Christian-Albrechts-Universität zu Kiel
Institut für Informatik
Lehrstuhl für Programmiersprachen und Übersetzerkonstruktion

Prof. Dr. Michael Hanus, Jan Tikovsky

Institut für Informatik

2. Übung zur Vorlesung „Programmierung“

Serie 2

Abgabe am Freitag, 07. November 2014 - 08:00

Von nun an sollen die Aufgaben über iLearn abgegeben werden:

https://ilearn.ps.informatik.uni-kiel.de/

Die Zugangsdaten zu iLearn erhalten Sie per E-Mail an Ihre stu-Adresse. Abrufen können Sie die E-Mails an Ihre stu-Adresse beispielsweise über das Webinterface unter folgender URL:

https://webmail.mail.uni-kiel.de/webmail/src/login.php

Vor der Abgabe der Aufgaben sollten Sie sich im iLearn mit Ihrem Übungsgruppenpartner zu einer Kleingruppe zusammenschließen!

Zum Testen Ihrer Abgaben gibt es im iLearn automatische Tests. **Um sicherzustellen, dass diese Testfälle auch funktionieren, halten Sie sich bei Ihrer Implementierung bitte an die in den Aufgabenstellungen vorgegebenen Prozedurnamen und Parameterreihenfolgen!** Dass alle Testfälle im iLearn erfolgreich sind, bedeutet noch nicht unbedingt, dass Ihre Abgabe fehlerfrei ist. Daher sollten Sie Ihre Abgaben in DrRacket ausführlich mit eigenen Testfällen testen!

Falls Sie Ihre Lösungen zusätzlich schriftlich in der Vorlesung abgeben, vergessen Sie bitte nicht Ihren Namen, Ihre Matrikelnummer und Ihre Übungsgruppe auf Ihre Abgabe zu schreiben.

Präsenzaufgabe 1 - Zylindervolumen

Definieren Sie eine Funktion, die das Volumen eines Zylinders aus dem Radius seiner Grundfläche und seiner Höhe berechnet. Gehen Sie dabei nach dem Konstruktionsprinzip aus der Vorlesung vor. Sie sollten also zunächst einen Kommentar schreiben, eine Signaturvereinbarung angeben und ausreichend Testfälle definieren.

Versuchen Sie Ihr Programm so zu strukturieren, dass die Prozeduren nicht zu komplex werden, sondern definieren Sie zunächst Prozeduren für Teilprobleme.

Präsenzaufgabe 2 - Anzahl der Pkw

In der Vorlesung haben Sie eine Funktion kennen gelernt, die aus der Anzahl von Autos und Motorrädern die Gesamtzahl der Reifen berechnet. Nun sollen Sie diese Funktion umkehren und eine Funktion definieren, die aus der Gesamtzahl der Reifen und der Anzahl der Fahrzeuge die Anzahl der Pkw berechnet.

Wie immer sollen auch hier zunächst ein geeigneter Kommentar, die Signaturvereinbarung und ausreichend Testfälle angegeben werden.

Aufgabe 3 - Celsius und Fahrenheit

4 Punkte

Sei C eine Temperatur in Grad Celsius. Die entsprechende Temperatur F in Grad Fahrenheit ergibt sich als:

$F = \frac{9}{5}C + 32$

1. Schreiben Sie eine Prozedur `celsius->fahrenheit`, die als Parameter eine Temperatur in °C akzeptiert und eine Temperatur in °F zurückgibt. Benutzen Sie dazu die Konstruktionsanleitung aus der Vorlesung:

Geben Sie einen Kommentar und die Signaturvereinbarung an, erstellen Sie dann die Testfälle und das Gerüst. Vervollständigen Sie danach den Rumpf der Prozedur und vergewissern Sie sich, dass die Tests erfolgreich laufen.

2. Schreiben Sie eine Prozedur `fahrenheit->celsius`, die als Parameter eine Temperatur in °F akzeptiert und eine Temperatur in °C zurückgibt. Benutzen Sie dazu ebenfalls die Konstruktionsanleitung aus der Vorlesung!

Aufgabe 4 - Benzinverbrauch

8 Punkte

In den USA und in Europa gibt es unterschiedliche Maße für die Energieeffizienz von Kraftfahrzeugen:

1. In Europa ist das gängige Maß der *Verbrauch* in Liter pro 100km (l/100km);
2. in den USA ist das gängige Maß die *Reichweite* in Meilen pro Gallone (mi/gal).

Schreiben Sie Prozeduren, die zwischen beiden Maßeinheiten umrechnen. Gehen Sie dazu wie folgt vor:

Halten Sie sich bei jeder Prozedur, die Sie schreiben, an die Konstruktionsanleitungen: Schreiben Sie zuerst einen Kommentar und die Signaturvereinbarung. Schreiben Sie als nächstes einige Testfälle. Leiten Sie danach das Gerüst von der Signaturvereinbarung her und vervollständigen Sie den Rumpf der Prozedur.

1. Schreiben Sie eine Prozedur `liters-per-hundred-kilometers`, die eine Menge Benzin in Liter und die Reichweite dieses Benzins in Kilometer akzeptiert und daraus den Verbrauch in Liter pro 100 km berechnet.
2. Schreiben Sie eine Prozedur `miles-per-gallon`, die eine Entfernung in Meilen und den Benzinverbrauch auf diese Entfernung in Gallonen akzeptiert und daraus die Reichweite in Meilen pro Gallone berechnet.
3. Definieren Sie eine Konstante `kilometers-per-mile` (eine US-Meile entspricht etwa $1,61$ Kilometer) und schreiben Sie zwei Prozeduren `kilometers->miles` und `miles->kilometers`, die jeweils eine Entfernung in einer Maßeinheit akzeptieren und die Entfernung in die jeweils andere Maßeinheit umrechnen.
4. Definieren Sie eine Konstante `liters-per-gallon` (eine Gallone entspricht etwa $3,79$ Liter) und schreiben Sie zwei Prozeduren `liters->gallons` und `gallons->liters`, die jeweils eine Menge in einer Maßeinheit akzeptieren und die Menge in die jeweils andere Maßeinheit umrechnen.
5. Schreiben Sie die Prozedur `l/100km->mi/gal`, die einen Verbrauch in Liter pro 100 km akzeptiert und in die Reichweite in Meilen pro Gallone umrechnet. Benutzen Sie dafür die Prozeduren, die Sie in den anderen Teilaufgaben erstellt haben. Sollten Sie auf weitere Teilprobleme stoßen, abstrahieren Sie diese Teilprobleme in eigene Prozeduren.
6. Schreiben Sie die Prozedur `mi/gal->l/100km`, die eine Reichweite in Meilen pro Gallone akzeptiert und in den Verbrauch in Liter pro 100 km umrechnet. Benutzen Sie dafür die Prozeduren, die Sie in den anderen Teilaufgaben erstellt haben. Sollten Sie auf weitere Teilprobleme stoßen, abstrahieren Sie diese Teilprobleme in eigene Prozeduren.

Aufgabe 5 - Compare

2 Punkte

Definieren Sie eine Prozedur `compare` zum Vergleich zweier reeller Zahlen. Ihre Prozedur soll die folgende Signaturvereinbarung haben:

```
(: compare (real real -> (one-of "LT" "EQ" "GT")))
```

Falls die erste Zahl kleiner als die zweite ist, soll `"LT"` (für less than) zurückgegeben werden. Ist die erste Zahl größer als die zweite, so wird `"GT"` (für greater than) zurückgegeben. Bei Gleichheit der beiden Zahlen soll das Ergebnis `"EQ"` (für equal) lauten.

Befolgen Sie bei der Implementierung die Konstruktionsanleitung aus der Vorlesung.

Aufgabe 6 - Jahreszeiten 2015 6 Punkte

1. Definieren Sie ein Prädikat `valid-date?`, das ein Datum in Form zweier natürlicher Zahlen (Tag und Monat) entgegennimmt und überprüft, ob es sich dabei um ein gültiges Kalenderdatum handelt. Schaltjahre brauchen Sie dabei nicht zu berücksichtigen, das heißt der Februar hat nur 28 Tage.

   Es folgen einige Beispielaufrufe:

   ```
   (valid-date? 29 2)
   #f
   (valid-date? 24 12)
   #t
   (valid-date? 31 4)
   #f
   (valid-date? 31 8)
   #t
   (valid-date? 25 13)
   #f
   ```

2. Definieren Sie eine Prozedur `season-2015`, die ein Datum in Form zweier natürlicher Zahlen (Tag und Monat) entgegennimmt und die zugehörige Jahreszeit für das Jahr 2015 zurückgibt. Verwenden Sie `one-of` zur Definition der Auswahl der Jahreszeiten (`"autumn"`, `"winter"`, `"spring"`, `"summer"` und `"invalid date"`) in der Signaturvereinbarung. Außerdem soll Ihre Prozedur das Prädikat `valid-date?` aus dem ersten Aufgabenteil nutzen, um zu überprüfen, ob es sich bei dem gegebenen Tag und Monat um ein gültiges Kalenderdatum handelt. Für ungültige Daten soll `"invalid date"` zurückgegeben werden.

   Da die astronomischen Jahreszeitenanfänge abhängig vom betrachteten Kalenderjahr variieren können, soll Ihre Prozedur die zugehörige Jahreszeit für ein Datum des Jahres 2015 bestimmen. Für das Jahr 2015 lauten die astronomischen Jahreszeitenanfänge wie folgt:

   - 20. März Frühlingsanfang
   - 21. Juni Sommeranfang
   - 23. September Herbstanfang
   - 22. Dezember Winteranfang

   Es folgen einige Beispielaufrufe:

   ```
   (season-2015 19 3)
   "winter"
   (season-2015 15 8)
   "summer"
   (season-2015 23 9)
   "autumn"
   (season-2015 29 2)
   "invalid date"
   ```

Denken Sie daran, für beide Prozeduren einen Kommentar, eine Signaturvereinbarung sowie Testfälle anzugeben. Ihre Tests sollten dabei alle möglichen Ausgaben abdecken.

Hinweis: Bitte geben Sie für den zweiten Aufgabenteil im iLearn nur Ihre Implementierung der Prozedur `season-2015` ab. Zur Ausführung der Testfälle wird die Musterlösung der Prozedur `valid-date?` automatisch dazu geladen. Somit können Sie die zweite Teilaufgabe auch lösen und im iLearn testen, falls Sie die erste Teilaufgabe nicht gelöst haben. Achten Sie aber darauf, die vorgegebenen Prozedurnamen zu verwenden!