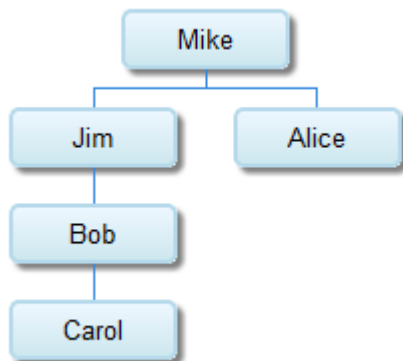


Prüfung Programming, Automation, Computation

1. Programming: Visualisierung mit Google Code Playground [5 Punkte]

Mit Google Code Playground können Organigramme visualisiert werden wie das folgende, wo Mike der Boss ist von Jim und Alice, Jim der Boss von Bob, Bob der Boss von Carol:



Die Beziehungen Mitarbeiter-Boss werden im Programmtext wie folgt beschrieben:

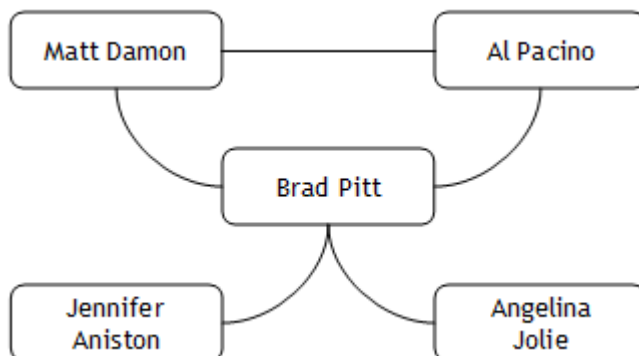
```
data.addColumn('string', 'Name Mitarbeiter');
data.addColumn('string', 'Name des Bosses des Mitarbeiters');
data.addRows(5);
data.setCell(0, 0, 'Mike');
data.setCell(1, 0, 'Jim');
data.setCell(1, 1, 'Mike');
data.setCell(2, 0, 'Alice');
data.setCell(2, 1, 'Mike');
data.setCell(3, 0, 'Bob');
data.setCell(3, 1, 'Jim');
data.setCell(4, 0, 'Carol');
data.setCell(4, 1, 'Bob');
```

1. Beschreiben Sie in zwei, drei Sätzen, wie dieses Programm die Mitarbeiter-Boss-Beziehungen definiert:

Die Beziehungen werden in Form einer Tabelle beschrieben. In der linken Spalte stehen die Mitarbeiter, in der rechten Spalte die Bosse.

2 Punkte.

Angenommen, Google Code Playground kenne eine Visualisierung für Beziehungsnetzwerke à la Facebook. Die folgende Abbildung zeigt ein Beispiel eines kleinen Beziehungsnetzwerkes von Schauspielern/innen, die zusammen Filme gedreht haben:



2. Beschreiben Sie im folgenden im Programmcode dieses Beziehungsnetzwerk:

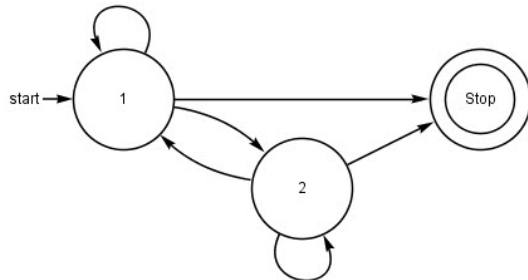
```
data.addColumn('string', 'Schauspieler');  
data.addColumn('string', 'Schauspieler');  
data.addRows(5);  
data.setCell(0, 0, 'Brad');  
data.setCell(0, 1, 'Matt');  
data.setCell(1, 0, 'Brad');  
data.setCell(1, 1, 'Al');  
data.setCell(2, 0, 'Brad');  
data.setCell(2, 1, 'Jennifer');  
data.setCell(3, 0, 'Brad');  
data.setCell(3, 1, 'Angelina');  
data.setCell(4, 0, 'Matt');  
data.setCell(4, 1, 'Al');
```

3 Punkte

- Schreiben Sie direkt in die Platzhalter _____ rein.
- Falls es zu viele Befehle hat, streichen Sie die überflüssigen Befehle gut erkenntlich durch.
- Sie können als Abkürzung auch einfach die Vornamen der Schauspieler/innen angeben.

2. Automation: Was macht das Programm? [5 Punkte]

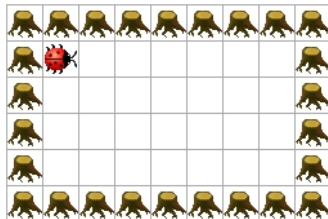
Sie finden folgendes Kara-Programm ohne jeden Hinweis darauf, welche Problemstellung das Programm löst. Nicht einmal die Zustände sind sprechend benannt:



1	2
<div>Kara macht: Nächster Zustand:</div> <div> <input checked="" type="checkbox"/> no <input type="checkbox"/> yes or no </div> <div> <input checked="" type="checkbox"/> yes <input type="checkbox"/> no </div> <div> <input checked="" type="checkbox"/> yes <input type="checkbox"/> yes </div> <div> <div>1</div> <div>2</div> <div>Stop</div> </div>	<div>Kara macht: Nächster Zustand:</div> <div> <input checked="" type="checkbox"/> no <input type="checkbox"/> yes or no </div> <div> <input checked="" type="checkbox"/> yes <input type="checkbox"/> no </div> <div> <input checked="" type="checkbox"/> yes <input type="checkbox"/> yes </div> <div> <div>2</div> <div>1</div> <div>Stop</div> </div>

1. Simulieren Sie die Programmausführung. Markieren Sie in der folgenden leeren Welt, wie die Welt nach der Ausführung des obigen Programmes aussieht:

- Tragen Sie einen Pfeil für Karas Position ein, der in seine Blickrichtung zeigt.
- Markieren Sie Kleeblätter mit K.



Lösung

2 Punkte



2. Beschreiben Sie in zwei, drei Sätzen, wie das Programm seine Aufgabe löst:

- Was macht das Programm in durch Bäumen begrenzten rechteckigen Welten?
- Welche Funktion hat Zustand 1?
- Welche Funktion hat Zustand 2?

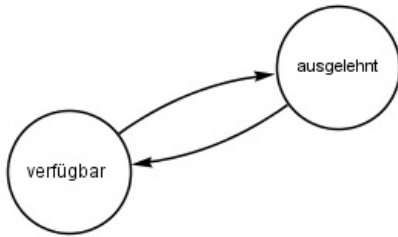
Das Programm legt horizontale Kleeblattstreifen in jede zweite Zeile innerhalb des durch die Bäume begrenzten Rechtecks:

- * Zustand 1 läuft von links nach rechts und legt Kleeblätter in der aktuellen Zeile.
- * Zustand 2 läuft von rechts nach links und lässt die aktuelle Zeile leer.

3 Punkte.

3. Automation: Automaten [5 Punkte]

Wir haben gesehen, dass Automaten verwendet werden können, um vielfältige Abläufe formal zu beschreiben. Eine Bibliothek möchte die Abläufe ihres Ausleihsystems mit Automaten beschreiben. Der erste Entwurf sieht wie folgt aus:



Dahinter stehen folgende Überlegungen:

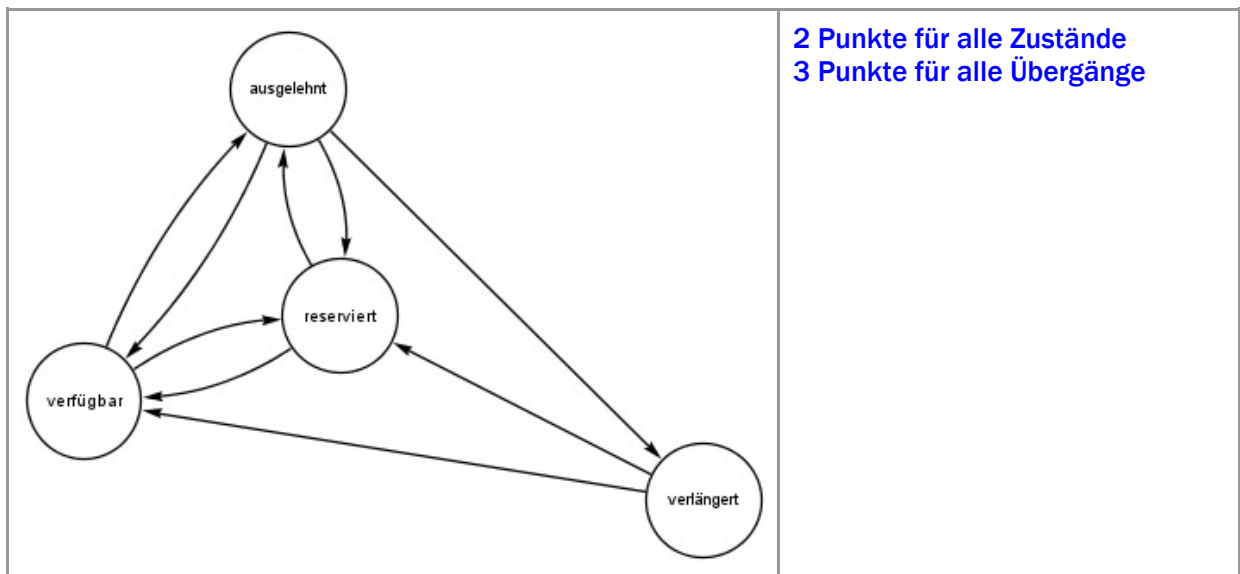
- Ein Buch kann verfügbar sein (Zustand verfügbar).
- Ein Buch kann ausgelohnt sein (Zustand ausgelohnt).
- Ein verfügbares Buch kann ausgelohnt werden (Übergang verfügbar→ausgelohnt).
- Ein ausgelohntes Buch kann verfügbar werden (Übergang ausgelohnt→verfügbar).

Bücher sollen auch reserviert werden können, und ausgelohnte Bücher sollen verlängert werden können. Entwerfen Sie für die Bibliothek einen Automaten, der folgende Überlegungen abdeckt:

- Ein Buch kann verfügbar sein.
- Ein Buch kann ausgelohnt sein.
- Ein Buch kann reserviert sein.
- Ein Buch kann verlängert sein.
- Ein verfügbares Buch kann ausgelohnt oder reserviert werden.
- Ein ausgelohntes Buch kann verfügbar oder reserviert oder verlängert werden.
- Ein reserviertes Buch kann ausgelohnt oder verfügbar werden.
- Ein verlängertes Buch kann reserviert oder verfügbar werden.

Tragen Sie unten die Zustände und Übergänge wie im Bild oben ein:

- Zustände klar benannt.
- Übergänge eindeutig erkennbar.



4. Automation: Probleme lösen [5 Punkte]

Im Unterricht haben wir die Marshmallow Challenge gemacht. Die Challenge ist nicht nur unterhaltsam, sie zeigt auch Vorgehensweisen bei Projekten auf. Sie beschliessen, die Challenge in Ihrer Pfadigruppe, Ihrem Fussballverein, Ihrer Hip-Hop-Band oder so durchzuführen.

1. Wählen Sie eine Gruppe aus Ihrem Umfeld aus. Beschreiben Sie in zwei, drei Sätzen, ein denkbares Projekt dieser Gruppe.

Unser Entwicklerteam bei der Swisscom muss einen Bestellprozess für ein neues Projekt entwickeln. In diesem Projekt geht es darum, den Kundenwunsch zu erfassen und abschliessend die Bestellung an ein Drittsystem zu schicken.

1 Punkt

2. Beschreiben Sie in wenigen ausformulierten Sätzen, was sie Ihrer Gruppe als Erkenntnis aus der Challenge erläutern würden.

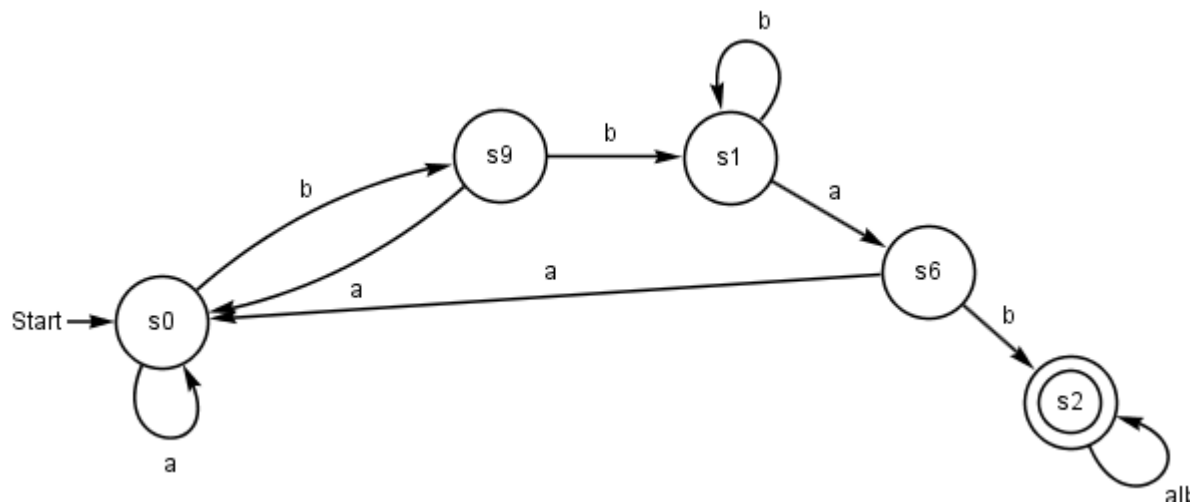
Das wichtigste Element im Bestellprozess ist das Abschicken der Bestellung. Das ist unser Marshmallow: Das Element, das einen ganzen Spaghetti-Turm zum Einsturz bringen kann. Daher soll zunächst der Bestellprozess nur soweit entwickelt werden, dass eine ganz einfache Bestellung abgeschickt werden kann. Optionen etc. kommen erst später hinzu. Denkbar ist auch, dass ganz zu Beginn einfach nur das Abschicken der Bestellung getestet wird: Noch ganz ohne Berücksichtigung des Kundenwunsches. So ist schon mal sichergestellt, dass der Marshmallow funktioniert.

4 Punkt

5. Computation: Automaten für Mustererkennung [5 Punkte]

Erstellen Sie einen Automaten für das Programm Exorciser, der alle Eingaben akzeptiert, die das Teilwort "bbab" beinhalten:

- Bezeichnen Sie die Zustände gut leserlich.
- Markieren Sie den Startzustand des Automaten.
- Markieren Sie deutlich, welche Zustände akzeptieren sind und welche nicht.
- Zeichnen Sie die Zustandsübergänge gut erkenntlich ein: Start- und Endzustand des Übergangs sowie Eingabe, für die dieser Übergang gewählt wird.



2 Punkte für alle Zustände
3 Punkte für alle Übergänge