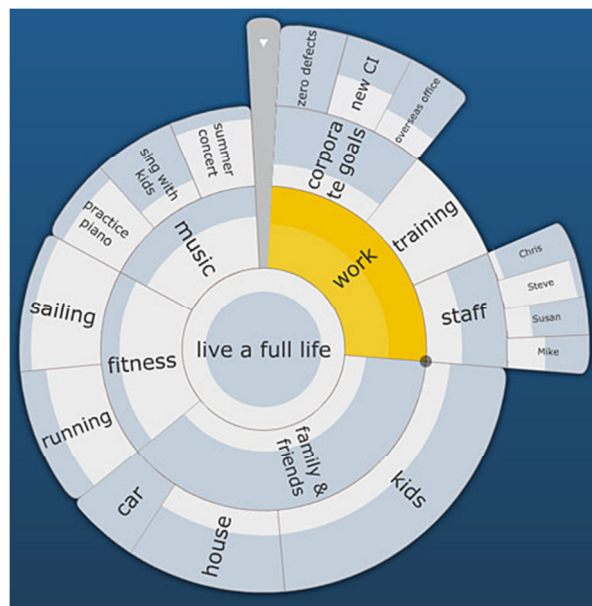


Prüfung Programming, Automation, Computation

Programming: Visualisierung mit Google Code Playground (5 Punkte)

Die Software GoalScape stellt Hierarchien kreisförmig dar. Die Visualisierung links unten stellt die Ziele einer (fiktiven) Person dar. Oberstes Ziel ist „live a full life“. Das Ziel für „work“ ist unterteilt in „corporate goals“, „training“ und „staff“. Analog sind andere Ziele wie „family & friends“ unterteilt.



Die Ziele werden im Programmtext wie folgt beschrieben (nur Auszug):

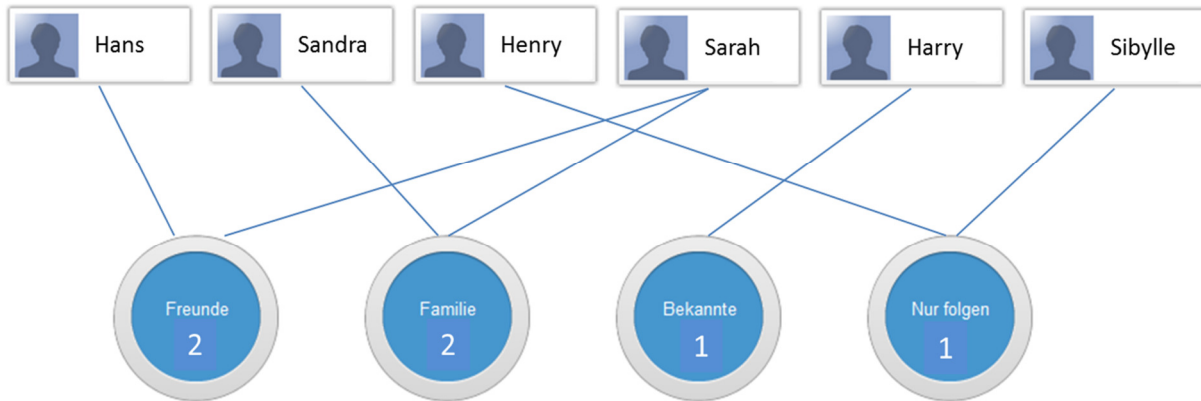
```
data.addColumn('string', 'Ziel 1');
data.addColumn('string', 'Ziel 2');
data.addRows(8);
data.setCell(0, 0, 'live a full life');
data.setCell(1, 0, 'work');
data.setCell(1, 1, 'live a full life');
data.setCell(2, 0, 'corporate goals');
data.setCell(2, 1, 'work');
data.setCell(3, 0, 'training');
data.setCell(3, 1, 'work');
data.setCell(4, 0, 'staff');
data.setCell(4, 1, 'work');
data.setCell(5, 0, 'family & friends');
data.setCell(5, 1, 'live a full life');
data.setCell(6, 0, 'fitness');
data.setCell(6, 1, 'live a full life');
data.setCell(7, 0, 'music');
data.setCell(7, 1, 'live a full life');
```

Google Code Playground: Aufgabe 1

Beschreiben Sie in zwei, drei Sätzen, wie beliebig verschachtelte Hierarchien mit nur zwei Spalten definiert werden können. Vielleicht hilft es Ihnen, wenn Sie sich dafür an die Visualisierung von Organigrammen erinnern.

Google Code Playground: Aufgabe 2

Angenommen, Google Code Playground kenne eine Visualisierung für Google+, für die Zuordnung von Personen zu den sogenannten „Circles“ (Gruppen von Personen, durch Benutzer definiert). Eine Visualisierung könnte wie folgt aussehen:



Beschreiben Sie im folgenden Programmcode diese Zuordnung von Personen zu Circles. Das Programm muss definieren, welche Person zu welchem oder zu welchen Circles gehört:

```
data.addColumn('string', '_____');
data.addColumn('string', '_____');
data.addColumn('string', '_____');
data.addColumn('string', '_____');
data.addRows(____);
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
data.setCell(____, ____, '_____');
```

- Schreiben Sie direkt in die Platzhalter _____ rein.
- Falls es zu viele Befehle hat, streichen Sie die überflüssigen Befehle gut erkenntlich durch.

Automation: Schreiben Sie einen Automaten für Kara (5 Punkte)

























Schreiben Sie einen Automaten, der Kara in einer Welt wie in der Abbildung links gezeigt Kleeblätter legen lässt. Kara soll aufhören, wenn vor ihm ein Pilz steht:



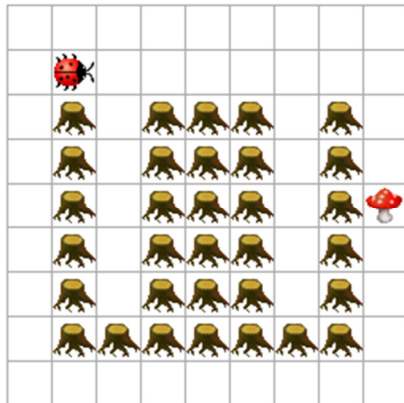
Kara weiss nicht, wie viele „Baumsäulen“ diese Welt hat, und er weiss nicht, wie „hoch“ diese Baumsäulen sind. Er weiss nur, dass der Pilz direkt nach einer Baumsäule folgt.

Schreiben Sie Ihren Automaten in die Zustandstabelle unten:

- Ein Zustand mit den drei angegebenen Sensoren reicht für eine Lösung aus.
- Machen Sie kenntlich, welche Sensorwerte Sie für einen Übergang erwarten.
- Geben Sie die Befehle an, die Kara ausführen soll.
- Notieren Sie den nächsten Zustand (laufe oder Stop):
- Sollte es zu viele Zeilen haben, können Sie die nicht-benötigten Zeilen ebenfalls streichen.

laufe			Kara macht:	Nächster Zustand:
				
				
				
				
				
				
				
				

Analysieren Sie Ihre Lösung: Funktioniert Sie auch in einer Welt wie in der folgenden Abbildung gezeigt, wo der Pilz woanders steht und wo eine der "Säulen" dicker ist?



Ja, meine Lösung funktioniert auch in dieser Welt, oder nein, meine Lösung funktioniert in dieser Welt nicht. Wichtig: Es spielt keine Rolle, ob die Antwort ja oder nein lautet! Es geht nur darum, dass Sie Ihr Programm in Gedanken noch einmal durchspielen.

Begründen Sie Ihre Antwort in zwei, drei Sätzen. Begründen Sie einerseits die in Bezug auf die "dicke" Säule, andererseits in Bezug auf die veränderte Position des Pilzes:

„Dicke“ Säule:

Veränderte Pilz Position:

Automation: Modellieren mit Automaten (5 Punkte)

Ihr neues Notebook von SuperComputer hat ein Problem: Der Bildschirm zeigt nur noch bunte Streifen. Sie rufen den Kundendienst von SuperComputer an. Ihre Anfrage wird wie folgt bearbeitet (die Beschreibung ist für unsere Zwecke idealisiert):

- Ihr Problem wird erfasst.
- Ihr Problem wird einer Sachverständigen zugewiesen.
- Ihr Problem wird von der Sachverständigen bearbeitet.
- Das Problem ist irgendwann behoben.
- Sie werden informiert, dass das Problem behoben sei
- Damit ist das Problem für SuperComputer erledigt.

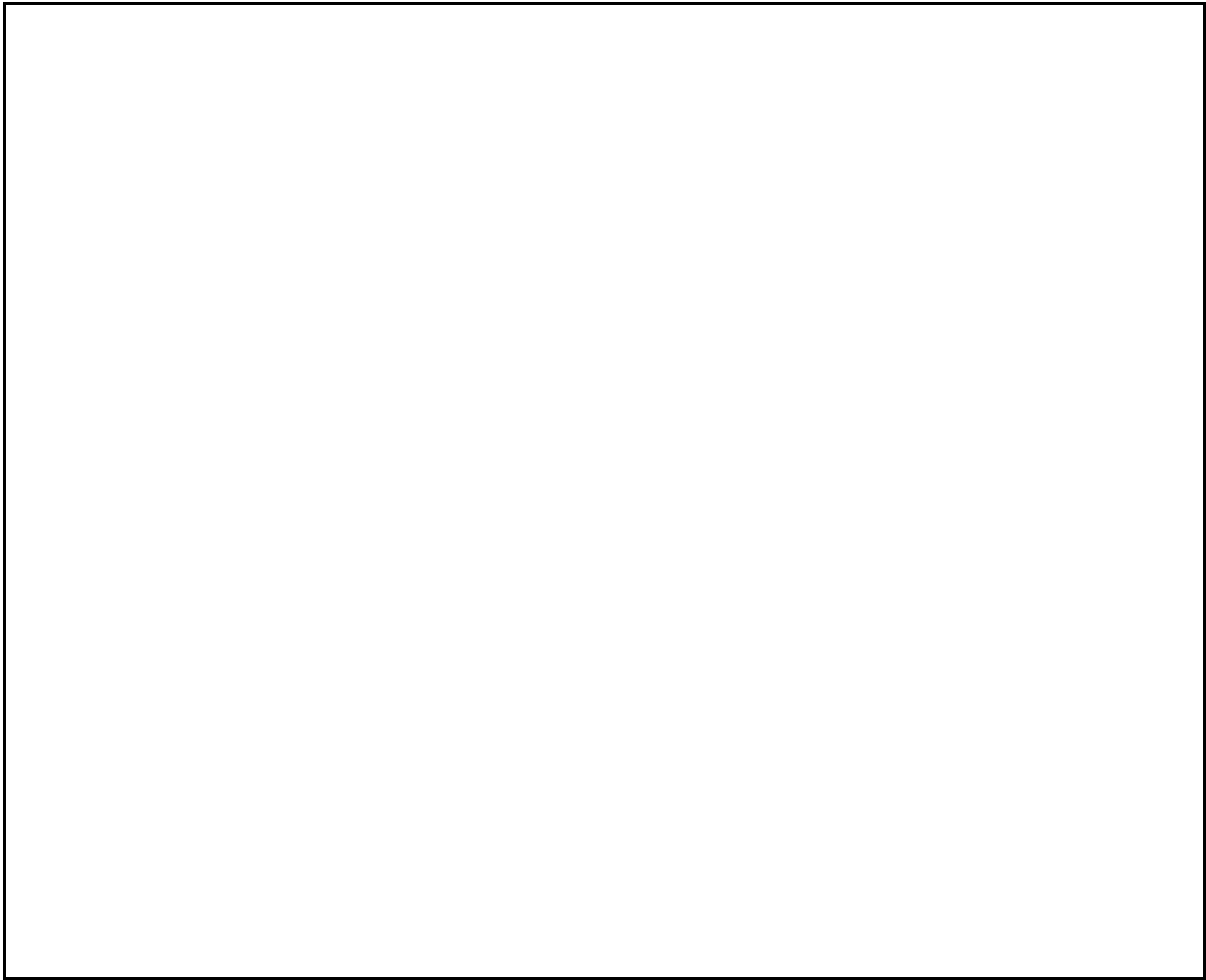
Es kann aber auch sein, dass die Sachverständige während der Zuweisung zum Schluss kommt, es liegt kein Fehler von SuperComputer vor: Ihr Anliegen wird abgewiesen. Es kann auch sein, dass die Sachverständige während der Bearbeitung Ihr Anliegen abweist. Sie werden auf jeden Fall informiert, dass Ihr Anliegen abgewiesen ist. Für SuperComputer ist die Sache damit erledigt: Einsprachen sind nicht möglich.

Wenn SuperComputer Sie informiert, dass das Problem behoben sei, und wenn Sie anderer Meinung sind, wird die Problembeschreibung überarbeitet. Anschliessend wird das Problem wieder einer Sachverständigen zugewiesen.

Modellieren Sie die beschriebenen Abläufe als endlichen Automaten:

<ul style="list-style-type: none">▪ Geben Sie Start- und Endzustand an.▪ Überlegen Sie sich, welche Zustände es braucht. Geben Sie den Zuständen sprechende Bezeichnungen, damit klar wird, welchem Schritt im Ablauf der Zustand entspricht.▪ Die Übergänge brauchen Sie nicht zu beschriften.	<p>Die Abbildung zeigt ein Beispiel, wie ein Automat gezeichnet wird:</p> <pre>graph LR; start((start)) --> Start((Start)); Start --> irgendwas((irgendwas)); irgendwas --> Start; irgendwas --> Stop(((Stop)))</pre>
---	---

Zeichnen Sie hier Ihren Automaten für die Abläufe im Kundendienst:

A large, empty rectangular box with a thin black border, intended for drawing an automaton (state machine) for customer service processes.

Automation: Endliche Automaten erklärt für Nicht-Informatiker (5 Punkte)

Wikipedia definiert endliche Automaten wie folgt (http://de.wikipedia.org/wiki/Endlicher_Automat):

„Ein endlicher Automat (EA, auch Zustandsmaschine, englisch finite state machine (FSM)) ist ein Modell eines Verhaltens, bestehend aus Zuständen, Zustandsübergängen und Aktionen.

Ein Automat heißt endlich, wenn die Menge der Zustände, die er annehmen kann (später S genannt), endlich ist. Ein EA ist ein Spezialfall aus der Menge der Automaten. Ein Zustand speichert die Information über die Vergangenheit, d.h. er reflektiert die Änderungen der Eingabe seit dem Systemstart bis zum aktuellen Zeitpunkt. Ein Zustandsübergang zeigt eine Änderung des Zustandes des EA und wird durch logische Bedingungen beschrieben, die erfüllt sein müssen, um den Übergang zu ermöglichen. Eine Aktion ist die Ausgabe des EA, die in einer bestimmten Situation erfolgt.“

Das ist zwar prägnant, aber ziemlich abstrakt! Erklären Sie Ihren Schulkolleg/innen, die nicht das Ergänzungsfach Informatik besuchen, was endliche Automaten sind. **Erklären Sie die Idee der Automaten und die Elemente der Automaten anschaulich anhand eines Beispiels aus dem Alltag.** In der Darstellung sind Sie frei. Sie können Zeichnungen verwenden, eine Geschichte schreiben usw. Ihre Kolleg/innen sollten einfach schnell ein Gefühl erhalten, was ein Endlicher Automat ist.

Computation: Reguläre Ausdrücke in endliche Automaten überführen (5 Punkte)

Erstellen Sie einen Automaten für das Programm Exorciser, der dem folgenden regulären Ausdruck entspricht:

$(b^*a)^*a$

- 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.

Lösungen

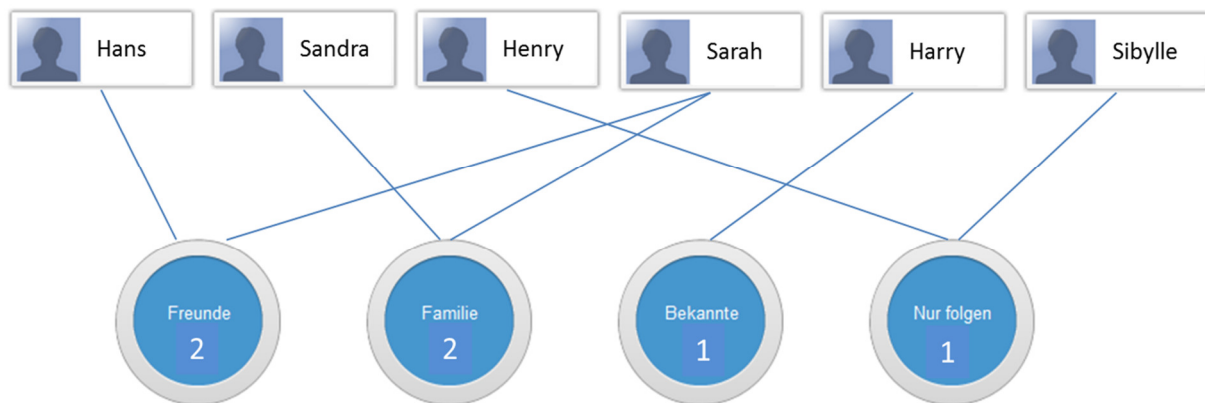
Programming: Visualisierung mit Google Code Playground (5 Punkte)

Beschreiben Sie in zwei, drei Sätzen, wie beliebig verschachtelte Hierarchien mit nur zwei Spalten definiert werden können (2 Punkte; 1 Punkt für Definition der beiden Spalten; 1 Punkt für Verschachtelung oder Mehrfachnennung):

Jede Zeile definiert ein Ziel und ein übergeordnetes Ziel: Die erste Spalte definiert ein Ziel, die zweite Spalte das dazugehörige übergeordnete Ziel.

Das Ziel in der zweiten Spalte kann dabei selbst ein übergeordnetes Ziel sein, für das wiederum ein übergeordnetes Ziel definiert werden kann.

Zu einem Ziel können zudem mehrere untergeordnete Ziele definiert werden: das gleiche Ziel kann in mehreren Zeilen vorkommen.



Beschreiben Sie im folgenden Programmcode diese Zuordnung von Personen zu Circles. Das Programm muss definieren, welche Person zu welchem oder zu welchen Circles gehört (3 Punkte: 1 Punkt für Definition der Spalten; 2 Punkte, wenn mindestens 5 der 7 Datensätze korrekt sind; 1 Punkt, wenn mindestens 3 der 7 Datensätze korrekt sind):

```
data.addColumn('string', 'Person');
data.addColumn('string', 'Circle');
data.addRows(7);
data.setCell(0, 0, 'Hans');
data.setCell(0, 1, 'Freunde');
data.setCell(1, 0, 'Sandra');
data.setCell(1, 1, 'Familie');
data.setCell(2, 0, 'Henry');
data.setCell(2, 1, 'Nur folgen');
data.setCell(3, 0, 'Sarah');
data.setCell(3, 1, 'Freunde');
data.setCell(4, 0, 'Sarah');
data.setCell(4, 1, 'Familie');
data.setCell(5, 0, 'Harry');
data.setCell(5, 1, 'Bekannte');
data.setCell(6, 0, 'Sibylle');
data.setCell(6, 1, 'Nur folgen');
```

Automation: Schreiben Sie einen Automaten für Kara (5 Punkte)

Die Lösung ist eine vereinfachte Version von „Wand entlang“ laufen: Kara weiss, dass immer rechts von ihm ein Baum sein muss. 3 Punkte: 2 Punkte, wenn das Programm Kara korrekt laufen lässt; 1 Punkt für die Kleeblätter und für die korrekte Behandlung des Pilzes.

wand entlang			
Kara macht:	Nächster Zustand:		
<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> </div>			
<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> </div>			
<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> </div>			
<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> <div style="margin-right: 10px;"> </div> </div>			

Ja, meine Lösung funktioniert auch in dieser Welt, oder nein, meine Lösung funktioniert in dieser Welt nicht:

Ja, sie funktioniert.

Begründen Sie Ihre Antwort in zwei, drei Sätzen. Begründen Sie einerseits die in Bezug auf die „dicke“ Säule, andererseits in Bezug auf die veränderte Position des Pilzes:

Bäume: Zeile zwei lässt Kara gerade auslaufen, solange rechts von ihm ein Baum ist und vorne keiner. Das funktioniert für das „Herunterlaufen“ in die Säule rein genauso wie für „Obendrauflaufen“.

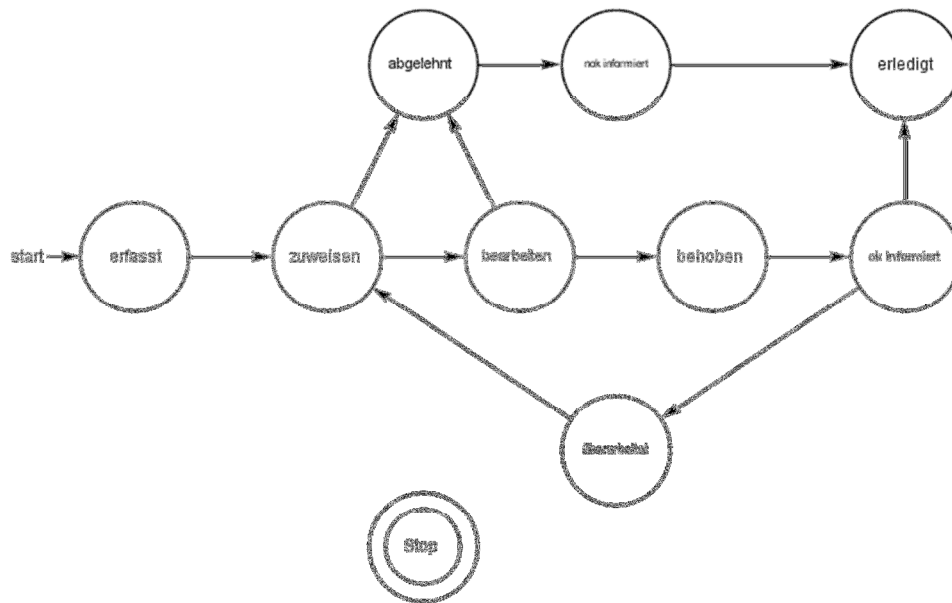
Pilz: Wenn rechts ein Baum von Kara ist (Zeile 2 und 3 im Programm), Kara macht nur dann einen Schritt vorwärts, wenn vor ihm kein Pilz ist. An der gezeigten Stelle stört der Pilz daher nicht.

2 Punkte; einer pro korrekte Antwort (zu Bäumen oder zu Pilz).

Modellieren mit Automaten (5 Punkte)

Mögliche Lösung. 3 Punkte für die Zustände (3 Punkte für ≥ 7 Zustände, 2 Punkte für ≥ 5 Zustände, 1 Punkte für ≥ 3 , sonst 0 Punkte), 2 Punkte für die Übergänge.

Startzustand: erfasst



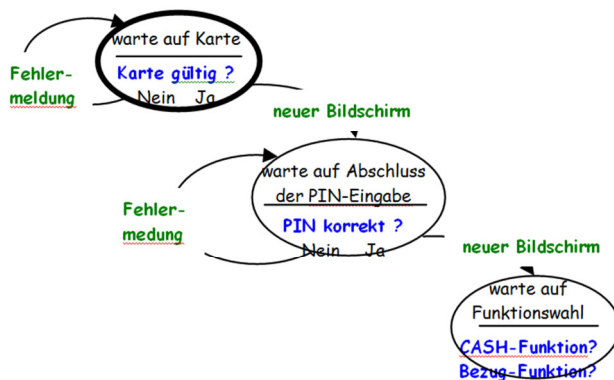
Endzustand: erledigt

Automation: Endliche Automaten erklärt für Nicht-Informatiker (5 Punkte)

Erklären Sie die Idee der Automaten und die Elemente der Automaten anschaulich anhand eines Beispiels aus dem Alltag. 5 Punkte: 1 Punkt für Verständlichkeit, je 1 Punkt für die Erklärung der einzelnen Elemente eines Automaten Zustand, Übergang, Bedingungen („Frage und Antwort“) Aktion.

Wir kennen Automaten aus dem Alltag: Billettautomat, Geldautomat, Parkuhren, Steuerung von Klimaanlage, CD-Players... Anhand eines Geldautomaten betrachten wir, wie ein Automat funktioniert und welche Begriffe seine Beschreibung benötigt.

Aus der Sicht des Geldautomaten läuft ein Kundenkontakt wie folgt ab (siehe Abbildung unten). Zu Beginn wartet er auf eine „Karte“. Wird eine Karte eingeschoben, prüft der Automat die Gültigkeit. Im negativen Falle zeigt er eine Fehlermeldung an und spuckt die Karte aus. Dann beginnt er von vorne und wartet wieder auf eine Karte. Im positiven Falle zeigt er einen neuen Bildschirm an. Jetzt wartet er auf den Abschluss der PIN-Eingabe. Dann stellt sich die Frage: „Ist der PIN gültig?“ Ist der PIN ungültig, so wartet der Automat wieder auf einen PIN. Ist der PIN korrekt, geht es weiter...



Dieser Ablauf ist die Beschreibung eines Automaten. Die Kreise sind dessen **Zustände**. Der fett gezeichnete Zustand ist der Startzustand. Sobald er eine Karte erhält, stellt er sich die **Frage**: „Ist diese Karte gültig?“ Je nach Antwort geht er in einen neuen Zustand über oder verbleibt im aktuellen Zustand. Die **Antworten** „Ja“ und „Nein“ sind mit diesen „Übergängen“ verbunden. Zu jedem **Übergang** gehören **Aktionen**, die auszuführen sind. Im Falle von „Ja“ ist die Aktion, einen neuen Bildschirm anzuzeigen.

Computation: Reguläre Ausdrücke in endliche Automaten überführen (5 Punkte)

LÖSUNG

