Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.1 Die Fibonacci-Zahlen sind folgendermaßen definiert:

$$f(n+2) = f(n+1) + f(n), \quad f(1) = 1, \quad f(0) = 0$$

1. Entwerfen Sie ein gefärbtes Petrinetz, das eine Stelle *fibs* besitzt und das die folgende Schaltfolge erlaubt
$$\mathbf{m}_0 \xrightarrow{*} \mathbf{m}_1 \xrightarrow{*} \cdots \xrightarrow{*} \mathbf{m}_n \xrightarrow{*} \cdots ,$$
wobei die Markierung der Stelle *fibs* in den $\mathbf{m}_n, n \in \mathbb{N}$ die folgende Multimenge ist:
$$\mathbf{m}_n(\mathit{fibs}) = 1'(0, f(0)) + \cdots 1'(n-1, f(n-1)) = \sum_{i=0}^{n-1} 1'(i, f(i))$$
Für die Markierung evtl. vorhandener weiterer Stellen wird nichts gefordert.
Entwerfen Sie das Netz so, dass die Übergäng von $\mathbf{m}_i \xrightarrow{*} \mathbf{m}_{i+1}$ durch das Schalten genau einer Transition (in möglicherweise verschieden Bindungen) realisiert wird.
2. Geben Sie alle Komponenten Ihres Netzes explizit an!
3. Implementieren Sie ihr Netz in *Renew*, simulieren sie es bis $\mathbf{m}_4$ und drucken Sie den Simulationsstatus aus!
Im Ausdruck sollen die Marken sichtbar sein, nicht nur die Markierungskardinalitäten. Zu erreichen mittels folgender Aktionen: (1) Edit → Select All → Nodes → Places und (2) Net → Marking → Tokens.
4. Variieren Sie das gefärbte Netz derart, dass die Markierung der Stelle *fibs* die $(0, f(0)), \ldots, (k-1, f(k-1))$ nicht als Multimenge, sondern als Sequenz $\langle f(k-1), \ldots, f(0) \rangle$ enthält. (Diese Reihenfolge ist am einfachsten zu erzeugen.) Verwenden Sie dazu den *Renew*-Listenoperator $\{\mathit{head} : \mathit{tail}\}$.

Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.2 Ein Fahrstuhl fährt zwischen den Stockwerken 1 und n. Er kann ein Stockwerk nach oben fahren (Aktion: up), ein Stockwerk nach unten fahren (Aktion: down), auf einem Stockwerk i halten (Aktion: stop) und die Fahrtrichtung umkehren (Aktion: turn)

In jedem Stockwerk y kann ein Fahrstuhl angefordert werden (Aktionen: req(i) mit $1 \leq i \leq n$).

In der Kabine kann ein Zielstock ausgewählt werden. (Aktionen: req(i) mit $1 \leq i \leq n$).

Die Steuerungslogik des Fahrstuhls ist die folgende: Er speichert alle Halteanforderungen, die für Stockwerke oberhalb seines aktuellen gelten, in einer Menge *ReqAbove*. Getrennt davon die Anforderungen für Stockwerke unterhalb *ReqBelow*.

Fährt die Kabine nach oben, so fährt sie weiter nach oben, solange es noch Anforderungen oberhalb gibt. Anderfalls wird die Richtung geändert. Analog, falls die Kabine nach nach unten fährt. Initial steht die Kabine im Stockwerk 1, Fahrtrichtung ist nach oben. Gibt es für das aktuelle Stockwerk eine Anforderung, so wird gestoppt.

1. Modellieren Sie dieses Szenario als gefärbtes Petrinetz. Verwenden Sie den in der Vorlesung präsentierten Editor RENEW, um das Netz zu erstellen!
2. Testen Sie das Modell, indem Sie im Simulationsmodus von RENEW zufällige Halteanforderungen generieren.

Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.3 Äquivalenzen in LTL.

1. Beweisen Sie die Äquivalenzen in LTL:

$$\begin{aligned} \mathbf{F}f &\equiv True\mathbf{U}f \\ \mathbf{G}f &\equiv \neg(\mathbf{F}\neg f) \\ \neg\mathbf{X}f &\equiv \mathbf{X}\neg f \end{aligned}$$

2. Zeigen Sie dass man anstelle des LTL-Modalitätensatz $\{X, F, G, U\}$ auch nur $\{X, U\}$ auskommen kann – ohne dass sich die Ausdruckskraft ändert.

3. Es gibt eine Variante des Until-Operators – der Weak-Until-Operator. Anders als bei $f_1\mathbf{U}f_2$ ist bei $f_1\mathbf{WU}f_2$ nicht garantiert, dass ein Zustand der f_2 erfüllt auch tatsächlich eingenommen wird. Der Operator ist wie folgt definiert:

$$\alpha \models f_1\mathbf{WU}f_2 \iff \big(\exists k \geq 0 : \alpha^k \models f_2 \wedge \forall 0 \leq j < k : \alpha^j \models f_1\big) \vee \forall 0 \leq j : \alpha^j \models f_1)$$

   Zeigen Sie, dass man im Modalitätensatz $\{X, F, G, U\}$ den Until-Operator durch den Weak-Until-Operator ersetezen kann – und umgekehrt auch – ohne dass sich die Ausdruckskraft ändert.

Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.4 Übersetzen Sie die folgenden Spezifikationstexte in eine LTL-Formel:

1. Nach einer Einschwingphase bleibt das System im User-Mode, es sei den ein Interrupt tritt auf.
2. Zunächst wird das Req-Signal am Req-Port gesetzt.

   Das Endergebnis liegt an den Output-Ports bereit, sobald das Ack-Signal anliegt.

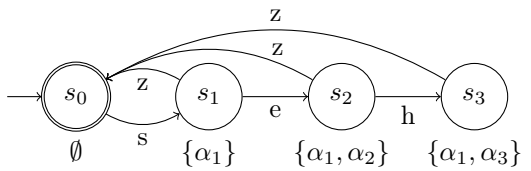
   In der Zwischenzeit sind die Output-Ports konstant auf Null gesetzt.

Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.5 Betrachten Sie die folgende Kripkestruktur H, das einen defekten Wasserkocher zeigt:



Aktion	Interpretation
s	Gerät in Steckdose einstöpseln
e	Gerät einschalten
h	Abschaltung durch Hitzesensor
z	Stecker ziehen

Die Zustände sind wie folgt etikettiert:

Aussagesymbol	Interpretation
α_1	Gerät ist mit Strom versorgt
α_2	Heizspirale ist in Betrieb
α_3	Sicherheitsabschaltung (Hitzesensor ist angeschlagen)

Betrachten Sie den unendlichen Zustandspfad $\pi = s_0 s_{i_1} s_{i_2} \ldots$ aus der Menge $(s_0 s_1 s_2)^\omega$.

Geben Sie an, ob für die folgenden LTL-Formeln f jeweils $H, \pi \models f$ und allgemeiner: $H \models f$ gilt. Geben Sie jeweils eine Begründung an!

f	$H, \pi \models f$	$H \models f$
$\circ(\alpha_3 \vee \alpha_1)$		
$\Box\alpha_2$		
$\Box(\alpha_1 \Longrightarrow \circ\alpha_2)$		
$\Box((\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3) \Longrightarrow \circ(\alpha_2 \vee \alpha_3))$		
$\Box\Diamond\alpha_2$		
$\Box((\neg\alpha_1)\ \mathbf{U}\ \alpha_1)$		

Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.6 Äquivalenzen in CTL.

1. Formulieren Sie die folgenden Äquivalenze in natürlicher Sprache und begründen Sie die Gültigkeit: $\mathbf{A}f \equiv \neg(\mathbf{E}\neg f)$

2. Beweisen Sie die Äquivalenzen:

$$\begin{aligned} \mathbf{AX}g &\equiv \neg\mathbf{EX}(\neg g) \\ \mathbf{EF}g &\equiv \mathbf{E}[\mathit{True}\mathbf{U}g] \\ \mathbf{AG}g &\equiv \neg\mathbf{EF}(\neg g) \\ \mathbf{AF}g &\equiv \neg\mathbf{EG}(\neg g) \end{aligned}$$

Tipp: Nutzen Sie in der Argumentation die einfacheren Äquivalenzen der ersten Teilaufgabe.

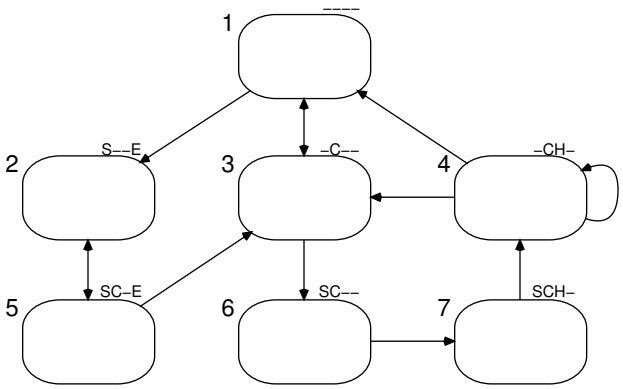
Teamnr.

	Vorname (lesbar!)	Name (lesbar!)
1		
2		
3		

Selbsteinschätzung: Wie gut ist Ihre Lösung? (Zutreffendes einkreisen) - - / - / + / + +

Übungsaufgabe 3.7 CTL-Model-Checking.

Betrachten Sie das Modell einer Mikrowelle:



1. Wenden Sie den CTL-Algorithmus aus Abschnitt 4.2 auf folgende Formel an:

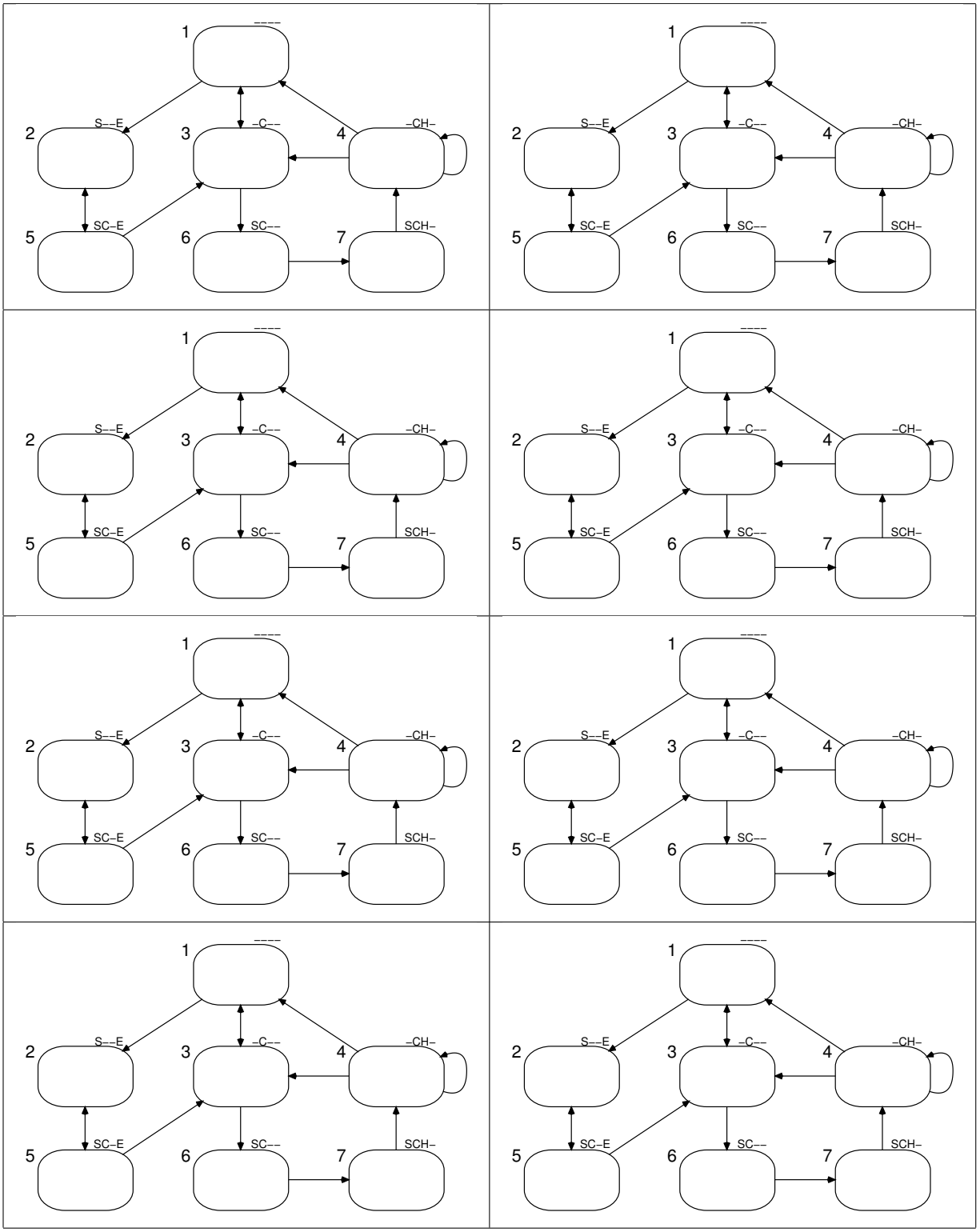
$$f = \Big(\mathbf{E}(\neg \mathit{Started})\mathbf{U}(\mathit{Closed} \wedge \mathbf{EX}\mathit{Heat})\Big) \wedge \Big(\mathbf{EG}(\neg \mathit{Closed} \vee \mathit{Heat} \vee \mathit{Error})\Big)$$

   Damit man die einzelnen Schritte besser nachvollziehen kann, soll (fast) jeder Rekursionsabschnitt in eine eigene Kopie der Kripkestruktur eingezeichnet werden. Nutzen Sie hierfür die acht Kopien der Abbildung.

   Notieren Sie an der Teilabbildung jeweils die Teilformel, die der Algorithmus bearbeitet. Geben Sie – wenn nötig – die SZKs o.ä. an.

2. Bestimmen Sie $S(f) := \{s \in S \mid M, s \models f\}$.

3. Entscheiden Sie, ob $M, 1 \models f$ gilt.


1
____
2
S__E
3
__C__
4
__CH_
5
SC_E
6
SC__
7
SCH_