Fachbereich Informatik
Modellierung und Analyse
von Informationssystemen
Prof. Dr.-Ing. Heiko Mantel

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Klausur für die Prüfung

Modellierung, Spezifikation und Semantik

07. März 2018

Nachname:	
Vorname:	
Matrikelnummer:	
Studiengang:	
E-mail:	
Unterschrift:	

Bitte lesen Sie die Klausurregeln auf der folgenden Seite.

VIEL ERFOLG!

Aufgabe	1	2	3	4	5	6	$\sum$
Punkte	19 (von 19)	14 (von 18)	14 (von 18)	9 (von 13)	9 (von 10)	22 (von 22)	87 (von 100)

Note: 1,3

[redacted]

CSP

Syntax von CSP

Ein Prozess wird durch einen Prozessausdruck aus der Sprache

$$P ::= \mathsf{STOP}_E \mid \mathsf{SKIP}_E \mid (x \to P) \mid (P \sqcap P) \mid (P \,\Box\, P) \mid (P\,;\,P) \mid (P \parallel P) \mid (P \,|||\, P) \mid id \mid (\mu X{:}E.F(X))$$

und durch eine Menge von Gleichungen der Form

$$\{id =_E P \mid id \in \mathsf{Id}\}$$

spezifiziert, wobei Id eine Menge von Prozessbezeichnern ist.

Semantik von CSP

Prämisse	Alphabet	Menge der Spuren
	$\alpha\mathsf{STOP}_E = E$	$\mathsf{traces}(\mathsf{STOP}_E) = \{()\}$
	$\alpha\mathsf{SKIP}_E = E \cup \{\checkmark\}$	$\mathsf{traces}(\mathsf{SKIP}_E) = \{(), (\checkmark)\}$
$x \in \alpha P$	$\alpha(x \to P) = \alpha P$	$\mathsf{traces}((x \to P)) = \{()\} \cup \{(x).t \mid t \in \mathsf{traces}(P)\}$
$\alpha P = \alpha Q$	$\alpha(P \sqcap Q) = \alpha P$	$\mathsf{traces}((P \sqcap Q)) = \mathsf{traces}(P) \cup \mathsf{traces}(Q)$
$\alpha P = \alpha Q$	$\alpha(P \,\Box\, Q) = \alpha P$	$\mathsf{traces}((P \,\Box\, Q)) = \mathsf{traces}(P) \cup \mathsf{traces}(Q)$
$\alpha P = \alpha Q$	$\alpha(P\,;\,Q) = \alpha P$	$\mathsf{traces}((P\,;\,Q)) = \{s \in \mathsf{traces}(P) \mid \checkmark \text{kommt in } s \text{ nicht vor}\}$ $\cup \{s.t \mid s.(\checkmark) \in \mathsf{traces}(P) \wedge t \in \mathsf{traces}(Q)\}$
	$\alpha(P \parallel Q) = \alpha P \cup \alpha Q$	$\mathsf{traces}((P \parallel Q)) = \{t \in (\alpha P \cup \alpha Q)^* \mid (t \upharpoonright \alpha P) \in \mathsf{traces}(P)$ $\wedge (t \upharpoonright \alpha Q) \in \mathsf{traces}(Q)\}$
$\alpha P = \alpha Q$	$\alpha(P \,|||\, Q) = \alpha P$	$\mathsf{traces}((P \,|||\, Q)) = \{s \in (\alpha P \cup \alpha Q)^* \mid \exists t \in \mathsf{traces}(P) : \exists u \in \mathsf{traces}(Q) :$ $s \in \mathsf{interleaving}(t, u)\}$

wobei

$$\mathsf{interleaving} : (\alpha P)^* \times (\alpha P)^* \to \mathcal{P}((\alpha P)^*),$$

$$\mathsf{interleaving}(t, u) := \begin{cases} \{t\} & \text{wenn } u = (), \\ \{u\} & \text{wenn } t = (), \\ \{(x).s \mid s \in \mathsf{interleaving}(t', u)\} & \\ \cup \{(y).s \mid s \in \mathsf{interleaving}(t, u')\} & \text{wenn } t = (x).t' \text{ und } u = (y).u' \end{cases}$$

Intuition von CSP

Prozessausdruck	Intuition des Prozessausdrucks
STOP_E	spezifiziert, dass gar nichts passiert.
SKIP_E	spezifiziert, dass der Prozess terminiert.
$(x \to P)$	spezifiziert, dass der Prozess zuerst am Ereignis x teilnimmt, und sich anschließend wie der durch P spezifizierte Prozess verhält.
$(P \sqcap Q)$	spezifiziert, dass der Prozess sich entweder wie der Prozess P oder wie der Prozess Q verhält. Die Systemumgebung hat keine Kontrolle, ob sich der Prozess entweder wie P oder wie Q verhält.
$(P \,\Box\, Q)$	spezifiziert, dass der Prozess sich entweder wie der Prozess P oder wie der Prozess Q verhält. Die Systemumgebung kann beeinflussen, ob sich der Prozess entweder wie P oder wie Q verhält.
$(P\,;\,Q)$	spezifiziert den Prozess, der sich zunächst wie der durch P spezifizierte Prozess verhält und, nachdem dieser terminiert wäre, sich wie der Prozess verhält, der durch Q spezifiziert ist.
$(P \parallel Q)$	spezifiziert einen Prozess, der die Prozesse, die durch P und Q spezifiziert sind, nebenläufig ablaufen läßt, wobei sich die Prozesse bei Ereignissen, die in beiden Alphabeten vorkommen synchronisieren.
$(P \,|||\, Q)$	spezifiziert einen Prozess, der die Prozesse, die durch P und Q spezifiziert sind, nebenläufig ablaufen läßt, ohne dass sich die Prozesse synchronisieren.

Name: ______________ [redacted]atrikelnr.: ______________

Klausurregeln

- Für die Klausur haben Sie **90 Minuten** Zeit.
- Sobald Sie eine Klausur erhalten haben, dürfen Sie den Raum bis zum Ende der Klausur nicht mehr verlassen. **Ausnahme:** Wenn Sie das WC benutzen wollen, melden Sie sich. Verlassen Sie Ihren Platz *nicht* unaufgefordert. Es darf immer nur ein Studierender außerhalb des Raumes sein. Während des Toilettengangs ist die Klausur bei der Aufsicht zu hinterlegen.
- Schreiben Sie mit Füller oder Kugelschreiber, *nicht* mit Bleistift. Schreiben Sie mit Schwarz oder Blau, *nicht* mit Rot. Schreiben Sie leserlich. Nicht lesbare Antworten werden nicht bewertet.
- Schreiben Sie auf jedes Blatt Ihren **Namen** und Ihre **Matrikelnummer**. Fällt eine Klausur auseinander, so werden Blätter, auf denen kein Name und keine Matrikelnummer stehen, nicht gewertet.
- Antworten Sie bei Aufgaben ohne Vorgaben (wie z. B. Boxen oder zu definierende Symbole) **in ganzen Sätzen**. Schreiben Sie Ihre Antworten in die freien Plätze nach den Teilaufgaben.
- Wenn Ihnen der Platz auf den Klausurblättern nicht ausreicht, erhalten Sie von uns zusätzliches Papier. Verwenden Sie **kein eigenes Papier**. Schreiben Sie Ihren Namen und Ihre Matrikelnummer auf jedes Blatt. Kennzeichnen Sie deutlich, zu welcher Aufgabe welche Antwort gehört.
- Wenn Sie **Schmierpapier** benötigen, erhalten Sie dieses von uns. Verwenden Sie kein eigenes Papier. Schreiben Sie Ihren Namen und Ihre Matrikelnummer auf jedes Blatt. Schreiben Sie „nicht bewerten" gut sichtbar oben auf das Schmierpapier.
- Am **Ende der Klausur** müssen Sie jegliches Papier mit Ausnahme Ihres vorgefertigten Blattes, das als Hilfsmittel erlaubt ist, abgeben. Ein Fortsetzen der Bearbeitung nach Ende der Klausur ist untersagt.
- Als **Hilfsmittel** sind ein beidseitig handbeschriebenes DIN-A4-Blatt (*kein* Ausdruck, *keine* Kopien) und eine Uhr zugelassen. Es sind *keine* Folien, Bücher, elektronischen Geräte, Gesprächspartner, usw. zugelassen. Schalten Sie Ihre Mobiltelefone und Smartwatches aus. Mobiltelefone und Smartwatches müssen während der gesamten Klausur abgeschaltet sein und dürfen nicht in unmittelbarer Reichweite aufbewahrt werden. Betriebsbereite Mobiltelefone und Smartwatches werden als Täuschungsversuch gewertet.
- **Täuschungsversuche** sind zu unterlassen. Versuchte Täuschung kann zu Nichtbestehen der Klausur führen und ggf. weitere Konsequenzen haben. Bei einem Täuschungsversuch kann die Aufsicht die bis dahin erzielten Ergebnisse einsammeln. Die Klausur wird dann höchstens „unter Vorbehalt" weitergeführt.
- Sollten Sie aus gesundheitlichen Gründen die Klausur abbrechen, haben Sie der Aufsicht den Grund Ihres Abbruches zu nennen. Es ist zusätzlich ein tagesaktuelles ärztliches Attest einzureichen. Die Entscheidung über den Rücktritt trifft die Prüfungskommission.
- **Äußere Störungen** sind unverzüglich bei der Aufsicht zu rügen.
- Falls Sie eine **Frage** haben, melden Sie sich. Wir kommen zu Ihnen.

Name:

Aufgabe 1: Formale Modellierung mit Symbolen, Mengen und Funktionen (19 Punkte) Σ 19/19

In dieser Aufgabe sollen Sie das folgende Modell einer Autovermietung erweitern. Im Modell der Autovermietung hat jedes Auto im Bestand einen Hersteller. Die Autovermietung bietet Kleinwagen, Sportwagen und SUVs an. Die Autovermietung wird durch folgende Mengen und Funktionen modelliert:

AUTO	modelliert die Autos, die die Autovermietung anbietet.
AUTO-TYP := {*klein*, *sport*, *suv*}	modelliert die möglichen Autotypen, wobei *klein* Autos vom Typ Kleinwagen modelliert, *sport* Autos vom Typ Sportwagen modelliert, und *suv* Autos vom Typ SUV modelliert.
typ-von : AUTO → AUTO-TYP	modelliert für jedes Auto a, welchen Autotyp das Auto a hat.
HERSTELLER	modelliert die möglichen Hersteller für ein Auto.
hersteller-von : AUTO → HERSTELLER	modelliert für jedes Auto a, welchen Hersteller das Auto a hat.

Ansonsten bleiben diese Mengen und Funktionen unterspezifiziert.

- **Notationskonvention:** Sie können die aus der Vorlesung bekannte Schreibweise (x, xs) als Abkürzung für die Folge $(x, x_1, ..., x_n)$ verwenden, wenn $xs = (x_1, ..., x_n)$ gilt.
- **Zur Information:** Alle vier nachfolgenden Aufgabenteile sind unabhängig voneinander lösbar. Sie dürfen in Ihrer Lösung Mengen, Relationen und Funktionen wiederverwenden, die in den vorigen Aufgabenstellungen deklariert wurden, auch wenn Sie diese Aufgabenteile nicht bearbeitet haben.

(3 P) 3P (A). Modellieren Sie, dass ein Auto einen bestimmten Hersteller hat, durch formale Definition einer Funktion hat-hersteller : (AUTO × HERSTELLER) → $\{\mathfrak{w}, \mathfrak{f}\}$. Die Funktion soll für ein Auto a und einen Hersteller h genau dann $\mathfrak{w}$ zurückgeben, falls das Auto a den Hersteller h hat, und genau dann $\mathfrak{f}$ zurückgeben, falls das Auto a nicht den Hersteller h hat.

Antwort:

hat-hersteller : (AUTO × HERSTELLER) → {w, f}

$$\text{hat-hersteller}(a, h) := \begin{cases} w & \text{, wenn gilt hersteller-von}(a) = h \\ f & \text{, sonst} \end{cases}$$ ✓

(4 P) 4P (B). Modellieren Sie die Kleinwagen, die die Autovermietung anbietet, durch formale Definition einer Menge KLEINWAGEN ⊆ AUTO.

Antwort:

KLEINWAGEN := $\{ a \in \text{AUTO} \mid$ ~~ALSO~~ $\text{typ-von}(a) = \text{klein} \}$ ✓

Name: ________________________, Matrikelnr.: __________

(C). Modellieren Sie, dass zwei Autos der Autovermietung ähnlich sind, durch formale Definition einer Relation ÄHNLICHE-AUTOS $\subseteq$ (AUTO $\times$ AUTO). Zwei Autos sind ähnlich, wenn sie den selben Hersteller haben, oder denselben Autotyp haben. (5 P)

5P

Antwort:

ÄHNLICHE-AUTOS $\subseteq$ (AUTO $\times$ AUTO)

(a, a') $\in$ ÄHNLICHE-AUTOS genau dann wenn gilt,

dass typ-von (a) = typ-von (a') $\vee$ hersteller-von (a) = hersteller-von (a')

Also:

ÄHNLICHE-AUTOS := {(a, a') $\in$ AUTO $\times$ AUTO | typ-von (a) = typ-von (a') $\vee$ hersteller-von (a) = hersteller-von (a')}

✓

(D). Modellieren Sie, dass verschiedene Autos denselben Hersteller haben, durch formale Definition einer rekursiven Funktion autos-von-hersteller : (AUTO* $\times$ HERSTELLER) $\rightarrow$ AUTO*. Die Funktion soll für eine gegebene Folge von Autos die Folge aller Autos zurückgeben, die vom gegebenem Hersteller sind. (7 P)

7P

Antwort:

autos-von-hersteller : (AUTO* $\times$ HERSTELLER) $\rightarrow$ AUTO*

autos-von-hersteller ((), h) := ()

autos-von-hersteller ((x, xs), h) := $\begin{cases} (x, \text{autos-von-hersteller}(xs, h)), & \text{wenn hersteller-von}(x) = h \text{ gilt.} \\ \text{autos-von-hersteller}(xs, h), & \text{sonst} \end{cases}$

✓

Name: ______________________, Matrikelnr.: ______________

Aufgabe 2: Formale Modellierung von Anforderungen (18 Punkte) $\sum$ 14/18

In dieser Aufgabe sollen Sie Anforderungen an das Modell der Autovermietung aus Aufgabe 1 modellieren. Dafür ist das Modell aus Aufgabe 1 um die folgenden Mengen und Funktionen erweitert:

STANDORT	modelliert die verschiedenen Standorte der Autovermietung.
autos-an : STANDORT $\to \mathcal{P}$(AUTO)	modelliert für jeden Standort s die Autos der Autovermietung, die am Standort s stationiert sind.
KUNDE	modelliert die Kunden der Autovermietung.
registriert-in : STANDORT $\to \mathcal{P}$(KUNDE)	modelliert für jeden Standort s die Kunden der Autovermietung, die für den Standort s registriert sind.

Ansonsten bleiben diese Mengen und Funktionen unterspezifiziert.

- **Zur Information:** Sie können diese Aufgabe auch dann lösen, wenn Sie Aufgabe 1 nicht bearbeitet haben. Alle vier nachfolgenden Aufgabenteile sind unabhängig voneinander lösbar. Sie dürfen in Ihrer Lösung alle Mengen, Relationen und Funktionen wiederverwenden, die in der Aufgabenstellung von Aufgabe 1 (inklusive Teilaufgaben) deklariert wurden, auch wenn Sie Aufgabe 1 nicht vollständig bearbeitet haben.
- **Notationskonvention:** Sie können die aus der Vorlesung bekannte Schreibweise (x, xs) als Abkürzung für die Folge $(x, x_1, ..., x_n)$ verwenden, wenn $xs = (x_1, ..., x_n)$ gilt.

(4 P) 4P (A). Die Anforderung „An jedem Standort sind mindestens 2 Autos stationiert" sei durch folgende Relation modelliert:

ANFORDERUNG1 $\subseteq$ (STANDORT $\to \mathcal{P}$(AUTO))

autos-an $\in$ ANFORDERUNG1 genau dann, wenn die prädikatenlogische Formel φ_1 gilt.

Definieren Sie die Formel φ_1 in Prädikatenlogik mit Hilfe von mathematischen Konzepten, die in der Vorlesung behandelt wurden.

Antwort:

φ_1:=

$\forall s \in \text{STANDORT} : \exists a, a' \in \text{AUTO} : \neg a = a' \wedge$ ~~autos~~ $a, a' \in \text{autos-an}(s)$ ✓

(4 P) 4P (B). Die Anforderung „An jedem Standort ist mindestens 1 Kleinwagen stationiert" sei durch folgende Relation modelliert:

ANFORDERUNG2 $\subseteq$ (STANDORT $\to \mathcal{P}$(AUTO))

autos-an $\in$ ANFORDERUNG2 genau dann, wenn die prädikatenlogische Formel φ_2 gilt.

Definieren Sie die Formel φ_2 in Prädikatenlogik mit Hilfe von mathematischen Konzepten, die in der Vorlesung behandelt wurden.

Antwort:

φ_2:=

$\forall s \in \text{STANDORT} : \exists a \in \text{AUTO} : \text{typ-von}(a) = \text{klein} \wedge a \in \text{autos-an}(s)$ ✓

Name: ____________________, Matrikelnr.:

(C). Die Anforderung „Wenn an einem Standort mindestens 10 Kunden registriert sind, dann sind an diesem Standort mindestens 4 Autos stationiert" sei durch folgende Relation modelliert: (5 P)

5P

ANFORDERUNG3 $\subseteq$ (STANDORT $\rightarrow \mathcal{P}$(AUTO))

autos-an $\in$ ANFORDERUNG3 genau dann, wenn die prädikatenlogische Formel φ_3 gilt.

Definieren Sie die Formel φ_3 in Prädikatenlogik mit Hilfe von mathematischen Konzepten, die in der Vorlesung behandelt wurden.

Antwort:

$\varphi_3 :=$

$\forall s \in$ STANDORT: $[|\text{registriert-in}(s)| \geq 10]$
$\Rightarrow [|\text{autos-an}(s)| \geq 4]$ ✓

Betrag gibt die Anzahl an Elementen in Menge zurück

(D). Die Anforderung „Jedes Auto ist an genau einem Standort stationiert" sei durch folgende Relation modelliert: (5 P)

1P

ANFORDERUNG4 $\subseteq$ (STANDORT $\rightarrow \mathcal{P}$(AUTO))

autos-an $\in$ ANFORDERUNG4 genau dann, wenn die prädikatenlogische Formel φ_4 gilt.

Definieren Sie die Formel φ_4 in Prädikatenlogik mit Hilfe von mathematischen Konzepten, die in der Vorlesung behandelt wurden.

Antwort:

$\varphi_4 :=$

(✓)

$\forall a \in$ AUTO: $\exists s, s' \in$ STANDORT: $\neg(a \in \text{autos-an}(s) \wedge a \in \text{autos-an}(s'))$

(✓) Syntax

↯ [illegible] ein Standort

↯ semantisch [illegible]

[redacted]______, Matrikelnr.: ________________

Aufgabe 3: Syntax und Semantik (18 Punkte) ℰ 14 / 18

- ▶ **Zur Information:** Alle zwei nachfolgenden Aufgabenteile sind unabhängig voneinander lösbar.
- ▶ **Zur Information:** Alle Kalkülregeln der operationalen Semantik der Programmiersprache IMP finden Sie zusammengefasst auf dem Beiblatt „Beiblatt zur Klausur Modellierung, Spezifikation und Semantik".

(6 P) (A). Sei Var := {x}. Seien $\sigma, \sigma' \in \Sigma$ beliebige Zustände, so dass $\sigma(\mathtt{x}) = 1$ und $\sigma' = \sigma[\mathtt{x}\backslash 2]$ gelten.

2 P

■ Vervollständigen Sie die nachfolgende Herleitung des Urteils

$$\langle \mathtt{x} := (\mathtt{x} \oplus 1)\, ;\, \mathtt{skip},\ \sigma\rangle \to \sigma'$$

im Kalkül $\mathcal{C}$ für die Programmiersprache IMP. Füllen Sie für Ihre Antwort nur die untenstehenden Boxen aus und modifizieren Sie keine anderen Teile der Aufgabenstellung.

Antwort:

$$\boxed{r_j} \quad \dfrac{\dfrac{\vdots\ \mathcal{H}_1}{\langle \mathtt{x} := (\mathtt{x}\oplus 1),\ \sigma\rangle \to \sigma'} \qquad \boxed{\langle \mathtt{x} := (\mathtt{x}\oplus 1);\mathtt{skip},\ \sigma\rangle \to \sigma'} \ \{\}\ \mathrm{f} \qquad \langle \mathtt{skip},\ \sigma'\rangle \to \sigma'}{\langle \mathtt{x} := (\mathtt{x}\oplus 1)\, ;\, \mathtt{skip},\ \sigma\rangle \to \sigma'}$$

wobei $\mathcal{H}_1$ die folgende Herleitung ist:

$$\boxed{r_{:=}} \quad \dfrac{\Big\{\ \langle (x+1),\ \sigma\rangle \Downarrow 2\ \Big\}_{\mathrm{f}} \quad (\checkmark)}{\langle \mathtt{x} := (\mathtt{x}\oplus 1),\ \sigma\rangle \to \sigma'} \quad \boxed{\sigma' = \sigma[x\backslash 2]}$$

Name: ______________________, Matrikelnr.:

(B). In dieser Aufgabe erweitern Sie die Programmiersprache IMP. Folgende Wertebereiche werden verwendet: (12 P)

12 P

Num	die Zahlen,	AExp	die arithmetischen Ausdrücke,
Bool	die Wahrheitswerte,	BExp	die Booleschen Ausdrücke,
Var	die Programmvariablen,	DCom	die Kommandos.

Die Sprache DIMP erweitert IMP um das im Folgenden grau hervorgehobene Kommando. Der Wertebereich DCom ist durch folgende Grammatik in BNF definiert, wobei $a \in$ AExp, $b \in$ BExp und $X \in$ Var:

$$c ::= \texttt{skip} \mid X := a \mid c;c \mid \texttt{if } b \texttt{ then } c \texttt{ else } c \texttt{ fi} \mid \texttt{while } b \texttt{ do } c \texttt{ od} \mid \boxed{\texttt{do } c \texttt{ while } b}$$

Die Intuition des Kommandos do c while b ist:

- **Fall 1:** Wenn der Ausdruck b im Zustand nach der Ausführung des Kommandos c zu false auswertet, dann terminiert die Ausführung des Kommandos do c while b im Zustand nach der Ausführung des Kommandos c.
- **Fall 2:** Wenn der Ausdruck b im Zustand nach der Ausführung des Kommandos c zu true auswertet, dann wird das Kommando do c while b im Zustand nach der Ausführung des Kommandos c erneut ausgeführt.

■ Definieren Sie die Kalkülregel rdwhf zur Herleitung von Instanzen des Urteils $\langle \texttt{do } c \texttt{ while } b,\ \sigma\rangle \to \sigma'$ im Fall, dass nach der Ausführung des Kommandos c der Ausdruck b zu false auswertet (Fall 1).

Antwort:

$$\text{rdwhf}\ \frac{\langle c, \sigma\rangle \to \sigma' \quad \cancel{\langle \text{do } c \text{ while } b, \sigma''\rangle \to \sigma'} \quad \langle b, \sigma'\rangle \Downarrow \text{false}}{\langle \texttt{do } c \texttt{ while } b,\ \sigma\rangle \to \sigma'}$$

■ Definieren Sie die Kalkülregel rdwht zur Herleitung von Instanzen des Urteils $\langle \texttt{do } c \texttt{ while } b,\ \sigma\rangle \to \sigma'$ im Fall, dass nach der Ausführung des Kommandos c der Ausdruck b zu true auswertet (Fall 2).

Antwort:

$$\text{rdwht}\ \frac{\langle c, \sigma\rangle \to \sigma'' \quad \langle \text{do } c \text{ while } b, \sigma''\rangle \to \sigma' \quad \langle b, \sigma''\rangle \Downarrow \text{true}}{\langle \texttt{do } c \texttt{ while } b,\ \sigma\rangle \to \sigma'}$$

[redacted] ________, Matrikelnr.: __________

Σ 9/13

Aufgabe 4: Determinismus von Programmiersprachen (13 Punkte)

In dieser Aufgabe führen Sie einen Teilbeweis für den Determinismus von Ausführungen der Programmiersprache ASM für Stackoperationen. Die Syntax und Semantik der Programmiersprache ASM wird im restlichen Aufgabentext definiert. Dabei werden folgende Wertebereiche verwendet:

Val	die Stackwerte,	$\mathsf{Com}_{\mathsf{ASM}}$	die Kommandos.
$\Sigma_{\mathsf{ASM}} := \mathsf{Val}^*$	die Stackzustände,		

Syntax. Der Wertebereich $\mathsf{Com}_{\mathsf{ASM}}$ ist durch folgende Grammatik in BNF definiert, wobei $v \in \mathsf{Val}$:

$$c ::= \texttt{push}\ v \mid \texttt{pop} \mid c\,;c$$

Die Intuition der einzelnen Kommandos ist:

- $\texttt{push}\ v$ legt den Stackwert v oben auf den Stack und lässt den Stack ansonsten unverändert
- $\texttt{pop}$ kann nur bei einem nicht-leeren Stack ausgeführt werden und entfernt das oberste Element eines nicht-leeren Stacks ohne den restlichen Stack zu verändern
- $c_1; c_2$ führt zunächst das Kommando c_1 aus und anschließend das Kommando c_2

Semantik. Die Semantik von ASM ist mit Hilfe des Urteils $\langle c,\ \sigma\rangle \rightrightarrows \sigma'$ definiert. Die Intuition des Urteils $\langle c,\ \sigma\rangle \rightrightarrows \sigma'$ ist, dass das Kommando c im Stackzustand σ zum Stackzustand σ' auswertet.
Das folgende Kalkül definiert die Semantik von ASM wobei die Schreibweise (x, xs) wie in der Vorlesung als Abkürzung für die Folge $(x, x_1, ..., x_n)$ verwendet wird wenn $xs = (x_1, ..., x_n)$ gilt:

$$\mathsf{rASMpush}\ \frac{}{\langle \texttt{push}\ v,\ \sigma\rangle \rightrightarrows \sigma'}\ \sigma' = (v, \sigma) \qquad \mathsf{rASMpop}\ \frac{}{\langle \texttt{pop},\ \sigma\rangle \rightrightarrows \sigma'}\ \sigma = (v, \sigma')$$

$$\mathsf{rASMseq}\ \frac{\langle c_1,\ \sigma\rangle \rightrightarrows \sigma'' \quad \langle c_2,\ \sigma''\rangle \rightrightarrows \sigma'}{\langle c_1\,; c_2,\ \sigma\rangle \rightrightarrows \sigma'}$$

Determinismus. Die Ausführung von ASM ist deterministisch, d.h. es gilt:

$$\forall c \in \mathsf{Com}_{\mathsf{ASM}}:\ P(c)$$

wobei $P(c) := \forall \sigma, \sigma', \sigma'' \in \Sigma_{\mathsf{ASM}} : (\langle c,\ \sigma\rangle \rightrightarrows \sigma'$ ist herleitbar $\wedge \langle c,\ \sigma\rangle \rightrightarrows \sigma''$ ist herleitbar $) \Rightarrow \sigma' = \sigma''$.

■ Vervollständigen Sie den Beweis der folgenden beiden Aussagen auf dieser und der nächsten Seite.
- $\forall v \in \mathsf{Val}:\ P(\texttt{push}\ v)$
- $\forall c_1, c_2 \in \mathsf{Com}_{\mathsf{ASM}}:\ P(c_1) \wedge P(c_2) \Rightarrow P(c_1\,; c_2)$

Antwort:

- $\forall v \in \mathsf{Val}:\ P(\texttt{push}\ v)$

Sei $v \in \mathsf{Val}$ beliebig aber fest. Es ist noch zu zeigen, dass $P(\texttt{push}\ v)$ gilt. Seien die Stackzustände $\sigma, \sigma', \sigma'' \in \Sigma_{\mathsf{ASM}}$ beliebig aber fest. Angenommen es gelten

✓ ~~rASMpush ...~~ $\langle \text{push } v, \sigma\rangle \rightrightarrows \sigma'$ ist herleitbar

und

~~rASMpush ...~~ $\langle \text{push } v, \sigma\rangle \rightrightarrows \sigma''$ ist herleitbar

. Da die Regel $\boxed{\text{rASMpush}}$ (✓) die letzte Regel in den Herleitungen von $\langle \texttt{push}\ v,\ \sigma\rangle \rightrightarrows \sigma'$ und $\langle \texttt{push}\ v,\ \sigma\rangle \rightrightarrows \sigma''$ sein muss, haben die Herleitungen die folgende Form:

- Form der Herleitung von $\langle \texttt{push}\ v,\ \sigma\rangle \rightrightarrows \sigma'$:

$$\text{rASMpush}\ \frac{}{\langle \text{push } v, \sigma\rangle \rightrightarrows \sigma'}\ \sigma' = (v, \sigma)$$

✓

Name: ________________________, Matrikelnr.: [redacted]

– Form der Herleitung von $\langle \texttt{push}\ v,\ \sigma\rangle \rightrightarrows \sigma''$:

$$\text{ASMpush}\ \frac{}{\langle \text{push}\ v, \sigma\rangle \rightrightarrows \sigma''}\quad \sigma'' = (v, \sigma)$$

✓

Aus $\boxed{\sigma' = (v, \sigma)}$ (v) und $\boxed{\sigma'' = (v, \sigma)}$ (v) folgt, dass $\sigma' = \sigma''$ gilt. □

- $\forall c_1, c_2 \in \mathsf{Com}_{\mathrm{ASM}}:\ P(c_1) \wedge P(c_2) \Rightarrow P(c_1\,;c_2)$

Seien $c_1, c_2 \in \mathsf{Com}_{\mathrm{ASM}}$ beliebig aber fest. Angenommen es gelten $P(c_1)$ und $P(c_2)$. Es ist noch zu zeigen, dass $P(c_1\,;c_2)$ gilt. Seien die Stackzustände $\sigma, \sigma', \sigma'' \in \Sigma_{\mathrm{ASM}}$ beliebig aber fest. Angenommen es gelten

✓ $\boxed{\langle c_1; c_2, \sigma\rangle \rightrightarrows \sigma' \text{ ist herleitbar}}$ und

$\boxed{\langle c_1; c_2, \sigma\rangle \rightrightarrows \sigma'' \text{ ist herleitbar}}$

Da die Regel $\boxed{\text{rASMseq}}$ (v) die letzte Regel in den Herleitungen von $\langle c_1\,;c_2,\ \sigma\rangle \rightrightarrows \sigma'$ und $\langle c_1\,;c_2,\ \sigma\rangle \rightrightarrows \sigma''$ sein muss, gibt es Stackzustände $\sigma^{*\prime}, \sigma^{*\prime\prime} \in \Sigma_{\mathrm{ASM}}$ sodass die Herleitungen die folgende Form haben:

– Form der Herleitung von $\langle c_1\,;c_2,\ \sigma\rangle \rightrightarrows \sigma'$:

$$\text{rASMseq}\ \frac{\vdots \atop \langle c_1, \sigma\rangle \rightrightarrows \sigma^{*'''} \qquad \vdots \atop \langle c_2, \sigma'''\rangle \rightrightarrows \sigma'}{\langle c_1; c_2, \sigma\rangle \rightrightarrows \sigma'}$$

↯ keine Herleitung (v)

– Form der Herleitung von $\langle c_1\,;c_2,\ \sigma\rangle \rightrightarrows \sigma''$:

$$\text{rASMseq}\ \frac{\vdots \atop \langle c_1, \sigma^{*}\rangle \rightrightarrows \sigma'''' \qquad \vdots \atop \langle c_2, \sigma''''\rangle \rightrightarrows \sigma''}{\langle c_1; c_2, \sigma\rangle \rightrightarrows \sigma''}$$

↯ keine Herleitung (v)

Aus $P(c_1)$ folgt, dass $\boxed{\langle c_1, \sigma\rangle \rightrightarrows \sigma''' \text{ bzw. } \langle c_1, \sigma\rangle \rightrightarrows \sigma''''}$ gilt. Damit folgt aus $\boxed{\sigma''' = \sigma''''}$, dass $\sigma' = \sigma''$ gilt. □

[redacted], Matrikelnr.: ____________

Aufgabe 5: Prozessalgebra CSP (10 Punkte) Σ 9/10

- **Zur Information:** Alle zwei nachfolgenden Aufgabenteile sind unabhängig voneinander lösbar.
- **Zur Information:** Die Semantik der Prozessausdrücke der Sprache CSP finden Sie auf dem Beiblatt „Beiblatt zur Klausur Modellierung, Spezifikation und Semantik“.

4 (4 P) (A). Seien $E_A := \{x, y\}$ und der Prozessausdruck P_A wie folgt definiert:

$$P_A := ((x \to \mathsf{STOP}_{E_A}) \sqcap (y \to (y \to \mathsf{SKIP}_{E_A})))$$

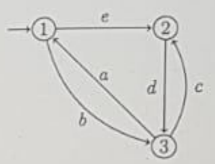
■ Geben Sie die durch den Prozessausdruck P_A definierte Menge von Spuren $\mathsf{traces}(P_A)$ an.

Antwort:

$\mathsf{traces}(P_A) =$ traces $((x \to STOP_{E_A})) \cup$ traces $((y \to (y \to SKIP_{E_A})))$
$= \{()\} \cup \{(x).t \mid t \in traces(STOP_{E_A})\}$
$\cup \{()\} \cup \{(y).t' \mid t \in traces((y \to SKIP_{E_A}))\}$
$= \{()\} \cup \{(x).t \mid t \in \{()\}\}$
$\cup \{(y).t' \mid t' \in \{()\} \cup \{(y).t'' \mid t'' \in \{(), (\checkmark)\}\}\}$ ✓

5 (6 P) (B). Sei folgendes Transitionssystem TS_B und die zugehörige Visualisierung als Diagramm gegeben.

$$\begin{aligned} TS_B &:= (S_B, S_0^B, E_B, \to_B), \\ S_B &:= \{1, 2, 3\} \\ S_0^B &:= \{1\} \\ E_B &:= \{a, b, c, d, e\} \\ \to_B &:= \{(1, e, 2), (1, b, 3), (2, d, 3), \\ &\quad (3, c, 2), (3, a, 1)\} \end{aligned}$$



■ Definieren Sie den Prozessbezeichner P_B durch ein Gleichungssystem aus Prozessausdrücken, sodass der Prozessausdruck P_B unter dem Gleichungssystem aus Prozessausdrücken den Prozess $(E_B, \text{E-Traces}(TS_B))$ spezifiziert.

Antwort:

$P_B := ((e \to P_2) \sqcap (b \to P_3))$
$P_2 := (d \to P_3)$
$P_3 := ((a \to P_B) \sqcap (c \to P_2))$ (✓)

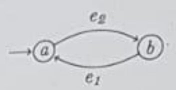
[redacted] ________, Matrikelnr.: ________ $\sum 22/22$

Aufgabe 6: Transitionssysteme und nebenläufige Ausführung (22 Punkte)

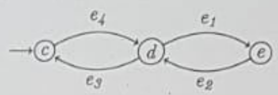
▸ **Zur Information:** Alle zwei nachfolgenden Aufgabenteile sind unabhängig voneinander lösbar.

10 (10 P) (A). Betrachten Sie die beiden nachfolgenden Transitionssysteme TS_1 und TS_2. Die Transitionssysteme werden jeweils durch das daneben stehende Diagramm veranschaulicht.

$$
\begin{aligned}
TS_1 &:= (S^{TS_1}, S_0^{TS_1}, E^{TS_1}, \rightarrow^{TS_1}) \\
S^{TS_1} &:= \{a, b\} \\
S_0^{TS_1} &:= \{a\} \\
E^{TS_1} &:= \{e_1, e_2\} \\
\rightarrow^{TS_1} &:= \{(a, e_2, b), (b, e_1, a)\}
\end{aligned}
$$



$$
\begin{aligned}
TS_2 &:= (S^{TS_2}, S_0^{TS_2}, E^{TS_2}, \rightarrow^{TS_2}) \\
S^{TS_2} &:= \{c, d, e\} \\
S_0^{TS_2} &:= \{c\} \\
E^{TS_2} &:= \{e_1, e_2, e_3, e_4\} \\
\rightarrow^{TS_2} &:= \{(c, e_4, d), (d, e_3, c), (d, e_1, e), (e, e_2, d)\}
\end{aligned}
$$



■ Vervollständigen Sie die unten angegebene message-passing Komposition $MPC := (S^{MPC}, S_0^{MPC}, E^{MPC}, \rightarrow^{MPC})$ von TS_1 und TS_2. Geben Sie dabei die Menge $\rightarrow^{MPC}$ durch explizite Aufzählung ihrer Elemente an (extensionale Mengendefinition).

Antwort:

$$
\begin{aligned}
MPC &:= (S^{MPC}, S_0^{MPC}, E^{MPC}, \rightarrow^{MPC}) \\
S^{MPC} &:= S^{TS_1} \times S^{TS_2} \\
S_0^{MPC} &:= \{(a, c)\} \\
E^{MPC} &:= E^{TS_1} \cup E^{TS_2}
\end{aligned}
$$

$\rightarrow^{MPC} :=$ $\{((s_1, s_2), e, (s_1', s_2')) \in S^{MPC} \times E^{MPC} \times S^{MPC} \mid$
$((s_1, e, s_1') \in \rightarrow^{TS_1} \wedge e \in (E^{TS_1} \setminus E^{TS_2}) \wedge s_2 = s_2')$
$\vee (s_2, e, s_2') \in \rightarrow^{TS_2} \wedge e \in (E^{TS_2} \setminus E^{TS_1}) \wedge s_1 = s_1')$
$\vee ((s_1, e, s_1') \in \rightarrow^{TS_1} \wedge (s_2, e, s_2') \in \rightarrow^{TS_2} \wedge e \in (E^{TS_1} \cup E^{TS_2}))\}$

$= \{((a, c), e_4, (a, d)), ((b, c), e_4, (b, d)),$
$((a, d), e_3, (a, c)), ((b, d), e_3, (b, c)),$
$((b, d), e_1, (a, e)), ((a, e), e_2, (b, d))\}$ ✓

Name: ______________________, Matrikelnr.: ______

(B). Betrachten Sie die folgenden Eigenschaften über Transitionssysteme $TS_B := (S^{TS_B}, S_0^{TS_B}, E^{TS_B}, \rightarrow^{TS_B})$. (12 P) 12

$P_1(TS_B) := \exists s' \in S^{TS_B} : \forall s \in (S^{TS_B} \setminus \{s'\}) : \exists e \in E^{TS_B} : (s, e, s') \in \rightarrow^{TS_B}$ Alle Zustände verweisen auf s'

$P_2(TS_B) := \exists \underset{q_1}{s} \in S^{TS_B} : \underbrace{\forall s' \in S^{TS_B}}_{q_0} : \forall e \in E^{TS_B} : \neg(\underset{q_1 \to q_0}{(s, e, s')} \in \rightarrow^{TS_B})$ Es gibt s ohne ausgehende Kanten

Widerlegen Sie, dass die Implikation $P_1(TS_B) \implies P_2(TS_B)$ für beliebige Transitionssysteme TS_B gilt. Antworten Sie durch

- Angabe eines Gegenbeispiels TS_B, sodass die Implikation $P_1(TS_B) \implies P_2(TS_B)$ nicht gilt,
- ein strukturiertes Argument in 1–4 Sätzen dafür, dass $P_1(TS_B)$ gilt, und
- ein strukturiertes Argument in 1–4 Sätzen dafür, dass $P_2(TS_B)$ nicht gilt.

Antwort:

Gegenbeispiel TS_B, für das $P_1(TS_B) \implies P_2(TS_B)$ nicht gilt:

$TS_B := (S^{TS_B}, S_0^{TS_B}, E^{TS_B}, \rightarrow^{TS_B})$

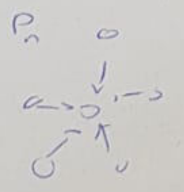
$S^{TS_B} :=$ $\{q_0, q_1\}$

$S_0^{TS_B} :=$ $\{q_0\}$

$E^{TS_B} :=$ $\{a\}$

$\rightarrow^{TS_B} :=$ $\{(q_0, a, q_1), (q_1, a, q_0)\}$

✓



– Die Aufgabe geht auf der nachfolgenden Seite weiter! –

[redacted] ____________, Matrikelnr.: ____________

Argument, dass $P_1(TS_B)$ gilt (1–4 Sätze):

$P_1(TS_B)$ gilt, weil es mindestens einen ~~Knoten~~ Zustand s' gibt aus der Menge S^{TS_B} für den gilt dass alle ~~anderen~~ Zustände außer s' eine Kante zu diesem ~~Knoten~~ Zustand haben.
~~Die Menge~~ Hier gibt es zwei Zustände für die die Aussage zutrifft: q_0 und q_1, weil sie ~~die~~ zusammen die Menge S^{TSB} bilden und ~~auf den~~ zu dem jeweilig anderen eine Transition haben.
~~Somit besteht eine Transition zwischen allen und~~

Argument, dass $P_2(TS_B)$ nicht gilt (1–4 Sätze):

$P_2(TS_B)$ gilt nicht, weil es keinen Zustand gibt der keine Transition zu allen anderen Zuständen hat
Die Menge S^{TS_B} besteht aus zwei Elementen.
folglich gibt es zwei Möglichkeiten ~~wann die~~ ~~Aus~~ damit die Aussage $P_2(TS_B)$ erfüllt ist.
Fall$_1$: q_0 hat keine Transition zu allen anderen Zuständen.
Da aber $\{q_1\} = E^{TS_B} \setminus \{q_0\}$ ist und eine Transition (q_0, a, q_1) existiert gilt dies nicht
Fall$_2$: q_1 hat keine Transition zu allen anderen Zuständen
Da aber $\{q_0\} = E^{TS_B} \setminus \{q_1\}$ ist und eine Transition (q_1, a, q_0) existiert, gibt diese auch nicht
Aus Fall 1 und Fall 2 folgt $P_2(TS_B)$ gilt nicht

Name: ____________________, Matrikeln[redacted]

Beiblatt zur Klausur
Modellierung, Spezifikation und Semantik

IMP

Kalkül $\mathcal{A}$: Regeln für die Herleitung des Urteils $\langle a,\ \sigma\rangle \Downarrow n$ für $a \in \mathsf{AExp}$.

$$\text{rNum}\ \frac{}{\langle n,\ \sigma\rangle \Downarrow n} \qquad \text{rVar}\ \frac{}{\langle X,\ \sigma\rangle \Downarrow n}\ \sigma(X)=n \qquad \text{r}\oplus\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1 \oplus a_2),\ \sigma\rangle \Downarrow n}\ n=n_1+n_2$$

$$\text{r}\ominus\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1 \ominus a_2),\ \sigma\rangle \Downarrow n}\ n=n_1-n_2 \qquad \text{r}\odot\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1 \odot a_2),\ \sigma\rangle \Downarrow n}\ n=n_1\cdot n_2$$

Kalkül $\mathcal{B}$: Regeln für die Herleitung des Urteils $\langle b,\ \sigma\rangle \Downarrow t$ für $b \in \mathsf{BExp}$.

$$\text{rtrue}\ \frac{}{\langle \mathtt{true},\ \sigma\rangle \Downarrow \mathtt{true}} \qquad \text{rfalse}\ \frac{}{\langle \mathtt{false},\ \sigma\rangle \Downarrow \mathtt{false}}$$

$$\text{reqt}\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1\ \mathsf{eq}\ a_2),\ \sigma\rangle \Downarrow \mathtt{true}}\ n_1=n_2 \qquad \text{reqf}\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1\ \mathsf{eq}\ a_2),\ \sigma\rangle \Downarrow \mathtt{false}}\ \neg(n_1=n_2)$$

$$\text{rleqt}\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1\ \mathsf{leq}\ a_2),\ \sigma\rangle \Downarrow \mathtt{true}}\ n_1\leq n_2 \qquad \text{rleqf}\ \frac{\langle a_1,\ \sigma\rangle \Downarrow n_1 \quad \langle a_2,\ \sigma\rangle \Downarrow n_2}{\langle (a_1\ \mathsf{leq}\ a_2),\ \sigma\rangle \Downarrow \mathtt{false}}\ \neg(n_1\leq n_2)$$

$$\text{rnott}\ \frac{\langle b,\ \sigma\rangle \Downarrow \mathtt{false}}{\langle \mathsf{not}\, b,\ \sigma\rangle \Downarrow \mathtt{true}} \qquad \text{rnotf}\ \frac{\langle b,\ \sigma\rangle \Downarrow \mathtt{true}}{\langle \mathsf{not}\, b,\ \sigma\rangle \Downarrow \mathtt{false}}$$

$$\text{randt}\ \frac{\langle b_1,\ \sigma\rangle \Downarrow \mathtt{true} \quad \langle b_2,\ \sigma\rangle \Downarrow \mathtt{true}}{\langle (b_1\ \mathsf{and}\ b_2),\ \sigma\rangle \Downarrow \mathtt{true}} \qquad \text{randf1}\ \frac{\langle b_1,\ \sigma\rangle \Downarrow \mathtt{false}}{\langle (b_1\ \mathsf{and}\ b_2),\ \sigma\rangle \Downarrow \mathtt{false}} \qquad \text{randf2}\ \frac{\langle b_2,\ \sigma\rangle \Downarrow \mathtt{false}}{\langle (b_1\ \mathsf{and}\ b_2),\ \sigma\rangle \Downarrow \mathtt{false}}$$

$$\text{rort1}\ \frac{\langle b_1,\ \sigma\rangle \Downarrow \mathtt{true}}{\langle (b_1\ \mathsf{or}\ b_2),\ \sigma\rangle \Downarrow \mathtt{true}} \qquad \text{rort2}\ \frac{\langle b_2,\ \sigma\rangle \Downarrow \mathtt{true}}{\langle (b_1\ \mathsf{or}\ b_2),\ \sigma\rangle \Downarrow \mathtt{true}} \qquad \text{rorf}\ \frac{\langle b_1,\ \sigma\rangle \Downarrow \mathtt{false} \quad \langle b_2,\ \sigma\rangle \Downarrow \mathtt{false}}{\langle (b_1\ \mathsf{or}\ b_2),\ \sigma\rangle \Downarrow \mathtt{false}}$$

Kalkül $\mathcal{C}$: Regeln für die Herleitung des Urteils $\langle c,\ \sigma\rangle \to \sigma'$ für $c \in \mathsf{Com}$.

$$\text{rsk}\ \frac{}{\langle \mathtt{skip},\ \sigma\rangle \to \sigma} \qquad \text{r:=}\ \frac{\langle a,\ \sigma\rangle \Downarrow n}{\langle X := a,\ \sigma\rangle \to \sigma'}\ \sigma'=\sigma[X\backslash n] \qquad \text{r;}\ \frac{\langle c_1,\ \sigma\rangle \to \sigma'' \quad \langle c_2,\ \sigma''\rangle \to \sigma'}{\langle c_1\,;\, c_2,\ \sigma\rangle \to \sigma'}$$

$$\text{rift}\ \frac{\langle b,\ \sigma\rangle \Downarrow \mathtt{true} \quad \langle c_1,\ \sigma\rangle \to \sigma'}{\langle \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2\ \mathtt{fi},\ \sigma\rangle \to \sigma'} \qquad \text{riff}\ \frac{\langle b,\ \sigma\rangle \Downarrow \mathtt{false} \quad \langle c_2,\ \sigma\rangle \to \sigma'}{\langle \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2\ \mathtt{fi},\ \sigma\rangle \to \sigma'}$$

$$\text{rwht}\ \frac{\langle b,\ \sigma\rangle \Downarrow \mathtt{true} \quad \langle c,\ \sigma\rangle \to \sigma'' \quad \langle \mathtt{while}\ b\ \mathtt{do}\ c\ \mathtt{od},\ \sigma''\rangle \to \sigma'}{\langle \mathtt{while}\ b\ \mathtt{do}\ c\ \mathtt{od},\ \sigma\rangle \to \sigma'} \qquad \text{rwhf}\ \frac{\langle b,\ \sigma\rangle \Downarrow \mathtt{false}}{\langle \mathtt{while}\ b\ \mathtt{do}\ c\ \mathtt{od},\ \sigma\rangle \to \sigma}$$