Aufgaben – Lösungen

Einführung in die Programmierung

Johannes Brauer

24. Januar 2020

Inhaltsverzeichnis

1 Aufgabe (Erste Schritte in Racket) 1

2 Aufgabe (Auswertung arithmetischer Ausdrücke) 2

3 Aufgabe (Aufschreiben elementarer Funktionen) 2

4 Aufgabe (Aufschreiben elementarer Funktionen) 3

5 Aufgabe (Anwenden der Aufschreibregeln) 4

6 Aufgabe (Profit für den Kinobesitzer) 4

7 Aufgabe (Modifikation von `kino.rkt`) 5

8 Aufgabe (Einsatz von Hilfsfunktionen) 5

9 Aufgabe (Ersetzungsmodell) 7

1 Aufgabe (Erste Schritte in Racket)

Machen Sie sich mit den NORDAKADEMIE-Rechnern vertraut und richten Sie Ihren Arbeitsplatz ein (Mail, Webbrowser, Verzeichnisse für die Vorlesungen usw.).

Finden und starten Sie DrRacket nach den Anweisungen in der Vorlesung. Werten Sie einen ersten Ausdruck aus, z. B. (* 6 7).

Welche Funktion haben die Buttons? Welche Menü-Befehle verstehen Sie schon?

Schauen Sie sich in einem Webbrowser die Seiten zu Racket unter `https://racket-lang.org/` an. Wo finden Sie Hilfe zur Bedienung von DrRacket? Wie können Sie sich über die Sprache Racket informieren? Wo finden Sie alle vordefinierten mathematischen Funktionen?

2 Aufgabe (Auswertung arithmetischer Ausdrücke)

1. Wie wird der Ausdruck
   `(* (+ 2 2) (/ (* (+ 3 5) (/ 30 10)) 2))`
   ausgewertet?

2. Experimentieren Sie mit verschiedenen Operatoren und Zahlenarten.

3. Werten Sie die folgenden Ausdrücke aus und vergleichen Sie die Resultate:

   ```
   (- 1.0 0.9)
   (- 1000.0 999.9)
   (- #i1000.0 #i999.9)
   ```

3 Aufgabe (Aufschreiben elementarer Funktionen)

Schreiben Sie für die folgenden mathematischen Formeln Racket-Funktionsdefinitionen auf:

1. $n^2 + 1$

   ```
   (define fa
     (lambda [n]
       (+ (* n n) 1)))
   ```

2. $\frac{1}{2}n^2 + 3$

   ```
   (define fb
     (lambda [n]
       (+ (/ (* n n) 2) 3)))
   ```

3. $2 - \frac{1}{n}$

   ```
   (define fc
     (lambda [n]
       (- 2 (/ 1 n))))
   ```

Geben Sie die Racket-Funktionen in das Definitionsfenster von DrRacket ein. Geben Sie anschließend in das Interaktionsfenster Funktionsaufrufe für diese Funktionen ein.

4 Aufgabe (Aufschreiben elementarer Funktionen)

In der Praxis findet der Programmierer selten mathematische Formeln vor. Aufgabenstellungen sind eher als Prosatext gegeben. Die Berechnungsformeln muss er selbst entwickeln durch

- eigenes Nachdenken,
- Nachschlagen in geeigneten Quellen oder
- Nachfragen beim Auftraggeber.

Finden Sie für die folgenden Aufgabenstellungen die passenden Formeln und schreiben Sie diese als Funktionsdefinitionen in *Racket* auf:

1. Berechnung des Rauminhalts eines Quaders aus dessen Länge, Breite und Höhe.

```
(define quader-volumen
  (lambda [laenge breite hoehe]
    (* laenge breite hoehe)))
```

2. Schreiben Sie eine Funktion, die aus der Entfernung und der Geschwindigkeit zweier Züge die Zeit ermittelt, nach der die Züge sich treffen, wenn Sie sich auf einem gemeinsamen Streckenabschnitt von ihren jeweiligen Startpunkten aus aufeinander zu bewegen.

```
(define zug-treffen
  (lambda [entfernung geschwndgkt1 geschwndgkt2]
    (/ entfernung (+ geschwndgkt1 geschwndgkt2))))
```

3. Berechnung der Miete, die eine andere Spielerin in Monopoly bezahlen muss, falls sie auf einen Bahnhof trifft, der einer anderen Spielerin gehört. Die Miete ist davon abhängig wie viele Bahnhöfe der anderen Spielerin gehören:

Anzahl der Bahnhöfe	Miete
1	500
2	1000
3	2000
4	4000

Hinweis: Ein Aufruf `(expt x y)` liefert x^y als Ergebnis.

```
(define bahnhofsmiete
  (lambda [anzahl-bahnhoefe]
    (* 500 (expt 2 (- anzahl-bahnhoefe 1)))))
```

5 Aufgabe (Anwenden der Aufschreibregeln)

Schreiben Sie die Funktion zur Berechnung der Bahnhofsmiete in Monopoly (s. o.) gemäß den Regeln 1 bis 3 aus der Vorlesung auf.

```
;; berechnet die Bahnhofsmiete aus der Anzahl der Bahnhöfe
(define bahnhofsmiete
  (lambda [anzahl-bahnhoefe]
    (* 500 (expt 2 (- anzahl-bahnhoefe 1)))))
;; Beispielanwendungen
(= (bahnhofsmiete 4) 4000)
(= (bahnhofsmiete 3) 2000)
(= (bahnhofsmiete 2) 1000)
(= (bahnhofsmiete 1)  500)
```

Wenn nichts anderes angeben ist, sind auch die Funktionen für die folgenden Aufgaben gemäß diesen Regeln aufzuschreiben!!!

6 Aufgabe (Profit für den Kinobesitzer)

Ein altmodisches Vorstadtkino besitzt eine einfache Formel für die Berechnung des Profits einer Vorstellung: Jeder Kinobesucher bezahlt 500 Währungseinheiten für die Eintrittskarte. Jede Vorstellung kostet das Kino 2000 Währungseinheiten plus 50 Währungseinheiten pro Besucher. Schreiben Sie eine Funktion `profit` zur Berechnung des Profits bei gegebener Besucherzahl.

```
;; berechnet den Profit eine Filmvorfuehrung aus 2000 WE Fixkosten
;; und 50 WE Kosten pro Besucher bei gegebener Besucherzahl und
;; 500 WE Kartenpreis
(define profit
  (lambda [besucherzahl]
    (- (* 500 besucherzahl)
       (+ (* 50 besucherzahl) 2000))))
;; Beispielanwendungen
(= (profit 100) 43000)
```

7 Aufgabe (Modifikation von `kino.rkt`)

1. Modifizieren Sie das Programm `kino.rkt` so, dass die Fixkosten einer Veranstaltung wegfallen und dafür 15 Währungseinheiten pro Besucher an Kosten anfallen.

2. Nehmen Sie die gleiche Modifikation auch an der Funktion `profit` aus der Vorlesung vor, die ohne Hilfsfunktionen auskommt, und vergleichen Sie die Ergebnisse.

8 Aufgabe (Einsatz von Hilfsfunktionen)

Die folgenden Aufgaben sind unter Benutzung von Hilfsfunktionen zu lösen. Befolgen Sie unbedingt alle in der Vorlesung angegebenen Regeln:

1. Schreiben Sie ein Programm, das das Volumen eines Zylinders zu berechnen erlaubt. Eingangsgrößen sind der Radius und die Höhe des Zylinders.

2. Schreiben Sie ein Programm, das die Oberfläche eines Zylinders zu berechnen erlaubt. Eingangsgrößen sind der Radius und die Höhe des Zylinders.

3. Schreiben Sie ein Programm, das die Oberfläche eines Rohrs zu berechnen erlaubt. Eingangsgrößen sind der Innenradius, die Wandstärke und die Länge des Rohrs.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;; Lösung von Simon Greßmann, I18b ;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;Aufgabe 8a
(define mein-pi 3.14)
;Berechnet Kreisfläche aus Radius
(define Kreisflaeche
  (lambda [radius]
    (* radius radius mein-pi)))
(= (Kreisflaeche 1) 3.14)
(= (Kreisflaeche 2) 12.56)
;Berechnet das Volumen eines Zylinders in Abhängigkeit von Radius und Höhe
(define Zylindervolumen
```

```
  (lambda [radius hoehe]
    (* (Kreisflaeche radius) hoehe)))
(= (Zylindervolumen 1 1) 3.14)

;Aufgabe 8b
;Berechnet den Umfang eines Kreises in Anhängigkeit von Pi.
(define Kreisumfang
  (lambda [radius]
    (* 2 mein-pi radius)))
(= (Kreisumfang 1) 6.28)

;Berechnet die Mantelfläche eines Zylinders in Abhängigkeit von Radius und Höhe
(define Mantelflaeche
  (lambda (radius hoehe)
    (* (Kreisumfang radius) hoehe)))
(= (Mantelflaeche 1 1) 6.28)
(= (Mantelflaeche 2 1) 12.56)

;Berechnet die Oberfläche eines Zylinders in Abhängigkeit von Radius und Höhe
(define Zylinderoberflaeche
  (lambda [radius hoehe]
    (+ (* 2 (Kreisflaeche radius)) (Mantelflaeche radius hoehe))))
(= (Zylinderoberflaeche 1 1) 12.56)

;Aufgabe 8c
;Berechnet die Fläche eines Kreisrings in Abhängigkeit von innenradius und Breite
(define Kreisringflaeche
  (lambda [innenradius breite]
    (- (Kreisflaeche (+ innenradius breite)) (Kreisflaeche innenradius))))
(= (Kreisringflaeche 1 1) 9.42)
;Berechnet die Oberfläche eines Rohrs in Abhängigkeit von dessen Inneradius, Wandstärk
(define Rohroberflaeche
  (lambda [innenradius wandstaerke laenge]
    (+ (Mantelflaeche (+ innenradius wandstaerke) laenge)
       (Mantelflaeche innenradius laenge)
       (* 2 (Kreisringflaeche innenradius <dstaerke)))))
(= (Rohroberflaeche 1 1 1) 37.68)
```

9 Aufgabe (Ersetzungsmodell)

Gegeben sei die folgende Funktionsdefinition:

```
(define f
   (lambda [x y]
       (+ (* 3 x) (* y y))))
```

Werten Sie die folgenden Ausdrücke Schritt für Schritt unter Anwendung des Ersetzungsmodells aus:

1. `(f 1 (* 2 3))`

```
(f 1 (* 2 3))
= ((lambda [x y] (+ (* 3 x) (* y y))) 1 (* 2 3))
= ((lambda [x y] (+ (* 3 x) (* y y))) 1 6)
= (+ (* 3 1) (* 6 6))
= (+ 3 36)
= 39
```

2. `(+ (f 1 2) (f 2 1))`

```
(+ (f 1 2) (f 2 1))
= (+ ((lambda [x y] (+ (* 3 x) (* y y))) 1 2)
     ((lambda [x y] (+ (* 3 x) (* y y))) 2 1))
= (+ (+ (* 3 1) (* 2 2))
     (+ (* 3 2) (* 1 1)))
= (+ (+ 3 4) (+ 6 1))
= (+ 7 7)
= 14
```

Ausführlich:

```
(+ (f 1 2) (f 2 1))
= (+ ((lambda [x y] (+ (* 3 x) (* y y))) 1 2)
     (f 2 1))
= (+ ((lambda [x y] (+ (* 3 x) (* y y))) 1 2)
     ((lambda [x y] (+ (* 3 x) (* y y))) 2 1))
= (+ (+ (* 3 1) (* 2 2))
     (+ ((lambda [x y] (+ (* 3 x) (* y y))) 2 1))
= (+ (+ (* 3 1) (* 2 2))
     (+ (* 3 2) (* 1 1)))
= (+ (* 3 4) (+ (* 3 2) (* 1 1)))
```

```
= (+ (+ 3 4) (+ 6 1))
= (+ 7 (+ 6 1))
= (+ 7 7)
= 14
```

3. `(f (f 1 (* 2 3)) 19)`

```
(f (f 1 (* 2 3)) 19)
= ((lambda [x y] (+ (* 3 x) (* y y))) (f 1 (* 2 3)) 19)
= ((lambda [x y] (+ (* 3 x) (* y y)))
     ((lambda [x y] (+ (* 3 x) (* y y))) 1 (* 2 3)) 19)
= ((lambda [x y] (+ (* 3 x) (* y y)))
     ((lambda [x y] (+ (* 3 x) (* y y))) 1 6) 19)
= ((lambda [x y] (+ (* 3 x) (* y y)))
     (+ (* 3 1) (* 6 6)) 19)
= ((lambda [x y] (+ (* 3 x) (* y y))) (+ 3 36) 19)
= ((lambda [x y] (+ (* 3 x) (* y y))) 39 19)
= (+ (* 3 39) (* 19 19))
= (+ 117 361)
= 478
```