

# Contents

Exercise 1.1	1
Exercise 1.2	2
Exercise 1.3	2
Exercise 1.4	2

## Exercise 1.1

In below expressions, the result printed by the interpreter is given. It's assumed that the sequence is to be evaluated in the order they are presented.

```
10
10
(+ 5 3 4)
12
(- 9 1)
8
(/ 6 2)
3
(+ (* 2 4) (- 4 6))
6
(define a 3)
implementation dependent a = 3
(define b (+ a 1))
implementation dependent b = 4
(+ a b (* a b))
19
(= a b)
false
(if (and (> b a) (< b (* a b)))
    b
    a)
4
(cond ((= a 4) 6)
      ((= b 4) (+ 6 7 a))
      (else 25))
16
(+ 2 (if (> b a) b a))
6
(* (cond ((> a b) a)
        ((< a b) b)
        (else -1))
   (+ a 1))
16
```

## Exercise 1.2

Translation of the following expression into prefix notation

$$\frac{5 + 4 + (2 - (3 - (6 + \frac{4}{5})))}{3(6 - 2)(2 - 7)}$$

```
(/ (+ 5
     4
    (- 2 (- 3
            (+ 6
                (/ 4 5))))))
(* 3
   (- 6 2)
   (- 2 7)))
```

## Exercise 1.3

Definition of a procedure that takes three numbers as arguments and returns the sum of the squares of the two larger numbers.

```
(define (sum-squares-two-larger-numbers a b c)
  (cond ((and (< b a)
              (< b c)) (+ (* a a)
                          (* c c)))
        ((and (< a b)
              (< a c)) (+ (* b b)
                          (* c c)))
        (else (+ (* a a)
                  (* b b)))))
```

## Exercise 1.4

Description of the behavior of the following procedure with the observation that our model of evaluation allows for combinations whose operators are compound expressions.

```
(define (a-plus-abs-b a b)
  ((if (> b 0) + -) a b))
```

The sub-expression `(if (> b 0) + -)` will evaluate to `+` if  $b > 0$  and `-` otherwise. So, the body of the procedure will become

- `(+ a b)` if  $b > 0$
- `(- a b)` if  $b \leq 0$

In sum, the procedure `a-plus-abs-b` is computing  $a + |b|$ .