Problem Set 1 Solutions
COMP301 Fall 2019
03.10.2019 17:30 - 18:45

Problem 1[1]:

(1) $\{3n+2 | n \in \mathbb{N}\}$
   (a) Top-down
   $n \in S$ if $n = 2$ or
   $n - 3 \in S$
   (b) Bottom-up
   S is the smallest set satisfying the two properties:
   $2 \in S$ and
   if $n \in S$ then $n + 3 \in S$
   (c) Rules of Inference
   $$\frac{}{2 \in S} \quad \frac{n \in S}{n+3 \in S}$$

(2) $\{2n+3m+1 | n, m \in \mathbb{N}\}$
   (a) Top-down
   $n \in S$ if $n = 1$ or
   $n - 2 \in S$ or
   $n - 3 \in S$.
   (b) Bottom-up
   S is the smallest set satisfying the two properties:
   $1 \in S$ and
   if $n \in S$ then $n + 2 \in S$ or $n + 3 \in S$
   (c) Rules of Inference
   $$\frac{}{1 \in S} \quad \frac{n \in S}{n+2 \in S} \quad \frac{n \in S}{n+3 \in S}$$

(3) $\{(n, 2n+1) | n \in \mathbb{N}\}$
   (a) Top-down
   $(m, n) \in S$ if $m = 0$ and $n = 1$ or
   $(m - 1, n - 2) \in S$
   (b) Bottom-up
   S is the smallest set satisfying the two properties:
   $(0, 1) \in S$ and
   if $(m, n) \in S$ then $(m + 1, n + 2) \in S$
   (c) Rules of Inference
   $$\frac{}{(0,1) \in S} \quad \frac{(m,n) \in S}{(m+1, n+2) \in S}$$

(4) $\{(n, n^2) | n \in \mathbb{N}\}$

[1] EOPL p.16 Exercise 1.1

(a) Top-down
$(m, n) \in S$ if $m = 0$ and $n = 0$ or
$(m - 1, n - 2m + 1) \in S$

(b) Bottom-up
S is the smallest set satisfying the two properties: $(0, 0) \in S$ and
if $(m, n) \in S$ then $(m + 1, n + 2m + 1) \in S$

(c) Rules of Inference

$$\frac{}{(0,0) \in S} \qquad \frac{(m,n) \in S}{(m+1, n+2m+1) \in S}$$

Problem 2[2]:

```
List-of-Int
→(Int . List-of-Int)
→(Int . (Int . List-of-Int))
→(Int . (Int . (Int . List-of-Int)))
→(-7 . (Int . (Int . List-of-Int)))
→(-7 . (3 . (Int . List-of-Int)))
→(-7 . (3 . (14 . List-of-Int)))
→(-7 . (3 . (14 . ())))
```

Problem 3[3]: If we dont check `(null? lst)` and we do `(car lst)`, it is possible that `car` might be applied to an empty list, which causes an error.

[2]EOPL p.16 Exercise 1.4
[3]EOPL p.16 Exercise 1.6

Problem 4[4]: Refer to *ps1solutions.scm*.

```
;Function:Subst symbol x symbol x s-list -> s-list
(define subst
  (lambda (new old slist)
    (if (null? slist)
        '()
        (cons
         ((lambda (new old sexp)
            (if (symbol? sexp)
                (if (eqv? sexp old) new sexp)
                (subst new old sexp))) new old (car slist))
         (subst new old (cdr slist))))))
```

[4]EOPL p.22 Exercise 1.12

Problem 5[5]: Refer to *ps1solutions.scm*.

```
;Function:Product list x list -> list
(define product
  (lambda (sos1 sos2)
    (prod-help sos1 sos2 sos2)))

;Function:Prod-help list x list x list -> list
(define prod-help
  (lambda (sos1 sos2 temp)
    (cond
      ((null? sos1) '())
      ((null? temp) (prod-help (cdr sos1) sos2 sos2))
      (else (cons (list (car sos1) (car temp))
                  (prod-help sos1 sos2 (cdr temp)))))))
```

[5]EOPL p.27 Exercise 1.21

Problem 6[6]: Refer to *ps1solutions.scm*.

```
;Function:Up list -> list
(define up
  (lambda (lst)
    (cond
      ((null? lst) '())
      ((pair? (car lst)) (append (car lst) (up (cdr lst))))
      (else (cons (car lst) (up (cdr lst)))))))

;Function:Down list -> list
(define down
  (lambda (lst)
    (cond
      ((null? lst) '())
      (else (cons (list (car lst)) (down (cdr lst)))))))
```

[6]EOPL p.28 Exercise 1.26

Problem 7[7]: Refer to *ps1solutions.scm*.

```
;Function:Path Sybmol x binary-tree(list) -> list
(define path
  (lambda (n bin-tree)
    (path-help n bin-tree '())))

;Function:Path Sybmol x binary-tree(list) x list -> list
(define path-help
  (lambda (n bin-tree waylst)
    (cond
      ((null? bin-tree) '())
      ((eqv? n (car bin-tree)) waylst)
      (else (append (path-help n (cadr bin-tree) (append waylst '(left)))
                    (path-help n (caddr bin-tree) (append waylst '(right))))))))
```

[7]EOPL p.30 Exercise 1.34

Problem 8[8]: Refer to *ps1solutions.scm*.

```
;Function:G list x list -> list
(define g
  (lambda (lst1 lst2)
       (cons lst1 (map (lambda (x) (list (+ 1 (car x)) (cadr x))) lst2))))

;Function:Number-elements list -> list
(define number-elements
  (lambda (lst)
    (if (null? lst) '()
       (g (list 0 (car lst)) (number-elements (cdr lst))))))
```

[8]EOPL p.30 Exercise 1.36