Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Informatik

Programming Concepts and Paradigms (PCP)

Programmierübung zu Scheme 1+2

Hauptthemen:	Funktionale Programmierung: Grundlagen, Funktionale Modellierung, Bedingte Ausdrücke, Strukturen
Zeitfenster:	ca. 4-6 Lektionen

R. Diehl, FS2020

Diese Übung repetiert und vertieft den in Scheme 1 und 2 vermittelten Stoff. Gehen Sie zur Vorbereitung nochmals durch die Unterrichtsfolien durch und schauen Sie insbesondere, dass Sie alle auf den Folien angegebenen Code-Beispiele verstanden haben.

Die mit * gekennzeichneten Aufgaben müssen dem Dozenten oder Assistenten als Teil des Testats gezeigt werden.

Interaktionsübungen mit DrRacket

(Einstellung in DrRacket: "Beginning Student")

1. Aufgabe

Gliedern Sie folgende Terme im Editor und verfolgen Sie die Berechnung jedes Terms mit dem Stepper:

a) `(42 + (25 - 3 * 4)) : 11`
b) `(24 + 32) : 7 + 3 * (17 - 15)`
c) `34428 : 38 - (1103 - 197)`

2. Aufgabe

Schreiben Sie in mathematischer Notation und berechnen Sie folgende Formen

```
> (/ 1 (+ 1 1))
> (/ 1 (+ 1 (/ 1 (+ 1 1))))
> (/ 1 (+ 1 (/ 1 (+ 1 (/ 1 (+ 1 1))))))
```

Was fällt bei den Ergebnissen auf?

3. Aufgabe

Definieren Sie Länge und Breite eines Rechtecks und berechnen Sie dessen Diagonale.

4. Aufgabe

Gegeben ist der komplexe Term: -(10a + 20b - 40c) - (-(7a - 14c) - (-(15b - 4c) - (27a – 5b)))

a) Erstellen Sie eine Funktion, der man die Parameter a, b, c übergeben kann und diese dann den Term berechnet.
b) Wie lassen sich die Koeffizienten von a, b und c einfach zusammen zählen?

Bedingte Ausdrücke

(Einstellung in DrRacket: "Beginning Student")

5. Aufgabe *

Darf man bei folgenden Beispielen die (cond ...)-Klauseln vertauschen ohne dass sich die Semantik des Programms ändert?

a) Temperaturen

```
(cond
   ((> temperatur 35) "heiss")
   ((> temperatur 25) "warm")
   ((> temperatur 15) "mittel")
   (else "kalt"))
```

b) Teilbar

```
(cond
   ((zero? (remainder zahl 2)) "durch 2 teilbar")
   ((zero? (remainder zahl 3)) "durch 3 teilbar")
   (else "weder durch 2 noch durch 3 teilbar"))
```

6. Aufgabe *

Analysieren Sie die folgende Funktion zur Maut Berechnung

```
(define (toll total-weight)
   (cond
      ((not (number? total-weight)) "Eingabe muss Zahl sein!")
      ((<= total-weight 0) "Zahl muss größer 0 sein!")
      ((<= total-weight 1000) 20)
      ((<= total-weight 2000) 30)
      ((<= total-weight 5000) 50)
      ((<= total-weight 10000) 100)
      (else 250)))
```

a) Welchen Sinn haben die beiden ersten Klauseln?
b) Begründen Sie, weshalb man hier die Reihenfolge der Klauseln nicht ändern darf
c) Nennen Sie ein konkretes Beispiel, bei dem man die Klauseln ändern darf

7. Aufgabe *

Gegeben ist

```
(* (cond ((> a b) a)
   ((< a b) b)
   (else -1))
   (+ a 1))
```

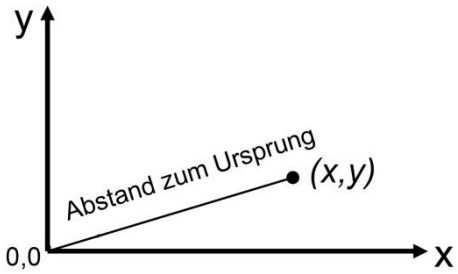
a) Handelt es sich um einen korrekten Scheme-Ausdruck? Begründung!
b) Definieren Sie a und b jeweils so, dass alle Klauseln einmal zutreffen. Welche Ergebnisse erhalten Sie?

Strukturen

(Einstellung in DrRacket: "Beginning Student")

8. Aufgabe

Erstellen Sie eine Funktion für die Abstandsberechnung vom Ursprung (0|0). Nehmen Sie dazu das Beispiel 2D Punkt (Folie in Scheme-2) als Vorlage.



9. Aufgabe *

Bei Skelettfunden schließt man aus der Länge der Knochen auf die Körpergröße; und zwar gilt (als statistischer Mittelwert) in cm:

- Körpergröße = 69.089 + 2.238 * Oberschenkelknochenlänge bei Männern
- Körpergröße = 61.412 + 2.317 * Oberschenkelknochenlänge bei Frauen

Ab dem 30. Lebensjahr nimmt die Körpergröße um 0,06 cm pro Jahr ab.

a) Definieren Sie einen Datentyp **`human`** mit den Feldern Alter, Geschlecht und Oberschenkelknochenlänge.
b) Erstellen Sie eine Funktion **`b-length`**, die aus einem Objekt vom Typ **`human`** die vermutete Körpergrösse berechnet.

Anhang: Vordefinierte Funktionen

Für <num> sind definiert

`+, -, *, /`	auch mit mehr als 2 Operanden
`=`	oder (equal? <num> <num>)

Nur für komplexe Zahlen sinnvoll sind

`angle, magnitude, imag-part, real-part, conjugate`

Für <real> sind definiert

`<, <=, >, >=`	auch mit mehr als 2 Operanden
`abs`	Betrag: |...|
`round`	rundet auf ganze Zahlen
`max, min`	auch mit mehr als 2 Operanden
`floor, ceiling`	untere/obere Gaussklammer
`make-polar`	Umkehrung zu angle und magnitude
`sign`	Vorzeichenfunktion (1, 0 -1)
`exp`	e-Funktion
`log`	ln (nat. Log.)
`expt`	allg. Exponentialfunktion
`sin, cos, tan, asin, acos, atan`	trigon. Funktionen inkl. Umkehrung
`sinh, cosh`	hyperbolische Funktionen

Für <rat> sind definiert

`numerator, denominator`	Zähler, Nenner eines gekürzten Bruches

Für <int> sind definiert

`quotient`	Ganzzahldivision
`remainder, modulo`	Rest bei Ganzzahldivision
`gcd, lcm`	ggT, kgV
`random`	Zufallszahlen 0 ... n-1

Umwandlungsfunktionen

`exact->inexact`
`inexact->exact`
`integer->char`

Konstanten

`e`	EULERsche Zahl e (#i2.718281828459045)
`pi`	π (#i3.141592653589793)