




Präsenzübungen zur Vorlesung
Deklarative Programmierung: Sommersemester 2018
Nr. 6

Aufgabe 6.1: Fernrohren

- a) Was ist lexikalisches Scoping? 
- b) Wie erreichen die Regeln APP und LOCAL lexikalisches Scoping? 
- c) Erläutern Sie den Begriff „shadowing“. 
- d) In wie weit trägt dies zur Modularisierung von Programmen bei?

Aufgabe 6.2: Interdimensionale SchindLuder +

Werten Sie die folgenden Ausdrücke schrittweise und unter Angabe der jeweils verwendeten Regel aus:




- a)

```
(define f (lambda (x)
  (local [(define y 3)]
    (* x y))))
(f 3)
```
- b)





```
(define f (lambda (x)
  (local [(define y 5)]
    (+ x (local [(define x 3)]
      (+ x y))))))
(f 7)
```

Aufgabe 6.3: Was macht der Freeman da?

Geben Sie eine Signatur sowie eine sinnvolle Aufgabenbeschreibung für die folgenden Funktionen an:

- a) `(define (f comp l)`
 `(foldr (lambda (x y) (if (comp x y) x y)) (first l)`
 `→ l))` 
- b) `(define (combine f g h)`
 `(lambda (x) (f (g x) (h x))))` 
- c) `(define (add f g)`
 `(lambda (x y) (+ (f x) (g y))))` 

Aufgabe 6.4: Aperture

- a) Schreiben Sie eine Funktion `(add-to-list n l)`, die die übergebene Zahl auf jedes Element der Liste addiert (gehen Sie davon aus, dass die Liste lediglich Zahlen enthält). 
- b) Geben Sie eine Signatur sowie eine sinnvolle Aufgabenbeschreibung für die obige Funktion an. 
- c) Modifizieren Sie die obige Funktion dahingehend, dass sie eine lokale Definition für das Addieren und die Transformation mittels `map` bewerkstelligt. 
- d) Ersetzen Sie Ihre lokale Funktionsdefinition durch einen Lambda-Ausdruck. 
- e) Was versteht man unter dem Begriff **closure**? 