Лабораторная работа № 2. Рекурсия, процедуры высшего порядка, обработка списков

30 ноября 2023 г.

Бойко Роман, ИУ9-12Б

Цель работы

Приобретение навыков работы с основами программирования на языке Scheme: использование рекурсии, процедур высшего порядка, списков.

Реализация

```
;; Number 1
(define (count x xs)
  (let loop ((i 0)
             (xs xs))
    (cond
      ((null? xs) i)
      ((equal? x (car xs)) (loop (+ i 1) (cdr xs)))
      (else (loop i (cdr xs))))))


;; Number 2
(define (delete pred? xs)
  (let loop ((new_xs '())
             (xs xs))
    (cond
      ((null? xs) new_xs)
      ((pred? (car xs)) (loop new_xs (cdr xs)))
      (else (loop (append new_xs (cons (car xs) '())) (cdr xs))))))


;; Number 3
(define (iterate f x n)
  (if (= n 0)
      '()
```

```scheme
      (cons x (iterate f (f x) (- n 1)))))


;; Number 4
(define (intersperse e xs)
  (let loop ((new_xs '())
             (xs xs)
             (flag #f))
    (cond
      ((null? xs) new_xs)
      (flag (loop (append new_xs (cons e '())) xs #f))
      (else (loop (append new_xs (cons (car xs) '())) (cdr xs) #t)))))


;; Number 5
(define (any? pred? xs)
  (and (not (null? xs)) (or (pred? (car xs)) (any? pred? (cdr xs)))))


(define (all? pred? xs)
      (or (null? xs) (and (pred? (car xs)) (all? pred? (cdr xs)))))


;; Number 6
(define (o . xs)
  (define (compose funcs)
    (if (null? funcs)
        (lambda (x) x)
        (lambda (x) ((car funcs) ((compose (cdr funcs)) x)))))
  (compose xs))
```

Тестирование

```scheme
(count 'a '(a b c a))
(count 'b '(a c d))
(count 'a '())

(delete even? '(0 1 2 3))
(delete even? '(0 2 4 6))
(delete even? '(1 3 5 7))
(delete even? '())

(iterate (lambda (x) (* 2 x)) 1 6)
(iterate (lambda (x) (* 2 x)) 1 1)
(iterate (lambda (x) (* 2 x)) 1 0)
```

```
(intersperse 'x '(1 2 3 4))
(intersperse 'x '(1 2))
(intersperse 'x '(1))
(intersperse 'x '())

(any? odd? '(1 3 5 7))
(any? odd? '(0 1 2 3))
(any? odd? '(0 2 4 6))
(any? odd? '())

(all? odd? '(1 3 5 7))
(all? odd? '(0 1 2 3))
(all? odd? '(0 2 4 6))
(all? odd? '())

(define (f x) (+ x 2))
(define (g x) (* x 3))
(define (h x) (- x))

((o f g h) 1)
((o f g) 1)
((o h) 1)
((o) 1)
```

Вывод

Приобрел навыки работы с рекурсией, процедурами высшего порядка, списков