

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

25 ноября 2023 г.

Семён Чайкин, ИУ9-11Б

Цель работы

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

Реализация

```
(define (count x xs)
  (if (null? xs)
    0
    (+ (if (equal? (car xs) x) 1 0) (count x (cdr xs)))))

(define (delete pred? xs)
  (if (null? xs)
    '()
    (if (pred? (car xs))
      (delete pred? (cdr xs))
      (cons (car xs) (delete pred? (cdr xs))))))

(define (iterate f x n)
  (if (= n 0)
    '()
    (cons x (iterate f (f x) (- n 1)))))

(define (intersperse x xs)
  (if (null? xs)
    '()
    (if (null? (cdr xs))
      (cons (car xs) (intersperse x (cdr xs)))
      (cons (car xs) (intersperse x (cdr xs))))))
```

```

  (cons (car xs) (cons x (intersperse x (cdr xs)))))

(define (any? pred? xs)
  (if (null? xs)
    #f
    (or (pred? (car xs)) (any? pred? (cdr xs)))))

(define (all? pred? xs)
  (if (null? xs)
    #t
    (and (pred? (car xs)) (all? pred? (cdr xs)))))

(define (o . fs)
  (define (o-rec fs x)
    (if (null? fs)
      x
      ((car fs) (o-rec (cdr fs) x))))
  (lambda (x) (o-rec fs x)))

```

Тестирование

```

Welcome to DrRacket, version 8.10 [cs].
Language: R5RS; memory limit: 128 MB.
> (count 1 '(1 2 3 4 5 6 7))
1
> (count 5 '(5 2 5 4 5 6 5))
4
> (count 0 '(1 2 3 4 5 6 7))
0
> (count 12 '())
0
> (delete even? '(1 2 3 4 5 6 7 8))
(1 3 5 7)
> (delete even? '(1 2 3 5 7))
(1 3 5 7)
> (delete even? '(7 5 3 1 9 7 3))
(7 5 3 1 9 7 3)
> (delete even? '(2 4 6 8))
()
> (delete even? '())
()
> (iterate (lambda (x) (* 2 x)) 1 6)
(1 2 4 8 16 32)
> (iterate (lambda (x) (* 3 x)) 5 1)
(5)

```

```

> (iterate (lambda (x) (+ 3 x)) 1 0)
()
> (intersperse 'x '(1 2 3 4 5 6))
(1 x 2 x 3 x 4 x 5 x 6)
> (intersperse 'x '(1 2))
(1 x 2)
> (intersperse 'x '(1))
(1)
> (intersperse 'x '())
()
> (any? even? '())
#f
> (any? even? '(1 2 3 4 5 7))
#t
> (any? even? '(1 3 5 7 9))
#f
> (all? even? '())
#t
> (all? even? '(1 2 3 4 5 7))
#f
> (all? even? '(1 3 5 7 9))
#f
> (all? even? '(2 4 6 8))
#t
> (define (f x) (+ x 2))
> (define (g x) (* x 3))
> (define (h x) (- x))
> ((o f g h) 1)
-1
> ((o f g) 1)
5
> ((o h) 1)
-1
> ((o) 1)
1

```

Вывод

Научился использовать рекурсию, процедуры высшего порядка и списки в языке программирования Scheme.