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

21 ноября 2023 г.

Лимонов Дмитрий, ИУ9-12Б

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? x (car xs))
          (+ 1 (count x (cdr xs)))
          (count x (cdr xs)))))

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

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

(define (intersperse e xs)
  (if (<= (length xs) 1)
      xs
```

```
      (append (list (car xs) e) (intersperse e (cdr xs)))))

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

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

(define (o . xs)
  (define (f x xs)
    (if (null? xs)
        x
        ((car xs) (f x (cdr xs)))))
  (lambda x (f (car x) xs)))
```

Тестирование

```
Welcome to DrRacket, version 8.11 [cs].
Language: R5RS; memory limit: 128 MB.
> (iterate (lambda (x) (* 2 x)) 1 6)
(1 2 4 8 16 32)
> (iterate (lambda (x) (* 2 x)) 1 0)
()
> (count 'a '(a b c a))
2
> (count 'b '(a c d))
0
> (count 'a '())
0
> (delete even? '(0 1 2 3))
(1 3)
> (delete even? '(0 2 4 6))
()
> (delete even? '(1 3 5 7))
(1 3 5 7)
> (delete even? '())
()
> (iterate (lambda (x) (* 2 x)) 1 6)
(1 2 4 8 16 32)
> (iterate (lambda (x) (* 2 x)) 1 1)
(1)
> (iterate (lambda (x) (* 2 x)) 1 0)
()
> (intersperse 'x '(1 2 3 4))
(1 x 2 x 3 x 4)
```

```
> (intersperse 'x '(1 2))
(1 x 2)
> (intersperse 'x '(1))
(1)
> (intersperse 'x '())
()
> (any? odd? '(1 3 5 7))
#t
> (any? odd? '(0 1 2 3))
#t
> (any? odd? '(0 2 4 6))
#f
> (any? odd? '())
#f
> (all? odd? '(1 3 5 7))
#t
> (all? odd? '(0 1 2 3))
#f
> (all? odd? '(0 2 4 6))
#f
> (all? odd? '())
#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

Удалось углубить свои знания в языке Scheme, а также применить основные операции со списками, а также рекурсией.