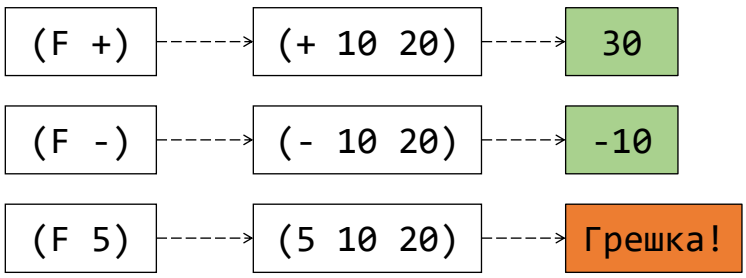
Функции от по-висок ред част 2: процедурите като параметри

доц. Атанас Семерджиев

1

Процедурите като параметри

```
(define (F op)
   (op 10 20)
)
```



2

2

Производна в точка

```
(define (derive x f dx)
  (/ (- (f (+ x dx))
        (f x))
     dx
  )
)
```

$$\frac{f(x+dx)-f(x)}{dx}$$

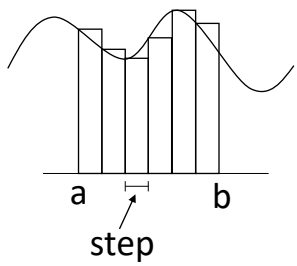
3

3

Интегриране (наивно решение)

```
(define (integrate f a b step)
  (if (> a b) 0
      (+ (* (f a)
            step)
         (integrate f
                    (+ a step)
                    b
                    step)
)))
```

$$\int_a^b f(x)dx$$



4

4

(accumulate)

5

```
for(i = a; i < b; i = next(i))
{}
```

```
(define (simple a next b)
  (define (loop i)
    (if (< i b)
        (loop (next i)) )
  )
  (loop a)
)
```

6

6

Специална форма (begin)

Общ вид:

(begin $expr_1$ $expr_2$ … $expr_n$)

Последователно се оценяват изразите $expr_1$, $expr_2$, … $expr_n$

Оценката на $expr_n$ става оценка на begin израза

```
(begin (display "abc")
       (display "def")
       (display "\n")
       5)
```

→ 5

7

7

```
for(i = a; i < b; i = next(i))
   op(i);
```

```
(define (simple-op op a next b)
  (define (loop i)
    (if (< i b)
        (begin (op i)
               (loop (next i)) ))
  )
  (loop a)
)
```

Няма смисъл без страничен ефект!

8

8

```
(define (1+ n) (+ 1 n))

(simple-op display 1 1+ 10)
```

```
Welcome to DrRacket, version 7.8 [3m].
Language: R5RS; memory limit: 128 MB.
123456789
```

9

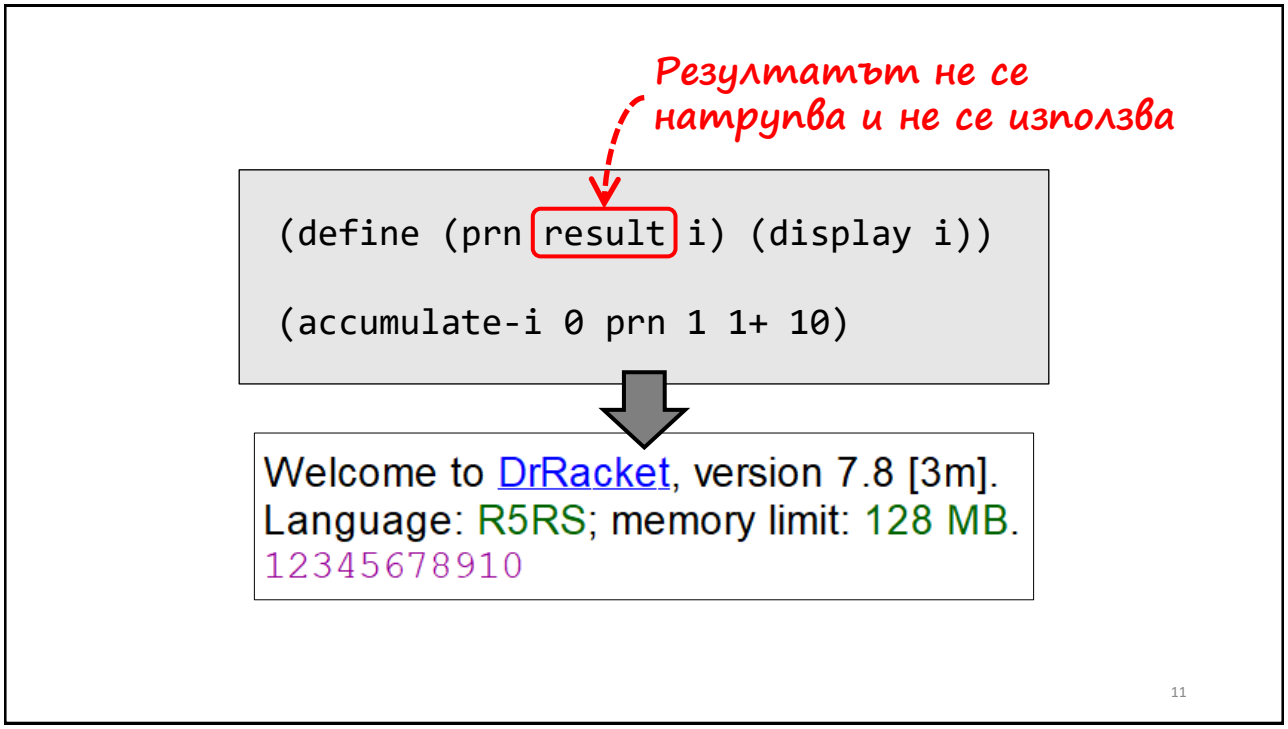
9

```
result = init;
for(i = a; i < b; i = next(i))
   result = op(result, i);
```

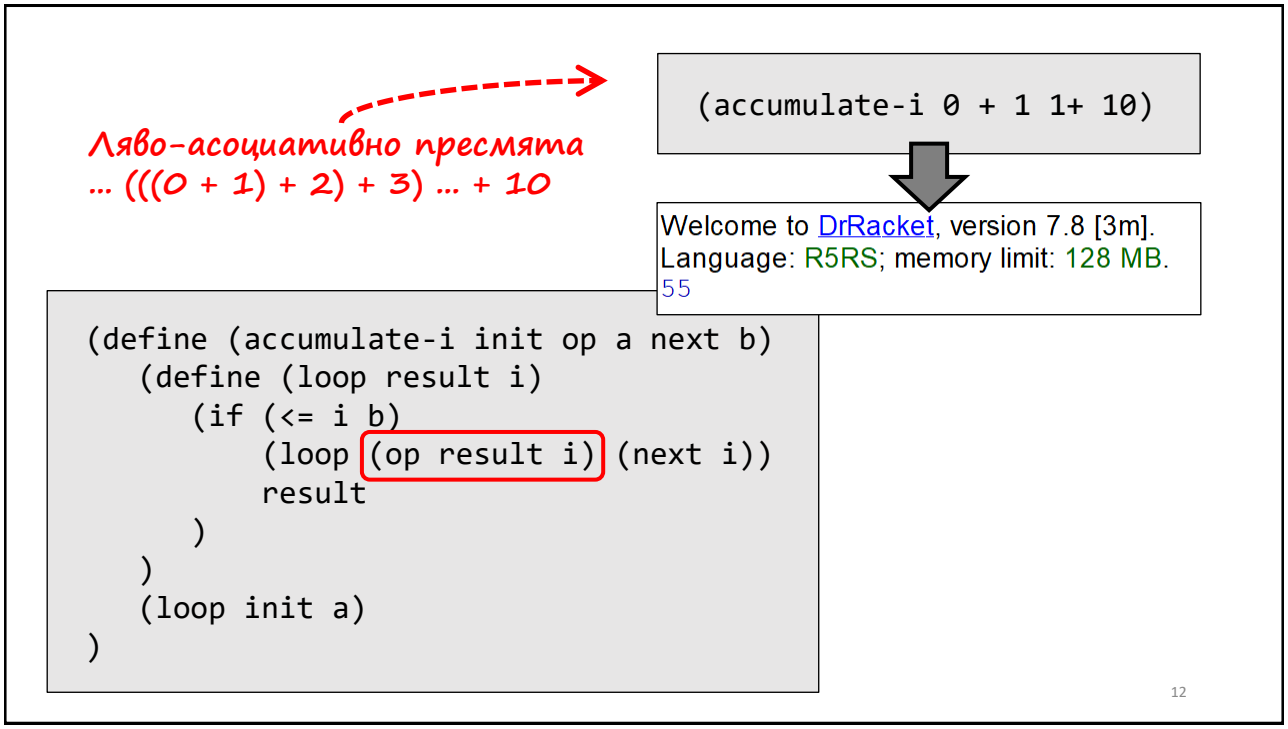
```
(define (accumulate-i init op a next b)
   (define (loop result i)
      (if (<= i b)
          (loop (op result i) (next i))
          result
      )
   )
   (loop init a)
)
```

10

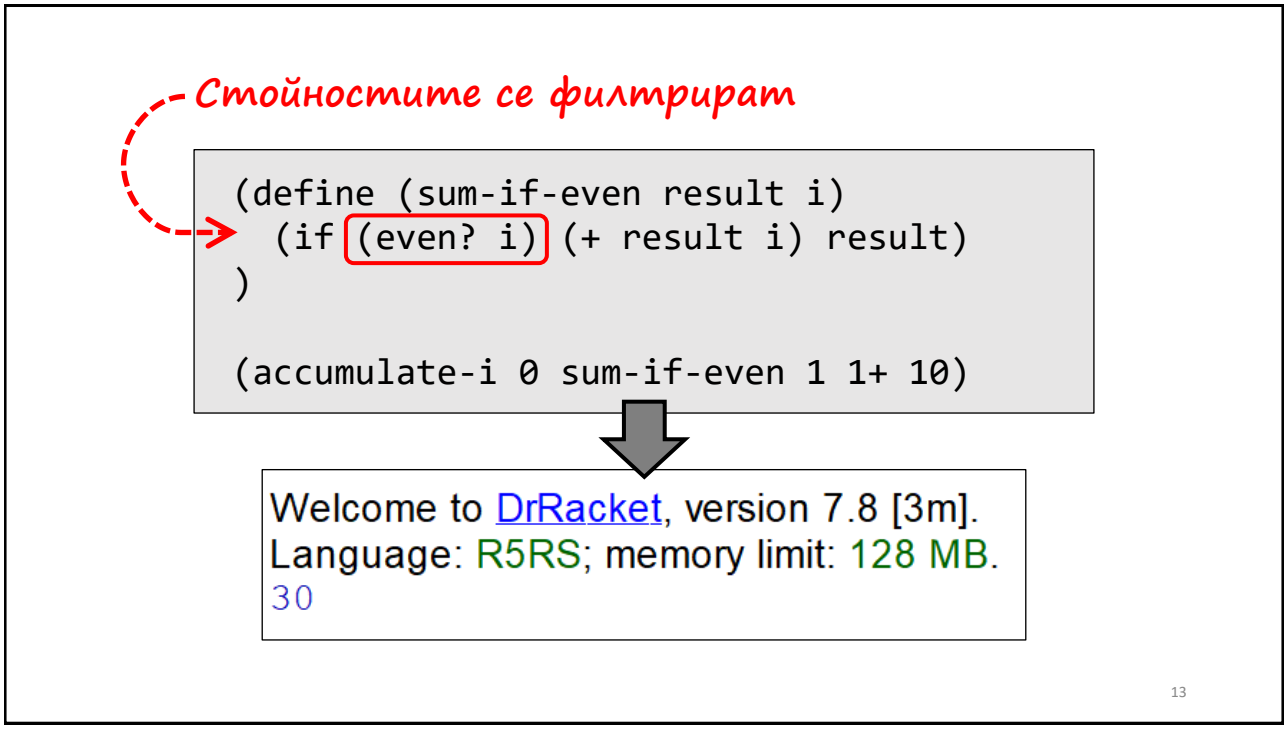
10


Резултатът не се натрупва и не се използва
(define (prn result i) (display i))
(accumulate-i 0 prn 1 1+ 10)
Welcome to DrRacket, version 7.8 [3m].
Language: R5RS; memory limit: 128 MB.
12345678910
11

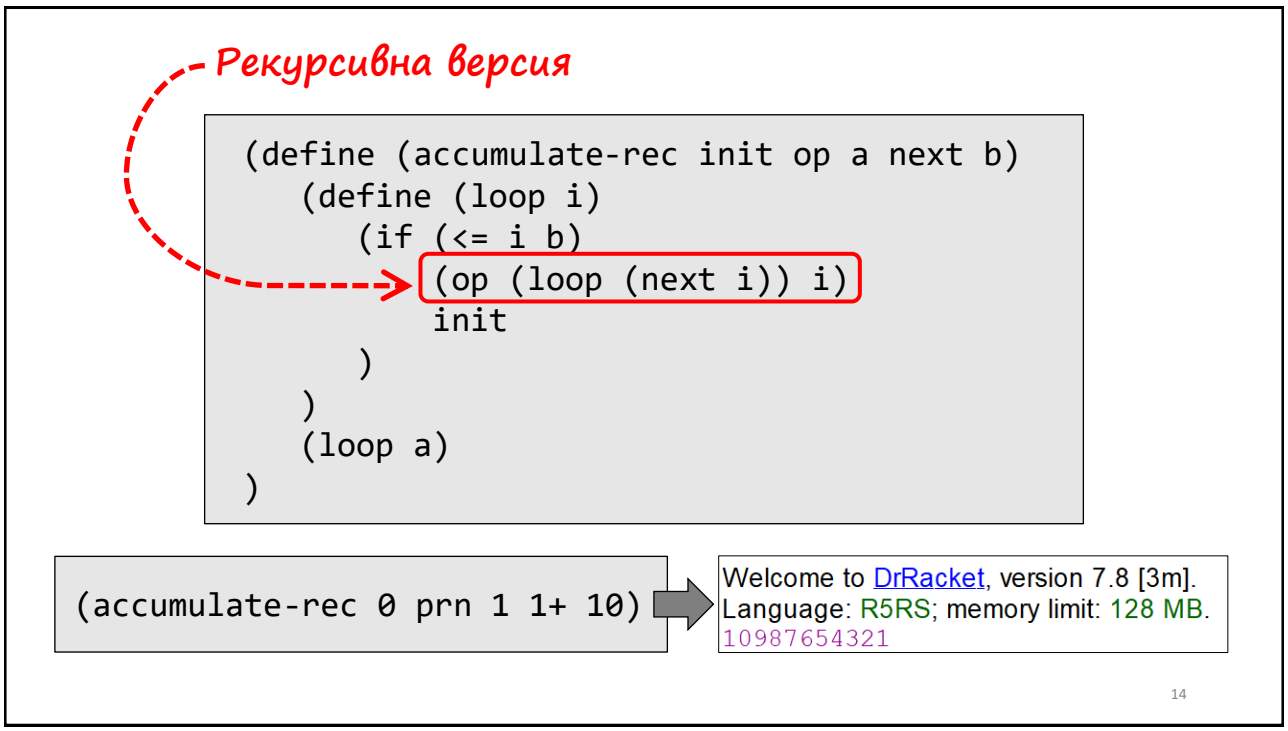
11


Ляво-асоциативно пресмята
... (((0 + 1) + 2) + 3) ... + 10
(accumulate-i 0 + 1 1+ 10)
Welcome to DrRacket, version 7.8 [3m].
Language: R5RS; memory limit: 128 MB.
55
(define (accumulate-i init op a next b)
(define (loop result i)
(if (<= i b)
(loop (op result i) (next i))
result
)
)
(loop init a)
)
12

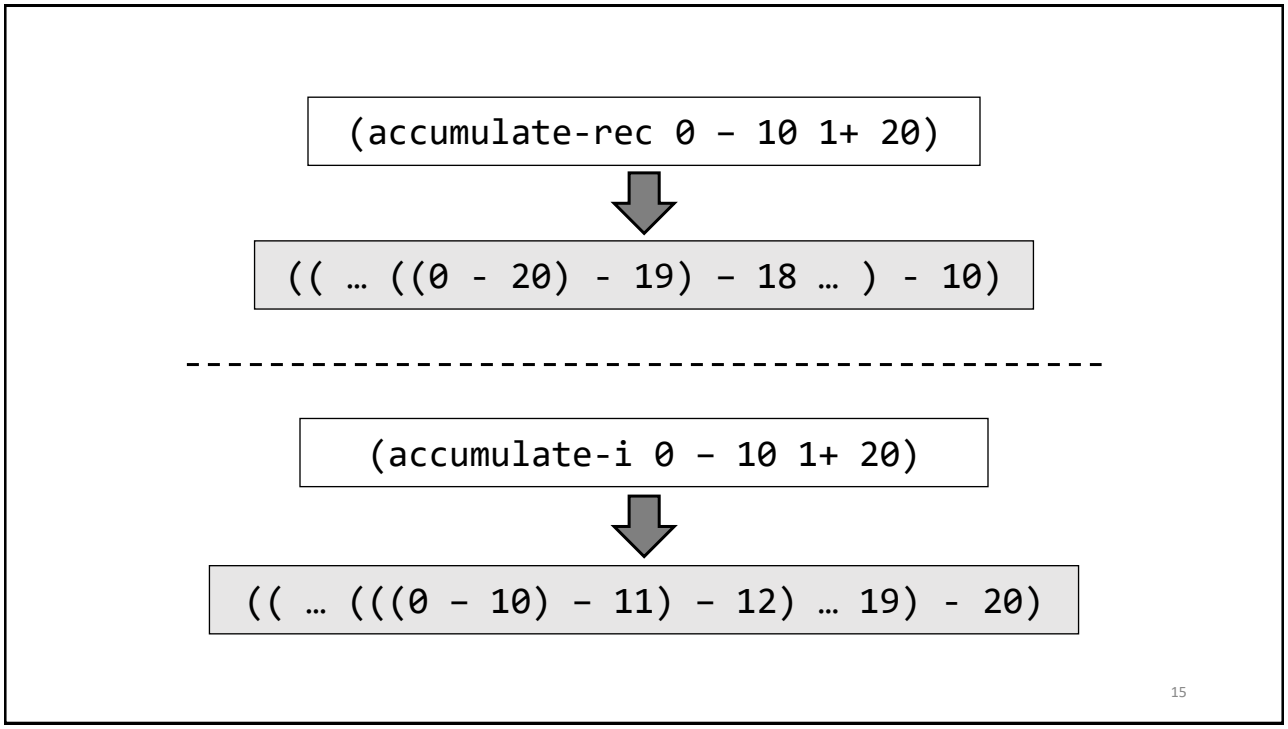
12


Стойностите се филтрират
(define (sum-if-even result i)
(if (even? i) (+ result i) result)
)
(accumulate-i 0 sum-if-even 1 1+ 10)
Welcome to DrRacket, version 7.8 [3m].
Language: R5RS; memory limit: 128 MB.
30
13

13


Рекурсивна версия
(define (accumulate-rec init op a next b)
(define (loop i)
(if (<= i b)
(op (loop (next i)) i)
init
)
)
(loop a)
)
(accumulate-rec 0 prn 1 1+ 10)
Welcome to DrRacket, version 7.8 [3m].
Language: R5RS; memory limit: 128 MB.
10987654321
14

14

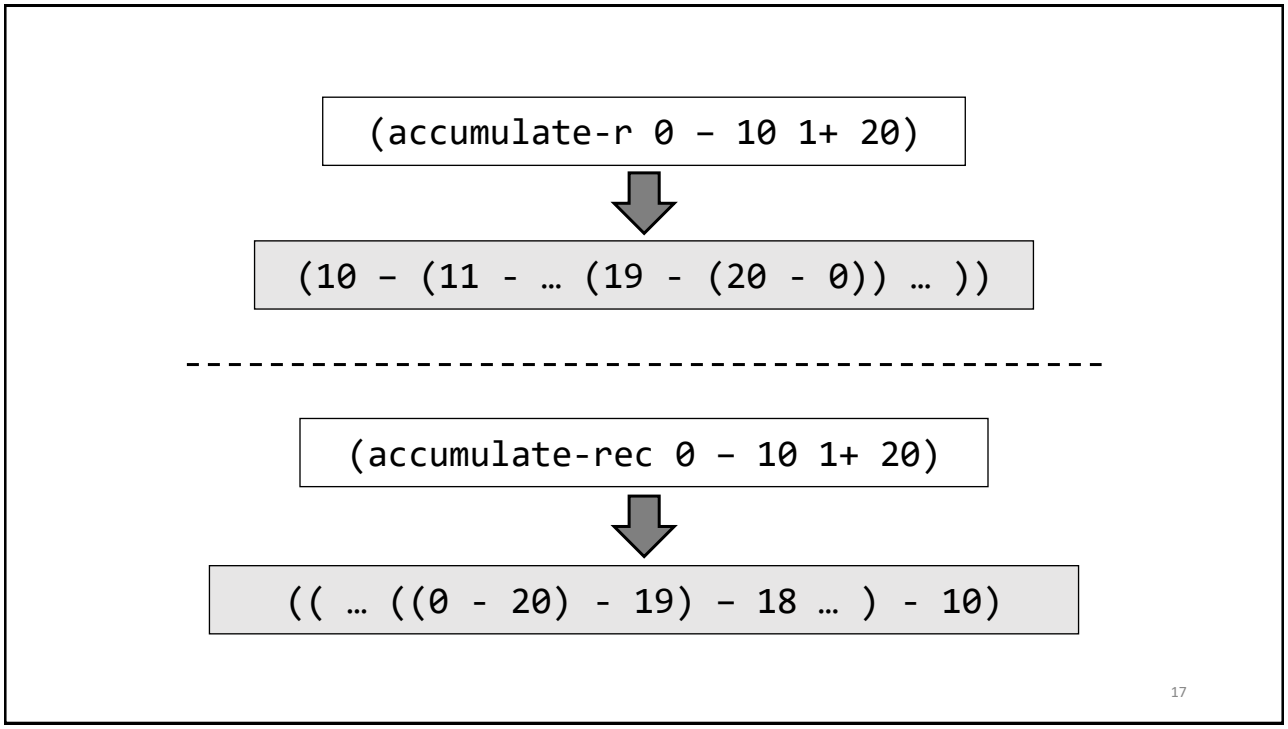


15

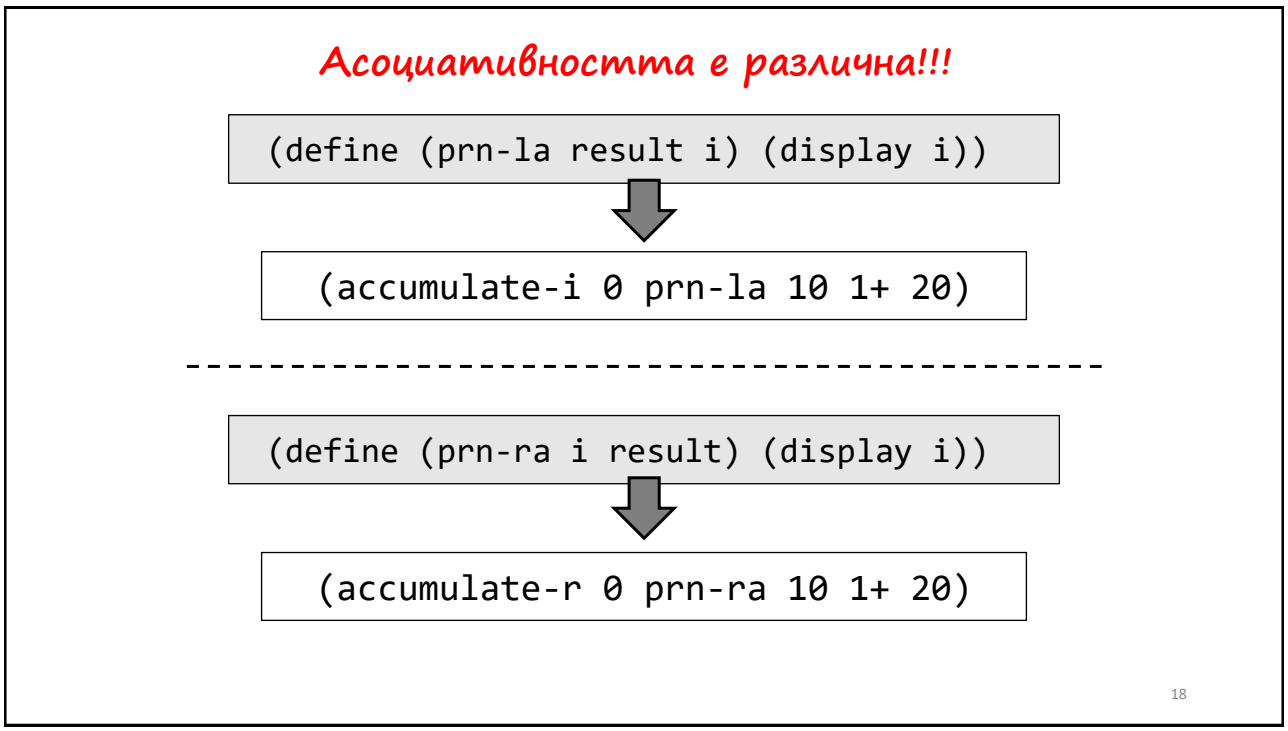
```
(define (accumulate-r init op a next b)
   (define (loop i)
      (if (<= i b)
          (op i (loop (next i)) )
          init
      )
   )
   (loop a)
)
```

16

16


(accumulate-r 0 – 10 1+ 20)
(10 – (11 - … (19 - (20 - 0)) …))
(accumulate-rec 0 – 10 1+ 20)
((… ((0 - 20) - 19) – 18 …) - 10)
17

17


Асоциативността е различна!!!
(define (prn-la result i) (display i))
(accumulate-i 0 prn-la 10 1+ 20)
(define (prn-ra i result) (display i))
(accumulate-r 0 prn-ra 10 1+ 20)
18

18

```
result = init;
for(i = a; i < b; i = next(i))
   result = op( term(i) );
```

```
(define (accumulate op term init a next b)
  (define (loop i)
      (if (<= i b)
          (op (term i) (loop (next i)) )
          init
  ))
  (loop a)
)
```

19

19

Сума на всички числа в [a, b]

```
(define (id x) x)

(define (1+ n) (+ n 1))

(define (sum-int a b)
  (accumulate + id 0 a 1+ b)
)
```

20

20

Сума на редицата $x^a + x^{a+1} + \ldots x^b$

```
(define (sum-row base a b)
  (define (1+ n) (+ n 1))
  (define (term t) (expt base t))
  (accumulate + term 0 a 1+ b)
)
```

21

21

Функционалността може да се мести, също както и в цикъла for!

```
(define (sum-row base a b)
  (define (1+ n) (+ n 1))
  (define (id i) i)
  (define (sum i result) (+ result (expt base i)))
  (accumulate sum id 0 a 1+ b)
)
```

22

22

Сума на четните числа в [a, b] (версия 1)

```
(define (2+ n) (+ n 2))

(define (sum-even a b)
  (accumulate +
              id
              0
              (if (even? a) a (+ a 1))
              2+
              b)
)
```

23

23

Итеративна функция accumulate

```
(define (accumulate-i op term init a next b)

  (define (loop i result)
     (if (> i b) result
         (loop (next i) (op result (term i)) )
  ))

  (loop a init)
)
```

24

24

Сума на четните числа в [a, b] (версия 2)

Работи за рекурсивната версия на accumulate, но не и за итеративната!!!

```
(define (even+ i result)
   (if (even? i) (+ i result) result)
)

(define (sum-even a b)
  (accumulate even+ id 0 a 1+ b)
)
```

25

Сума на четните числа в [10, 20] (версия 2)

```
(accumulate even+ id 0 10 1+ 20)
```

↓

```
(10 + (11 + (12 + … (19 + (20 + 0)) … ))
```

```
(accumulate-i even+ id 0 10 1+ 20)
```

↓

```
… (((0 + 10) + 11) + 12) …
```

26

Сума на простите числа в [a, b]

```
(define (next-prime a)
   (if (prime? (+ a 1))
       (+ a 1)
       (next-prime (+ a 1))
))
(define (sum-prime a b)
  (accumulate + id 0
              (if (prime? a) a (next-prime a)
              next-prime
              b)
)
```

27

27

```
(define (filter-accumulate p? op term init a next b)
  (define (loop i)
     (cond ((> i b) init)
           ((p? i) (op (term i) (loop (next i))) )
           (else (loop (next i)))
  ))
  (loop a)
)
```

28

28

Сума на простите числа в [a, b]

```
(define (sum-prime a b)
  (filter-accumulate prime?
                     +
                     id
                     0
                     a
                     1+
                     b)
)
```

29

29

```
(define (next-prime a)
   (if (prime? (+ a 1))
       (+ a 1)
       (next-prime (+ a 1))
))
(define (sum-prime a b)
  (accumulate + id 0
              (if (prime? a) a (next-prime a)
              next-prime
              b) )
```

```
(define (sum-prime a b)
  (filter-accumulate prime? + id 0 a 1+ b) )
```

30

30

31