Christian-Albrechts-Universität zu Kiel
Institut für Informatik
Lehrstuhl für Programmiersprachen und Übersetzerkonstruktion
Prof. Dr. M. Hanus, F. Reck

Institut für Informatik

11. Übung zur Vorlesung „Informatik I“

Abgabe am Freitag, 23.1.2009, 14 Uhr

Hinweis: Bitte denken Sie daran, sich rechtzeitig bis zum *25. Januar 2009* elektronisch unter

`https://sodom.informatik.uni-kiel.de:8484/studierende/Login.html`

zur Modulabschlussprüfung des Moduls

- G1.1 - Programmierung

anzumelden! Die Abschlussklausur zu Informatik I findet am *23. Februar 2009* von 12:00 bis 14:00 Uhr im Hörsaal CAP2-G statt.

Hinweis: Zeichnungen dürfen Sie wahlweise auf Papier oder als Datei über die *Upload*-Funktion von iLearn abgeben.

Präsenzaufgabe 1

In der Vorlesung wurde die Implementierung einer Warteschlange durch veränderbare Datenstrukturen vorgestellt. Zeichnen Sie die Umgebungen, Bindungen und Prozeduren, die bei der Auswertung der folgenden Ausdrücke entstehen.

```
(define q (konstr-warteschlange))
(define p (konstr-warteschlange))
(hinzufuegen-warteschlange! q 1)
(hinzufuegen-warteschlange! p 2)
(entfernen-warteschlange! p)
```

Präsenzaufgabe 2

Der in der Vorlesung vorgestellte Halbaddierer basiert auf zwei UND-Gattern, einem ODER-Gatter und einem Inverter.

(a) Geben Sie die Verzögerungszeit dieses Halbaddierers abhängig von den Variablen und-gatter-verzoegerung, oder-gatter-verzoegerung und inverter-verzoegerung an.

(b) Definieren Sie ein XOR-Gatter für die Simulation digitaler Schaltkreise. Benutzen Sie das XOR-Gatter, um eine alternative Implementierung eines Halbaddierers anzugeben! Welche Verzögerungszeit besitzt der neue Halbaddierer?

Aufgabe 3 5+2 Punkte

In der Vorlesung wurden veränderbare Datenstrukturen verwendet, um eine Warteschlange zu implementieren. In dieser Aufgabe sollen Sie die Implementierung so erweitern, dass Elemente sowohl am Anfang als auch am Ende hinzugefügt und entfernt werden können. Den Quellcode aus der Vorlesung können Sie über die Übungswebsite herunterladen.

(a) Definieren Sie

- einen Selektor `(ende q)` der das letzte Element von `q` zurückgibt,
- einen Mutator `(vorne-hinzufuegen-warteschlange! q e)`, der ein Element `e` vorne an `q` anhängt, und
- einen Mutator `(hinten-entfernen-warteschlange! q)`, der das letzte Element aus `q` entfernt.

(b) Skizzieren Sie die Strukturen, die bei jedem Schritt der Ausführung von

```
(define q (konstr-warteschlange))
(vorne-hinzufuegen-warteschlange! q 42)
(entfernen-warteschlange! q)
(hinzufuegen-warteschlange! q 42)
```

entstehen.

Aufgabe 4 2+5 Punkte

In der Vorlesung wurde ein Simulator für digitale Schaltkreise vorgestellt und am Beispiel eines Halbaddierers demonstriert. Die entsprechenden Programmteile können von der Übungs-Website heruntergeladen werden und stehen in iLearn bereits zur Verfügung.

(a) Aus den in der Vorlesung vorgestellten Schaltelementen lässt sich ein 1-Bit-Volladdierer zusammenbauen, der aus *drei* Eingabebits ein Summen- und ein Übertragsbit berechnet. Zeichnen Sie das Schaltbild eines 1-Bit-Volladdierers.

(b) Definieren Sie aufbauend auf Ihrem Schaltbild einen 1-Bit-Volladdierer

```
(volladdierer a b cin s cout)
```

mit den Eingängen `a`, `b` und `cin` sowie den Ausgängen `s` für das Summen- und `cout` für das Übertragsbit.

Aufgabe 5 2+4 Punkte

In der Vorlesung wurden Beschränkungsnetze vorgestellt. Die entsprechenden Programmteile können von der Übungs-Website heruntergeladen werden und stehen in iLearn bereits zur Verfügung.

(a) Ein Quadrierer ist eine Beschränkung zwischen zwei Größen `a` und `b`, wobei der Wert von `b` das Quadrat des Wertes von `a` sein muss. Warum ist die folgende Definition eines Quadrierers ungeeignet?

```
(define (quadrierer a b) (multiplikator a a b))
```

(b) Definieren Sie einen korrekten Quadrierer und darauf aufbauend eine Beschränkung

```
(pythagoras a b c)
```

die festlegt, dass $a^2 + b^2 = c^2$ gilt.