

Gisela Rosenbach  
Seiteneinsteigerin

Koblenz, den 26.04.06

Staatliches Studienseminar  
für das Lehramt an Gymnasien Koblenz

---

Unterschrift

## **Entwurf für die erste benotete Lehrprobe im Fach Informatik**

**Vertreterin des Studienseminars:** Frau StD' Hanna Mentges  
**Fachleiter:** Herrn StR' Martin Jakobs  
**Schule:** Max-von-Laue Gymnasium  
**Klasse / Kurs:** 11.1  
**Raum:** 306  
**Datum:** 26. April 2006  
**Zeit:** 2. Stunde (8.45 – 9.30)

**Thema der Unterrichtsreihe:** Zustandsbasierte Modellierung

**Thema der Stunde:** Einführung in die Zustandsbasierte Modellierung  
(Der Zustand als Gedächtnis)

## **Inhaltsverzeichnis**

<b>1. Intentionen</b>	<b>2</b>
1.1 Hauptintention	2
1.2 Teilintentionen	2
<b>2. Einordnung der Stunde in den Reihenkontext</b>	<b>2</b>
<b>3. Sachanalyse</b>	<b>4</b>
<b>4. Didaktische Analyse</b>	<b>6</b>
<b>5. Methodische Planung</b>	<b>8</b>
5.1 Begrüßung und Einstieg	8
5.2 Erarbeitung I	8
5.3 Sicherung I	9
5.4 Optional: Sicherung II	9
5.5 Optional: Erarbeitung II	10
5.6 Optional: Sicherung III	10
5.7 Hausaufgabe	10
<b>6. Literatur</b>	<b>11</b>
<b>7. Geplanter Stundenverlauf</b>	<b>12</b>

## **Anhang**

# 1. Intentionen

## 1.1 Hauptintention

Die Schüler lernen, ein erstes einfaches Problem (Heizsystem) zustandsbasiert zu modellieren und dann systematisch in Delphi zu implementieren

## 1.2 Teilintentionen

Die Schüler ...

- erkennen mit Hilfe des Heizsystems, dass man für die Implementierung bestimmter Systeme ein Gedächtnis braucht
- erkennen die Notwendigkeit der Modellierung
- lernen den Zustand als Gedächtnis kennen
- verstehen den Zustandsgraphen und die Zustandstabelle als Modellierungsmittel, können es anwenden und erweitern
- lernen die Modellierung als Vorlage für die Implementation kennen und anwenden

## 2. Einordnung der Stunde in den Reihenkontext

Die Unterrichtsreihe „Zustandsbasierte Modellierung“ umfaßt 6 - 7 Unterrichtsstunden. Die heutige Stunde ist die erste Stunde dieser Reihe. Sie ist die Einführung in die Zustandsbasierte Modellierung und rückt vor allem die fundamentale Idee dieser Modellierung, nämlich das Verständnis des Zustands als Gedächtnis ins Zentrum. Dadurch wird die grundlegende Basis geschaffen, dieses Modellierungskonzept verstehen und anwenden zu können.

In den nächsten Stunden wird die Anwendung und die Erweiterung des Modells zum endlichen Automaten im Vordergrund stehen und eine Ampelschaltung und ein Kaffeeautomat modelliert und simuliert werden.

In den letzten Wochen haben die Schüler die shape-Komponente in Delphi kennen- und anwenden gelernt. Sie haben die Komponenten über arrays verwaltet und kennen die Methode, die shape-Komponenten zu färben. Außerdem beherrschen sie

die grundlegenden Daten- und Kontrollstrukturen in Delphi und können unter bestimmten Bedingungen (z.B. wenn ein Knopf gedrückt wird) einen Farbwechsel der Komponente einleiten.

Diese Programmierfertigkeiten reichen zunächst aus, um das einfache Heizsystem in Delphi zu implementieren

### 3. Sachanalyse

Informatische Systeme, in denen in Abhängigkeit von Eingaben zeitliche Abläufe beschrieben werden, brauchen ein Gedächtnis.

Das System ist also nicht nur abhängig von Eingaben, die von außen kommen, sondern auch von „inneren Zuständen“.

Dieses Verhalten läßt sich modellieren durch ein Zustandsübergangsdiagramm, eingerichteter Graph (Zustandsgraph).

Ein System, das so modelliert werden kann, wird in der Informatik Automat genannt.

Mit Hilfe des Übergangsdiagramms gelangt man mit jeder Eingabe in eine neue Position (Zustand). Diese Zustände werden als Kreise dargestellt.

Die Zustände sind durch Pfeile (Kanten) verbunden und symbolisieren die Eingaben. Die Eingabe ist die auslösende Aktion, durch die das System von einem Zustand in den nächsten gelangt.

Der Zustand, in dem sich am Anfang des Prozesses die Kontrolle befindet, bezeichnen wir als Startzustand und markieren ihn mit einem eingehenden Pfeil.

Das Zustandsübergangsdiagramm soll in diesem Zusammenhang deterministisch sein, d.h., dass ein beliebiger Zustand für keine Eingabe mehr als eine Ausgangskante hat. So ist eindeutig bestimmt, welchen Zustand wir erreichen, nachdem eine Eingabe erfolgt ist (Eine Implementierung eines nichtdeterministischen Modells erfordert meist einen backtracking Algorithmus und wird zu komplex).

Ist unser System auf diese Weise modelliert, gibt es eine systematische Methode, das Zustandsübergangsdiagramm zu implementieren und zwar eine Methode, die prinzipiell für alle Zustandsübergangsdiagramme funktioniert:

Jedem Zustand entspricht ein Programmteil, das wir aufteilen können in ein Code-Fragment für den Zustand (in Abhängigkeit davon, was das System leistet) und in ein Code-Fragment zur Ermittlung des Folgezustands, welches prinzipiell immer gleich ist.

Beherrschen Schüler grundlegende Daten- und Kontrollstrukturen, wie Alternativen (**if .. then** oder case-Anweisung), sind sie in der Lage, ein modelliertes System ohne größere Probleme zu simulieren.

Deklariert man z.B. für jeden Zustand eine boolesche Variable (z.B. **var** z1, z2: boolean;), dann realisieren wir den Übergang von z1 in den Folgezustand z2 folgendermaßen:

**if** (z1 = true) **then**

.....{Code-Fragment für Zustand}

z1 := **false**;

z2 := **true**;

Die Größe der mit dieser Methode erstellten Programme ist proportional zur Anzahl der Zustände und Kanten.

Da die Anzahl der Knoten und Kanten leicht sehr groß werden können und die Darstellung des Zustandsübergangsdiagramms damit sehr unübersichtlich wird, ist die Zustandstabelle eine gleichwertige, übersichtliche Beschreibung:

Gegenwärtiger Zustand	Eingabe/ Folgezustand
Bsp. z1	z2

## 4. Didaktische Analyse

Ein durchgängiges Konzept im Informatikunterricht ist die Modellbildung und Simulation.

Modelle geben ein Abbild eines Ausschnittes der realen Welt wieder und finden sich in allen Wissenschaftsbereichen. Graphen sind dabei das geeignete Mittel zur Beschreibung.

In der Informatik dienen solche Modelle als Vorlage zur Implementierung bzw. Simulation von Informatiksystemen.

Neben der Modellierung durch Diagramme (z.B. Entity-Relationship-Diagramme, Use Case-Diagramme, Sequenzdiagramme, Klassendiagramme, Zustandsdiagramme) kennt die Informatik die Algebraische Modellierung durch Signaturen.

Zustandsbasierte Modellierung scheint für den Einstieg in Modellierungstechniken geeignet (s. auch Lehrplanentwurf für das Fach Informatik, RLP), da es sich hierbei um einen relativ einfachen Modellierungsansatz handelt. Einfache Modelle können auch von SchülerInnen, die keine Programmierer sind, simuliert werden.

Der Modellierungsansatz wird in allen Bereichen der Informatik eingesetzt, speziell auch durch die Erweiterungsmöglichkeit (Mealy-Automat, Akzeptor, Kellerautomat, Turingmaschine).

Anfängliche Modellierungen von Heizsystemen, Kaffeeautomaten, Ampeln etc. greifen die natürliche Vorstellung eines Automaten auf und verbessern das technische Verständnis für solche Systeme.

Durch eine Hinführung von zunächst relativ einfachen Modellierungsmittel zu mathematischen Formulierungen wird die Abstraktionsfähigkeit der SchülerInnen gefördert.

Durch den Umgang mit der graphischen Modellierung wird ihnen eine solche Darstellung und die begrifflichen Bezeichnungen vertraut. Dies verhilft den SchülerInnen zu einem sicheren Umgang mit graphischer Modellierung, die ihnen in vielen Fächern wieder begegnet (z.B. Strukturformeln in Chemie, Klassenhierarchien in der Biologie). Insbesondere verhilft es dazu, diese „lesen“ zu können

In dieser Unterrichtsreihe, speziell in dieser ersten Einführungsstunde, begegnen die SchülerInnen zum ersten Mal der Modellierung.

Bisher kennen sie nur eine direkte Umsetzung eines Problems ohne Zwischenstufe in ein lauffähiges Programm. Als einzige Strukturierung haben sie das Struktogramm kennengelernt.

Dagegen sind sie mit den algorithmischen Kontrollstrukturen (Sequenzen, Alternativen, Wiederholungen) vertraut, so dass es ihnen nicht schwer fallen wird, ein Zustandsübergangsdiagramm zu implementieren, sobald sie diese systematische Methode kennen- und verstehengelern haben.

Durch die Aufforderung an die SchülerInnen, zunächst ohne Zwischenschritt ein stark vereinfachtes Heiz-System zu programmieren, wird die Notwendigkeit einer Modellierung provoziert und ebenso die Notwendigkeit eines Gedächtnis erfahrbar gemacht, da es in einem Zustand aufgrund der Eingabe nicht möglich ist, zu entscheiden, zu welchem Zustand gewechselt wird.

So lernen die SchülerInnen den Kern des Zustands-Begriffes, nämlich den Zustand als Gedächtnis kennen. Diese Erkenntnis ist wesentlich für die Fähigkeit des zustandsbasierten Modellierens und bereitet eine Grundlage für die weiteren Unterrichtsstunden, aber auch für das Verständnis der Funktion von technischen Maschinen (Geräten) überhaupt.

Durch das Heizsystem wird den Schülern ein realitätsbezogenes einfaches Modell präsentiert, an dem alle wesentlichen Aspekte, welche eine zustandsbasierte Modellierung definiert, gezeigt und erfahrbar gemacht werden können.

Auch der fortschreitende Abstraktionsgrad läßt den Realitätsbezug nicht verlorengehen.

## 5. Methodische Planung

### 5.1 Begrüßung und Einstieg

Nach einer kurzen Begrüßung werden die SchülerInnen mit einer Betriebsanleitung eines Heizsystems konfrontiert. Die Schüler machen sich damit vertraut.

Ein kurzes Unterrichtsgespräch im Anschluss klärt Verständnisprobleme und verdeutlicht den Realitätsbezug (Motivation).

### 5.2 Erarbeitung I

Diese Phase bildet die Hauptphase dieser Stunde.

Nachdem der Arbeitsauftrag gestellt ist, sollen die SchülerInnen ohne Lehrerhilfe die gestellte Aufgabe lösen. Damit möglichst viele Gelegenheit zum Programmieren haben, aber dennoch sich mit jemandem beraten können, arbeiten sie in Partnerarbeit.

Mit shape-Komponenten sind sie vertraut. So werden sie auch sofort beginnen und sich des Problems, das noch nicht herausgearbeitet wurde, auch nicht bewußt sein.

Im Verlauf der Erarbeitungsphase wird das Problem aber zunehmend greifbar werden. Sie erkennen, dass das Problem mit den Mitteln, die sie bisher angewendet haben, nicht zu lösen ist.

Sie erkennen, dass es im Zustand hellblau zwei Möglichkeiten des Farbwechsel gibt, nämlich zu blau oder rot.

Einige gute Schüler werden eine Lösung finden. Sie werden sich evtl. mit einer booleschen Variablen merken, woher sie gekommen sind.

Andere SchülerInnen werden keinen Lösungsansatz finden. Für sie besteht die Möglichkeit, das zu Verfügung gestellte Arbeitsblatt zu bearbeiten. Hier finden sie einen Ansatz zur Modellierung des Problems, der zu vervollständigen ist und das Problem, nämlich eine eindeutige Entscheidung im Zustand hellblau, deutlich werden läßt.

Falls SchülerInnen frühzeitig das Problem zufriedenstellend gelöst haben, erhalten diese den Auftrag, das System graphisch darzustellen. So werden auch diese SchülerInnen erstmals mit einer Modellierung konfrontiert.

Durch diese Binnendifferenzierung wird sichergestellt, dass alle SchülerInnen sich entsprechend ihres Leistungsniveaus intensiv mit dem Problem auseinandersetzen.

### **5.3 Sicherung I**

Es werden zunächst 2 bis 4 Programmierarbeiten (je nach Abweichung der Ergebnisse) vorgestellt. Da die SchülerInnen aufgefordert sind, das aufgetretene Problem und die Grundidee ihrer Lösung zu beschreiben, sollen die Lösungen auf einer Folie präsentiert werden.

Alternativ könnten die Lösungen mit einem Beamer präsentiert werden. Da es aber hier nicht so sehr auf das einwandfrei laufende Programm mit schöner Oberfläche ankommt, wurde hier aufgrund der schnelleren Abwicklung die Lösung mit Folie, die vorher vorbereitet werden kann, vorgezogen.

Ergebnis dieser Phase ist die Erkenntnis, dass wir um unser Problem zu simulieren, im Zustand hellblau ein Gedächtnis brauchen.

### **5.4 Optional: Sicherung II**

Diese Phase wird dann an Sicherung I angeschlossen, falls SchülerInnen das Arbeitsblatt bearbeitet haben.

2 bis 3 SchülerInnen stellen (ergänzend) die Ergebnisse ihrer Arbeitsblätter vor.

Sind noch Ergänzungen vorzunehmen, werden dies nun mit allen SchülerInnen und mit Hilfe der Lehrerin vorgenommen.

Anschließend können auch die Schüler, die eigenständig modelliert haben, ihre Ergebnisse einbringen.

Nun können Vor- und Nachteile einer Modellierung von allen diskutiert werden.

Die SchülerInnen erkennen, dass eine Modellierung wesentlich zur einfachen Programmierung beitragen kann.

Da nun die wichtigsten Aspekte (Gedächtnis, Modellierung) und Fachbegriffe (Zustand, Zustandsübergangsdigramm bzw. Zustandsgraph, Knoten, Kanten, Zustandstabelle, innerer Zustand) dieser Stunde gezeigt, bzw. erfahrbar gemacht wurden, kann nun das neue Thema (Zustandsbasierte Modellierung) bzw. das Thema der Stunde (Einführung: Der Zustand als Gedächtnis) an der Tafel notiert werden. Die Fachbegriffe werden nur stichwortartig notiert, da sie sich nochmals auf der Zusammenfassung der Stunde wiederfinden, die nun ausgeteilt wird.

### **5.5 Optional: Erarbeitung II**

Falls keiner der Schüler sich mit dem Arbeitsblatt beschäftigt hat, wird das Arbeitsblatt gemeinsam mit der Lehrerin in einem gelenkten Unterrichtsgespräch erarbeitet. Dies zum einen aus Zeitgründen, zum anderen aber auch, weil eine eigene Bearbeitung des Arbeitsblattes für die SchülerInnen, die eine Lösung gefunden und eine eigene Modellierung versucht haben, nicht motivierend ist.

### **5.6 Optional: Sicherung III**

Wie in Sicherung II. Die Präsentation und Ergänzung des Arbeitsblattes entfällt hier natürlich.

### **5.7 Hausaufgabe**

Zur Festigung des Gelernten und Weiterentwicklung wird das Problem jetzt von allen implementiert und zwar, indem für jeden Zustand des Modells eine boolesche Variable deklariert wird.

Als Weiterführung wird das Problem erweitert. Das Heizsystem soll nun 4 verschiedene Heizstufen haben. Zur Einübung des Modellierens ist hier der Zustandsgraph zu entwerfen.

## 6. Literatur

Aho, A.V. / Sethi, R. / Ullmann, J. D.: Compilerbau Bd1, Addison-Weseley GmbH  
1993

Baumann R.: Informatik für die Sekundarstufe II, Klett Verlag, 1993

Duden Informatik, Dudenverlag 1993

Gumm, H. P. / Summer, M: Einführung in die Informatik, 6. Auflage, Oldenburg  
Wissenschaftsverlag GmbH, 2004

Hubwieser, P. Didaktik der Informatik, Springer Verlag, 2004

Lehrplanentwurf für das Leistungsfach Informatik Rheinland-Pfalz, 2004

Modrow, E.: Theoretische Informatik mit Delphi, emu-online, 2005

Schubert, S. / Schwill, A.: Didaktik der Informatik, Spektrum Akademischer Verlag,,  
2004

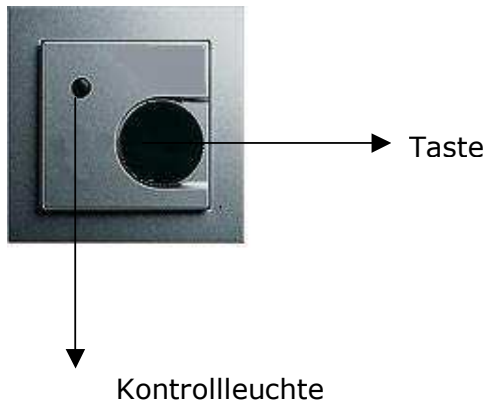
## 7. Geplanter Unterrichtsverlauf

Unterrichtsphase (Zeit)	Unterrichtsschritte	Methode (Medium)
Begrüßung und Einstieg (5 Min)	Die Lehrerin begrüßt die SchülerInnen und Gäste und präsentiert dann den SchülerInnen eine Betriebsanleitung einer Heizungsregelung. Nachdem die Schüler sich damit vertraut gemacht haben, nennen die Schüler nach Aufforderung Vorteile einer solchen Steuerung und Übertragbarkeit in andere Bereiche.	UG (OHP)
Erarbeitung I (15 Min)	Die Lehrerin stellt den Arbeitsauftrag, die Schaltung zu implementieren. <u>Binnendifferenzierung:</u> Die Lehrerin fordert die SchülerInnen, die bis zur Hälfte der zur Programmierung zur Verfügung gestellten Zeit keinen Lösungsansatz gefunden haben, auf, das zur Verfügung stehende Arbeitsblatt zu bearbeiten. SchülerInnen, die frühzeitig ihr Programm fertiggestellt haben erhalten den Arbeitsauftrag, das Problem graphisch darzustellen	Partnerarbeit, (PC)  Einzelarbeit (AB)
Sicherung I (10 Min)	3 Programmier-Ergebnisse werden von den SchülerInnen vorgestellt. Die SchülerInnen beurteilen und bewerten die vorgestellten Ergebnisse und erkennen, dass der Zustand „hellblau“ eine Merkfähigkeit besitzen muss.	SV (OHP) UG
Optional: Sicherung II (10 Min)	Falls SchülerInnen bis dahin das Arbeitsblatt bearbeitet haben, stellen zwei SchülerInnen ihre Ergebnisse vor. Die SchülerInnen korrigieren oder ergänzen das Arbeitsblatt mithilfe der Lehrerin. Anschließend untersuchen die SchülerInnen Vor- oder Nachteile einer solchen Modellierung. Sie erkennen, dass eine Modellierung eines Problems wesentlich zur einfachen Implementierung beitragen kann. In diesem Zusammenhang wichtige Fachbegriffe werden von der Lehrerin benannt und an die Tafel geschrieben. Ebenso wird das Thema der Reihe und der Stunde nun an die Tafel geschrieben.	SV (AB, OHP) UG Tafel
Optional: Erarbeitung II (5 Min)	Falls keiner der SchülerInnen das Arbeitsblatt bearbeitet hat, wird es von den SchülerInnen gemeinsam mit Unterstützung der Lehrerin bearbeitet.	guG, (AB, OHP)
Optional: Sicherung III (5 Min)	Die SchülerInnen untersuchen Vor- oder Nachteile einer solchen Modellierung. Sie erkennen, dass eine Modellierung eines Problems wesentlich zur einfachen Implementierung beitragen kann. In diesem Zusammenhang wichtige Fachbegriffe werden von der Lehrerin benannt und an die Tafel geschrieben.	UG Tafel
Hausaufgabe (2 Min)	Die SchülerInnen erhalten ein Blatt mit der Zusammenfassung der Stunde, worauf auch die Hausaufgabe notiert ist: Die Implementierung des Programmes mit vier booleschen Variablen und die Modellierung (Zustandsgraph) der Heizungsregelung: Erweiterung auf 4 verschiedene Heizstufen	LV (AB)

---

Bedienungsanleitung  
MLV colormatic  
Heizungsregelung

## 1. Geräteübersicht



Individuell heizen senkt die Kosten. Das MLV colormatic System sorgt dafür, dass die Heizungsanlage auch aus Ihren Wohnräumen gesteuert werden kann. Ganz wie es Ihrem Tagesablauf und Ihrer persönlichen Vorliebe entspricht, können Sie die Grundtemperatur durch einfaches Drücken der Taste verändern.

## 2. Wie funktioniert die Regelung?

Sie haben die Möglichkeit, zwischen drei verschiedenen Grundeinstellungen zu wählen (der aktuelle Status wird durch die Farbe der Kontrollleuchte angezeigt) :

Heizung aus (Kontrollleuchte ist blau)

Heizstufe I: abgesenkte Temperatur 15° (Kontrollleuchte ist hellblau)

Heizstufe II: normale Temperatur 22° (Kontrollleuchte ist rot)

Bei jedem Drücken der Taste ändert sich die aktuelle Heizleistung wie folgt:

Heizung aus  $\rightleftharpoons$  Heizstufe I  $\rightleftharpoons$  Heizstufe II  $\rightleftharpoons$  Heizstufe I  $\rightleftharpoons$  aus usw.

---

## Aufgabe (Partnerarbeit):

1.

Implementieren Sie die Heizungsregelung.

Sie soll neben dem Schalter eine Kontrollleuchte (Shape-Komponente) besitzen, welche die aktuelle Heizleistung in verschiedenen Farben anzeigt:

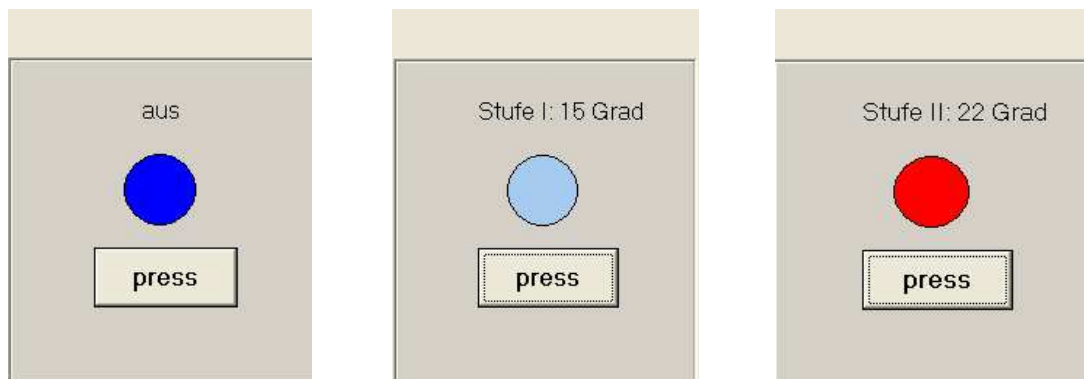
Heizung aus: blau (clBlue);

Heizstufe I: hellblau (clSkyBlue);

Heizstufe II: rot (clRed)

Der Wechsel der einzelnen Heizstufen soll in einer Prozedur "change" erfolgen.

Screenshots:



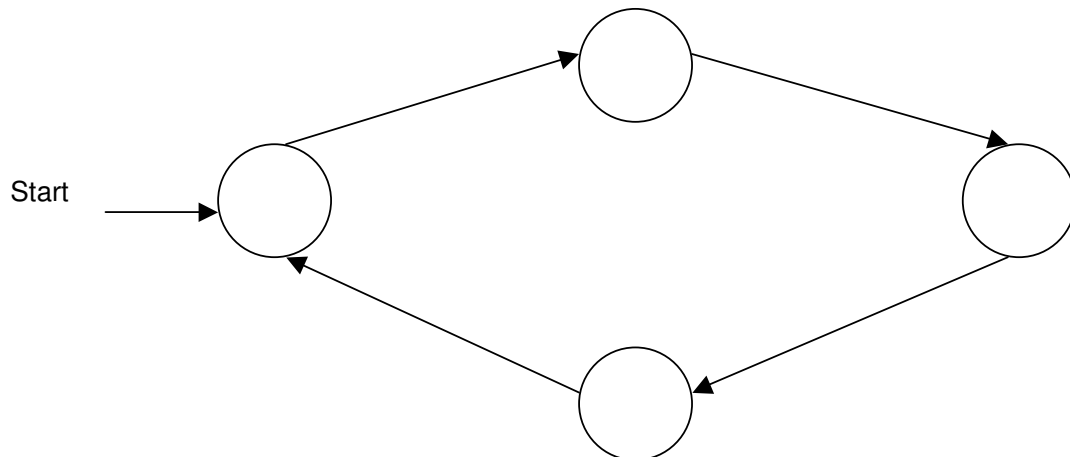
2.

Sichern Sie Ihre Ergebnisse (procedure change) auf der Folie.

Beschreiben Sie die Grundidee Ihrer Lösung und evtl. aufgetretene Probleme

# Arbeitsblatt: Modellierung der Heizungsregelung

## 1. Graphische Modellierung (Zustandsgraph):



### Aufgabe:

Beschriften Sie die Kreise mit den jeweiligen Zuständen (**blau**, **hellblau**, **rot**), die Pfeile mit der jeweiligen Eingabe (Drücken der Taste).

## 2. Tabellarische Darstellung (Zustandstabelle):

Ausgangszustand	Eingabe	Folgezustand
	Drücken der Taste (ButtonClick)	
Z0: <b>blau</b>		
Z1: <b>hellblau</b>		
Z2: <b>rot</b>		
Z3: <b>hellblau</b>		

### Aufgabe:

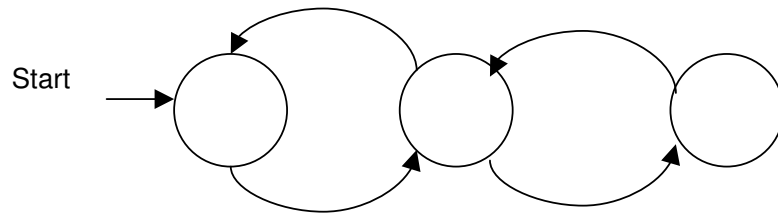
Tragen Sie in die Tabelle den jeweiligen Folgezustand ein.  
Erklären Sie, warum das Übertragen des graphischen Modells in die Tabelle Vorteile hat.

---

---

**3.**

Ein Schüler möchte sich Arbeit sparen und hat folgendes Modell aufgezeichnet:



- a) Beschriften Sie diesen Graphen.
- b) Erklären Sie dem Schüler, warum später bei der Umsetzung Probleme auftreten können

---

---

---

---

- c) Erklären Sie dem Schüler die Vorteile eines graphischen Modells bei der Lösung des Problems.

---

---

---

---

## Einführung in die zustandsbasierte Modellierung (Zustand als Gedächtnis)

### 1.

Implementierung einer Heizungsregelung:

Sie besitzt neben dem Schalter eine Kontrollleuchte (Shape-Komponente), welche die aktuelle Heizleistung in verschiedenen Farben anzeigt:

Heizung aus: blau (clBlue);

Heizstufe I: hellblau (clSkyBlue);

Heizstufe II: rot (clRed)

Der Wechsel der einzelnen Heizstufen erfolgt in einer Prozedur "change".

### 2.

Die Reaktion auf Drücken des Buttons kann sein:

- Wechsel in den Zustand „blau“,
- Wechsel in den Zustand „hellblau“                      oder
- Wechsel in den Zustand „rot“

#### **Problem:**

Befindet sich die Schaltung in dem Zustand „hellblau“, kann aufgrund der einzig möglichen Eingabe, nämlich dem Drücken des Buttons, das System in zwei verschiedene Zustände übergehen, nämlich entweder in „blau“ oder „rot“. Es kommt hierbei auf die „Vorgeschichte“ an, d.h., das System muss sich den vorherigen Zustand **merken**.

Dieses Verhalten wird in der Informatik mit dem Begriff „**Innerer Zustand**“ erfasst:

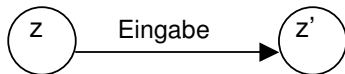
Die **Ausgabe** (Heizleistung) unseres System hängt also nicht nur von der **Eingabe** (Drücken des Buttons) ab, sondern auch von dem momentan **inneren Zustand** des Systems ab.

### 3.

Die **Modellierung** mithilfe eines **Zustandsgraphen** verdeutlicht das Problem (siehe Arbeitsblatt):

Jedem Zustand entspricht ein **Knoten** des Graphen. Gelangt man von einem Zustand  $z$  mittels der Eingabe in einen Folgezustand  $z'$ , so führt ein Pfeil (**Kante**) vom Knoten  $z$  zum Knoten  $z'$ .

An den Pfeil notieren wir die Eingabe.



Eine gleichwertige, übersichtliche Beschreibung ist die **Zustandstabelle** (siehe Arbeitsblatt).

### 4.

#### **Hausaufgabe:**

Der Zustandsgraph lässt sich auf einfache Weise implementieren, indem Sie für jeden Zustand eine boolesche Variable deklarieren (**var**  $z_0, z_1, z_2, z_3$  : boolean;).

Bsp.: **if** ( $z_0 = \text{true}$ ) **then** .....

- a) Implementieren Sie die Heizungssteuerung mit den vier booleschen Variablen.
- b) Erweitern Sie die Heizungssteuerung um eine weitere Heizstufe und modellieren Sie diese durch einen Zustandsgraphen.