Technische Universität Berlin
Institut für Kommunikations- und Softwaretechnik
FG Softwaretechnik
Franklinstr. 28/29
10587 Berlin

Koch
Bittner

Klausur

Softwaretechnik

19. Februar 1998

Aufgabe	1	2	3	4	Gesamt
Maximal	14	36	5	10	65
Erreicht:					

Aufgabe 1 (Fusion)

(Geschätzte Bearbeitungszeit: 30 min.)

1. Ein Werkzeugkasten für Arbeiten im Haushalt enthalte die folgenden 6 Abteilungen:

   - Schraubenzieher
   - Bohrer
   - Nägel, Schrauben und Dübel
   - Hämmer
   - Zangen
   - verschiedene weitere Werkzeuge, wie Meißel, Scheren, Zollstock, Wasserwaage etc.

   Bohrer sind charakterisiert durch ihre Stärke; sie können eingeteilt werden in Stein-, Holz- und Metallbohrer.

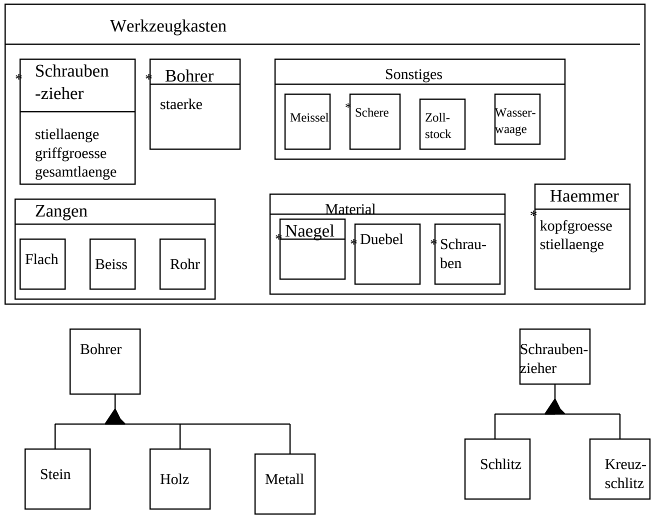
   Hämmer sind charakerisiert durch Größe des Kopfes und Länge des Stiels.

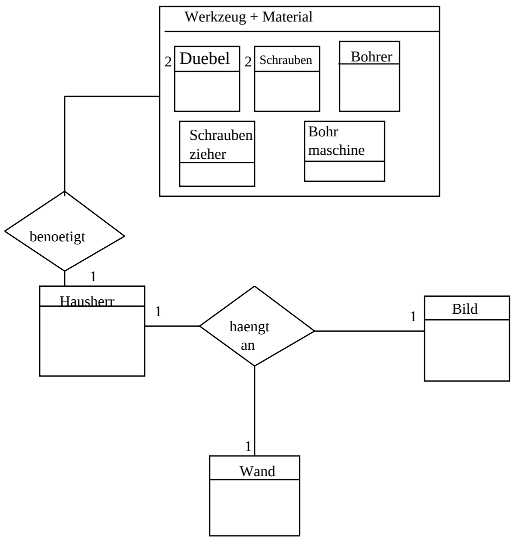
   Schraubenzieher haben die Attribute Stiellänge, Griffgröße und Gesamtlänge. Es gibt Schlitz- und Kreuzschlitz-Schraubenzieher.

   An Zangen befinden sich Flachzangen, Beißzangen und Rohrzangen im Kasten.

   Zum Aufhängen eines großen Bildes benötigt der Hausherr zwei Schrauben, zwei Dübel, einen Bohrer, eine Bohrmaschine und einen Schraubenzieher.

   Für das hier beschriebene Szenario ist nach der Methode Fusion ein Objektmodell zu entwerfen. Modellieren Sie Aggregationen und Generalisierung-Spezialisierungs-Beziehungen in getrennten Diagrammen. *10 Punkte*

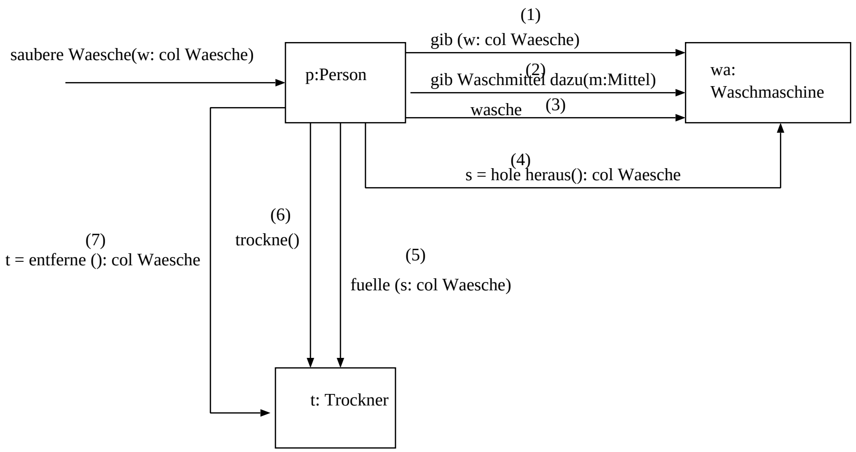




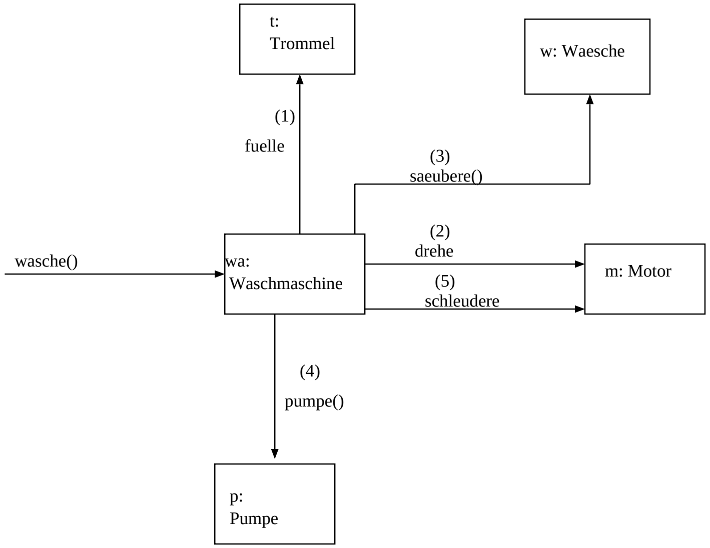
Hinweis: Verwenden Sie nur die oben beschriebenen Informationen

2. Sichtbarkeitsgraphen und Klassenbeschreibungen

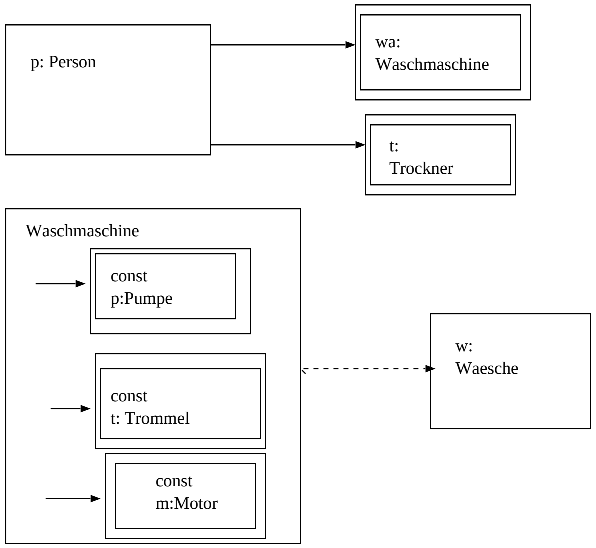
Objektinteraktionsgraph Waschen



Objektinteraktionsgraph Waschmaschine



a) Entwickeln Sie anhand der vorgegebenen Objektinteraktionsgraphen die dazugehörigen Sichtbarkeitsgraphen. *2 Punkte*



b) Stellen Sie mit Hilfe der vorgegebenen Objektinteraktionsgraphen und der von Ihnen erstellten Sichtbarkeitsgraphen die entsprechenden Klassenbeschreibungen auf. *2 Punkte*

class Person
attribute t : exclusive Trockner
attribute wa: exclusive Waschmaschine
method saubere Waesche(w: col Waesche)
endclass
class Waschmaschine
attribute const t: bound exclusive Trommel
attribute const m: bound exclusive Motor
attribute const p: bound exclusive Pumpe
method wasche()
method gib (w:col Waesche)
method gib Waschmittel dazu (m:Mittle)
method hole heraus ():col Waesche
endclass
class Trockner
...
endclass
class Motor
...
endclass
class Pumpe

...
endclass
class Waesche
...
endclass

Aufgabe 2 (Z)

(Geschätzte Bearbeitungszeit: 65 min.)

1. Der Terminkalender für einen Monat (**30 Tage**) einer Person sieht vor, daß täglich private oder geschäftliche Termine mit Dauer und Grund einer Verabredung eingetragen werden können. Termine pro Tag können in der Zeit zwischen 10 und 19 Uhr stattfinden. Es gibt nur ein- oder zweistündige Termine, die pünktlich zur vollen Stunde beginnen und enden. Die Anfangszeits des letzten Termins ist also entweder 17 Uhr bei einem zweistündigen Termin oder 18 Uhr bei einem einstündigen Termin.

   Das folgende Z-Spezifikationsfragment ist entsprechend der Teilaufgaben a) und b) zu vervollständigen.

   Die folgende unvollständige Spezifikation definiert den Kalender:

   Der Basistyp $Text$ bezeichnet den Grund des Termins.

$$[Text, Report]$$

$$frei : Text$$

$$\begin{array}{ll} Report ::= & zweistuendiger\_Termin\_bestaetigt \\ & \mid\ kein\_Termin\_moeglich \\ & \mid\ einstuendiger\_Termin\_bestaetigt \end{array}$$

Kalender

$$\begin{array}{l} termin : (1..30) \times (10..18) \rightarrow Text \\ freie\_termine : \mathbb{P}(1..30) \times (10..18) \\ \hline freie\_termine = \mathrm{dom}(termin \rhd \{frei\}) \end{array}$$

InitKalender

$$\begin{array}{l} Kalender' \\ \hline \forall\, t : 1..30, h : 10..18 \\ \hline termin'(t, h) = frei \end{array}$$

Zweistuendigen_Termin_eintragen

$$\begin{array}{l} \Delta Kalender \\ wunsch? : (1..30) \times (10..18) \\ grund? : Text \\ dauer? : 1..2 \\ confirm! : Report \\ \hline grund? \neq frei \\ dauer? = 2 \\ second(wunsch?) \neq 18 \\ wunsch? \in freie\_termine \wedge (first(wunsch?), second(wunsch?) + 1) \in freie\_termine \\ termin' = termin \oplus \{wunsch? \mapsto grund?, \\ (first(wunsch?), second(wunsch) + 1) \mapsto grund?\} \\ freie\_termine' = freie\_termine \setminus \{wunsch?, (first(wunsch?), second(wunsch) + 1) \\ confirm! = zweistuendiger\_Termin\_bestaetigt \end{array}$$

Wunschtermin_nicht_frei

$$\begin{array}{l} \Xi Kalender \\ term? : ((1..30) \times (10..18)) \times Text \\ wunsch? : (1..30) \times (10..18) \\ dauer? : 1..2 \\ confirm! : Report \\ \hline \textbf{if } dauer? = 1 \textbf{ then } wunsch? \notin freie\_termine \\ \ \textbf{else } second(wunsch?) = 18 \vee \\ \{wunsch?, (first(wunsch?), second(wunsch?) + 1)\} \\ \cap freie\_termine = \varnothing \\ confirm! = kein\_Termin\_moeglich \end{array}$$

a) Spezifizieren Sie die fehlenden Prädikatenteile der Operationsschemata Einstündigen_Termin_eintragen und Termin_loeschen. **20 Punkte**

Einstuendigen_Termin_eintragen

$$\begin{array}{|l}
\Delta Kalender \\
wunsch? : (1..30) \times (10..18) \\
grund? : Text \\
dauer? : 1..2 \\
confirm! : Report \\
\hline
grund? \neq frei \\
dauer? = 1 \\
wunsch? \in freie_termine \\
termin' = termin \oplus \{terminwunsch? \mapsto grund?\} \\
freie_termine' = freie_termine \setminus \{wunsch?\} \\
confirm! = einstuendigen_Termin_bestaetigt
\end{array}$$

Termin_loeschen

$$\begin{array}{|l}
\Delta Kalender \\
term? : ((1..30) \times (10..18)) \times Text \\
dauer? : 1..2 \\
confirm! : Report \\
\hline
termin' = termin \oplus \\
\textbf{if } dauer? = 1 \textbf{ then } \{first(term?) \mapsto frei\} \\
\ \textbf{else } \{first(term?) \mapsto frei, (first(first(term?)), \\
second(first(term?) + 1)) \mapsto frei\} \\
freie_termine' = freie_termine \cup \\
\textbf{if } dauer? = 1 \textbf{ then } \{first(term?)\} \\
\ \textbf{else } \{first(term?), first(first(term?)), second(first(term?)) + 1\}
\end{array}$$

b) Spezifizieren mit Hilfe der im Fragment angegebenen Operationen und des Schemakalküls die neue Operation **Terminwunsch_behandeln** [2 Punkte]

$Termin\_Wunsch\_behandeln == einstuendigen\_Termin\_eintragen \lor zweistuendigen\_Termin\_eintragen \lor Wunschtermin\_nicht\_frei$

c) Geben Sie den **Typ** der Variablen **termin** an. [2 Punkte]

termin: $\mathbb{P}(\mathbb{Z} \times \mathbb{Z} \times Text)$

Anmerkungen: Zum Zugriff auf die Koordinaten eines geordneten Paares **p** beliebigen kartesischen Produkttyps stehen die generischen Projektionsfunktionen **first (p)** und **second (p)** zur Verfügung. Ebenfalls zur Verfügung steht der bedingte Ausdruck **if** *Prädikat* **then** *expression1* **else** *expression2* .

2. Zum japanischen Go-Spiel gehören zwei gleichmächtige Mengen von Spielsteinen der Farben weiß und schwarz. Während des Spiels wird abwechselnd von beiden Spielern pro Zug ein Stein aus dem jeweils dem Spieler zugeordneten freien Fundus von weißen bzw. schwarzen Spielsteinen entnommen und auf das anfänglich leere Spielbrett gesetzt. Im Verlaufe des Spiels ist jedem gesetzten Stein entweder das Attribut *lebendig* oder *tot* zugeordnet.

   - Bei ausschließlicher Betrachtung der Spielsteine ist der **Zustandsraum** des Go-Spiels in Z zu spezifizieren. [12 Punkte]

$[Steine, Farbe]$

$$\begin{array}{l}
Farbe ::= weiss \\
\qquad\quad\ |\ \ schwarz
\end{array}$$

$max : \mathbb{N}$

Go

$$\begin{array}{|l}
stein : Steine \twoheadrightarrow Farbe \\
frei : Steine \twoheadrightarrow Farbe \\
tot : Steine \twoheadrightarrow Farbe \\
lebendig : Steine \twoheadrightarrow Farbe \\
\hline
\forall f_1, f_2 : Farbe \mid f_1 \neq f_2 \bullet \# \mathrm{dom}\, stein \rhd \{f_1\} \\
= \# \mathrm{dom}\, stein \rhd \{f_2\} = max \\
\forall f : Farbe \bullet (\mathrm{dom}\, frei \rhd \{f\} \cup \mathrm{dom}\, tot \rhd \{f\} \\
\cup \mathrm{dom}\, lebendig \rhd \{f\} = \mathrm{dom}\, stein \rhd \{f\}) \\
\land \\
(\mathrm{dom}\, frei \rhd \{f\} \cap \mathrm{dom}\, tot \rhd \{f\} \cap \mathrm{dom}\, lebendig \rhd \{f\} = \varnothing)
\end{array}$$

1.Alternative:

Go

$$\begin{array}{|l}
stein : Steine \twoheadrightarrow Farbe \\
frei : \mathbb{P}\, Steine \\
tot : \mathbb{P}\, Steine \\
lebendig : \mathbb{P}\, Steine \\
\hline
\# stein(weiss) = \# stein(schwarz) = max \\
\#(frei \cup tot \cup lebendig) = 2 * max \\
\mathrm{dom}\, stein = frei \cup lebendig \cup tot \\
frei \cap tot \cap lebendig = \varnothing
\end{array}$$

2. Alternative:

$$
\begin{array}{|l}
\textbf{Go} \\
\hline
w\_steine, w\_frei, w\_tot, w\_lebendig, \\
s\_steine, s\_frei, s\_tot, s\_lebendig : \mathbb{P}\,Steine \\
\hline
w\_steine \cap s\_steine = \varnothing \\
\#w\_steine = \#s\_steine = max \\
w\_frei \cup w\_tot \cup w\_lebendig = w\_steine \\
w\_frei \cap w\_tot \cap w\_lebendig = \varnothing \\
s\_frei \cup s\_tot \cup s\_lebendig = s\_steine \\
s\_frei \cap s\_tot \cap s\_lebendig = \varnothing \\
\hline
\end{array}
$$

Aufgabe 3 (OO-Z)

(Geschätzte Bearbeitungszeit: 5 min.)

Skizzieren Sie die Übertragung der Z-Spezifikation des Terminkalenders in eine OO-Z Klasse Kalender. *5 Punkte*

Geben Sie in jedem Fall das Zustandsschema, die Initialisierung und die Deklarationsteile der Operationen aus Aufgabe 2 an.

$$
\begin{array}{|l}
\textbf{Kalender} \\
\begin{array}{|l}
\hline
termin : (1..30) \times (10..18) \rightarrow Text \\
freie\_termine : \mathbb{P}(1..30) \times (10..18) \\
\hline
freie\_termine = \mathrm{dom}(termin \rhd \{frei\}) \\
\hline
\end{array} \\
\\
\begin{array}{|l}
\textbf{INIT} \\
\hline
\forall\, t : 1..30, h : 10..18 \\
\hline
termin'(t,h) = frei \\
\hline
\end{array} \\
\\
\begin{array}{|l}
\textbf{Einstuendigen\_Termin\_eintragen} \\
\hline
\Delta(termin, freie\_termine) \\
wunsch? : (1..30) \times (10..18) \\
grund? : Text \\
dauer? : 1..2 \\
confirm! : Report \\
\hline
grund? \neq frei \\
dauer? = 1 \\
wunsch? \in freie\_termine \\
termin' = termin \oplus \{terminwunsch? \mapsto grund?\} \\
freie\_termine' = freie\_termine \setminus \{wunsch?\} \\
confirm! = einstuendigen\_Termin\_bestaetigt \\
\hline
\end{array}
\end{array}
$$

$$
\begin{array}{l}
\textit{Zweistuendigen\_Termin\_eintragen} \\
\hline
\Delta(termin, freie\_termine) \\
wunsch? : (1..30) \times (10..18) \\
grund? : \ Text \\
dauer? : 1..2 \\
confirm! : Report \\
\hline
grund? \neq frei \\
dauer? = 2 \\
second(wunsch?) \neq 18 \\
wunsch? \in freie\_termine \ \wedge \\
(first(wunsch?), second(wunsch?) + 1) \in freie\_termine \\
termin' = termin \oplus \{wunsch? \mapsto grund?, \\
(first(wunsch?).second(wunsch) + 1) \mapsto grund?\} \\
freie\_termine' = freie\_termine \setminus \\
\{wunsch?, (first(wunsch?), second(wunsch) + 1)\} \\
confirm! = zweistuendiger\_Termin\_bestaetigt
\end{array}
$$

$$
\begin{array}{l}
\textit{Wunschtermin\_nicht\_frei} \\
\hline
term? : ((1..30) \times (10..18)) \times Text \\
wunsch? : (1..30) \times (10..18) \\
dauer? : 1..2 \\
confirm! : Report \\
\hline
\textbf{if } dauer? = 1 \textbf{ then } wunsch? \notin freie\_termine \\
\textbf{else } second(wunsch?) = 18 \ \vee \\
\{wunsch?, (first(wunsch?), second(wunsch?) + 1)\} \\
\cap freie\_termine \ = \emptyset \\
confirm! = kein\_Termin\_moeglich
\end{array}
$$

$$
\begin{array}{l}
\textit{Termin\_loeschen} \\
\hline
\Delta(termin, freie\_termine) \\
term? : ((1..30) \times (10..18)) \times Text \\
dauer? : 1..2 \\
confirm! : Report \\
\hline
termin' = termin \oplus \\
\textbf{if } dauer? = 1 \textbf{ then } \{first(term?) \mapsto frei\} \\
\textbf{else } \{first(term?) \mapsto frei, (first(first(term?)), \\
second(first(term?) + 1)) \mapsto frei\} \\
freie\_termine' = freie\_termine \cup \\
\textbf{if } dauer? = 1 \textbf{ then } \{first(term?)\} \\
\textbf{else } \{first(term?), first(first(term?)), second(first(term?)) + 1\}
\end{array}
$$

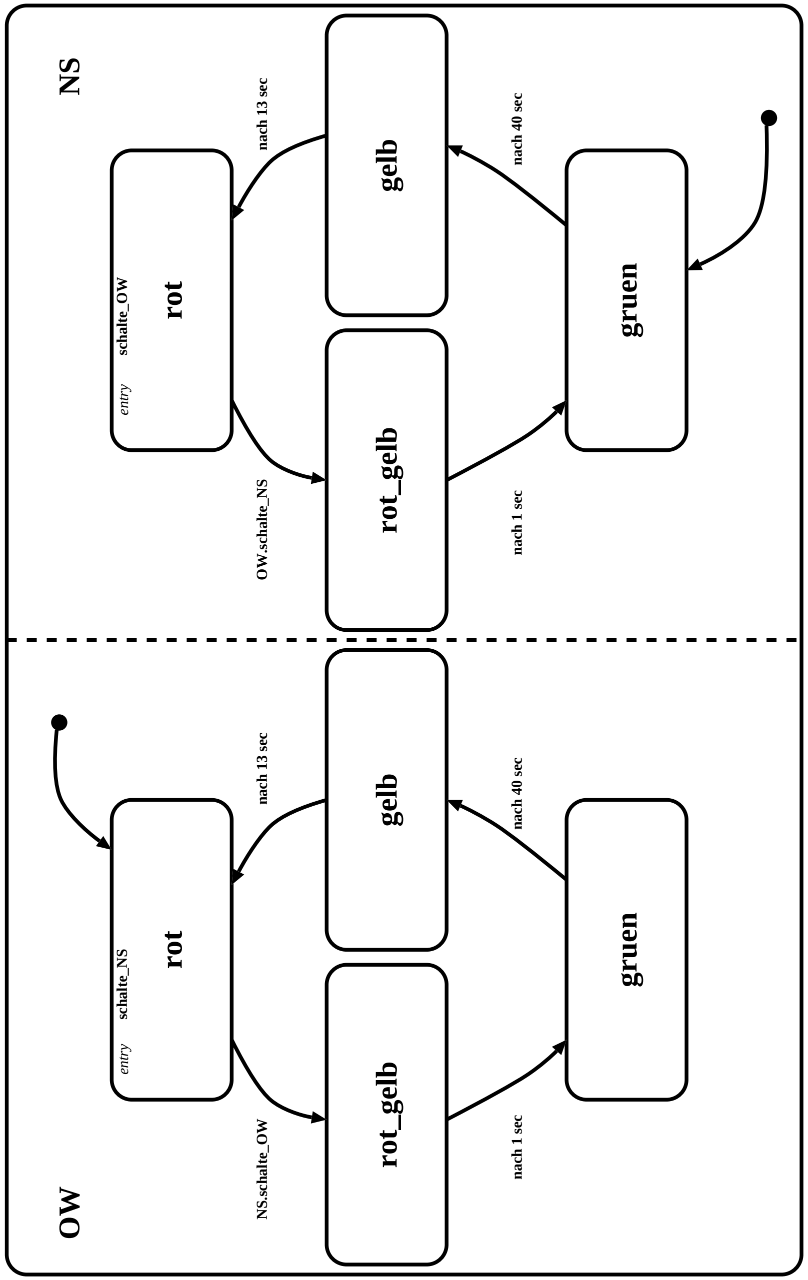
Aufgabe 4(Statecharts)

(Geschätzte Bearbeitungszeit: 20 min.)

Eine Ampelanlage steht an einer Kreuzung zweier gleichberechtigter Straßenzüge in Nord - Süd bzw. Ost - West Richtung. In der Spezifikation der Anlage entsprechen diesem Sachverhalt zwei parallele Teilsysteme NS und OW, die jeweils alle synchron geschalteten Lichtsignalanlagen (LSA) für den jeweilen Straßenzug modellieren. Jedes Teilsystem hat die Zustände rot, rotgelb,gelb und grün. Beim Schalten einer LSA im Zyklus von rot nach rotgelb, rotgelb nach grün, grün nach gelb und gelb nach rot verharrt die jeweils andere LSA im Zustand rot. Jede LSA ändert ihren Zustand von rot nach rotgelb, wenn die jeweils andere LSA in ihren eigenen Zustand rot eintritt.

Im initialen Zustand des Gesamtsystems ist eine der LSA auf rot, die andere auf grün geschaltet. Jede LSA schaltet nach 1 sec von rotgelb nach grün, nach 40 sec von grün nach gelb und nach 13 sec von gelb nach rot.

Spezifizieren Sie die Ampelanlage gemäß der oben angegebenen Beschreibung. *10 Punkte*


NS
rot
entry schalte_OW
gelb
rot_gelb
gruen
nach 13 sec
nach 40 sec
OW.schalte_NS
nach 1 sec
OW
rot
entry schalte_NS
gelb
rot_gelb
gruen
nach 13 sec
nach 40 sec
NS.schalte_OW
nach 1 sec