

Функции от по-висок ред – част 1

Трифон Трифонов

Функционално програмиране, 2024/25 г.

16 октомври 2024 г.

Тази презентация е достъпна под лиценза Creative Commons Признание-Некомерсиално-Споделяне на споделеното 4.0 Международен 

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → ?

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → ?

$$\begin{aligned} \sin(0) &= 0 \\ e^{\frac{1}{e}} &\neq 1 \end{aligned}$$

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0) → #t`
- (`(fixed-point? exp 1) → #f`

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → ?

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → ?

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → 24

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`(define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → 24
- (`(define (id x) x)`



Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → 24
- (`define (id x) x)`
- (`(branch number? log id "1")` → ?

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → 24
- (`define (id x) x)`
- (`(branch number? log id "1")` → "1"

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → 24
- (`define (id x) x)`
- (`(branch number? log id "1")` → "1"
- (`(branch string? number? procedure? symbol?)` → ?

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) (if (p? x) (f x) (g x)))`
- (`(branch odd? exp fact 4)` → 24
- (`define (id x) x)`
- (`(branch number? log id "1")` → "1"
- (`(branch string? number? procedure? symbol?)` → #t

Подаване на функции като параметри

В Scheme функциите са „първокласни“ стойности.

Примери:

- (`define (fixed-point? f x) (= (f x) x))`
- (`(fixed-point? sin 0)` → #t
- (`(fixed-point? exp 1)` → #f
- (`(fixed-point? expt 0)` → Грешка!
- (`define (branch p? f g x) ((if (p? x) f g) x))`
- (`(branch odd? exp fact 4)` → 24
- (`define (id x) x)`
- (`(branch number? log id "1")` → "1"
- (`(branch string? number? procedure? symbol?)` → #t

Функции от по-висок ред

Дефиниция

Функция, която приема функция за параметър или връща функция като резултат се нарича *функция от по-висок ред*.

Функции от по-висок ред

Дефиниция

Функция, която приема функция за параметър или връща функция като резултат се нарича *функция от по-висок ред*.

- fixed-point? и branch са функции от по-висок ред

Функции от по-висок ред

$$\sum_{n=1}^{\infty} f(n)$$

Дефиниция

$$\frac{d}{dx} : (\mathbb{R} \rightarrow \mathbb{R}) \rightarrow (\mathbb{R} \rightarrow \mathbb{R})$$

Функция, която приема функция за параметър или връща функция като резултат се нарича *функция от по-висок ред*.

- fixed-point? и branch са функции от по-висок ред
- Примери за математически функции от по-висок ред?

$$o : (\mathbb{R} \rightarrow \mathbb{R})^2 \rightarrow (\mathbb{R} \rightarrow \mathbb{R})$$

Функции от по-висок ред

$$\int dx : (\mathbb{R} \rightarrow \mathbb{R}) \rightarrow (\mathbb{R} \rightarrow (\mathbb{R} \rightarrow \mathbb{P}))$$

Дефиниция

Функция, която приема функция за параметър или връща функция като резултат се нарича *функция от по-висок ред*.

- fixed-point? и branch са функции от по-висок ред
- Примери за математически функции от по-висок ред?

$$\int x dx = \frac{x^2}{2} + C = F_{(C)}(x)$$

$$F_{(c)}(x) = \frac{x^2}{2} + c$$

Функции от по-висок ред

$$\{ \text{дължината на} : (\mathbb{R} \rightarrow \mathbb{R}) \times \mathbb{R}^2 \rightarrow \mathbb{R}$$

Дефиниция

Функция, която приема функция за параметър или връща функция като резултат се нарича *функция от по-висок ред*.

- fixed-point? и branch са функции от по-висок ред
- Примери за математически функции от по-висок ред?

Функции от по-висок ред

Дефиниция

Функция, която приема функция за параметър или връща функция като резултат се нарича *функция от по-висок ред*.

- fixed-point? и branch са функции от по-висок ред
- Примери за математически функции от по-висок ред?
- Всички функции в λ -смятането са от по-висок ред!

Задачи за сумиране

Задача: Да се пресметнат следните суми:

① $k^2 + (k+1)^2 + \dots + 100^2$ за $k \leq 100$

② $\int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$

③ $x + e^x + e^{e^x} + e^{e^{e^x}} + \dots$ докато поредното събираме е $\leq 10^{1000}$

Задачи за сумиране

Задача: Да се пресметнат следните суми:

- ① $k^2 + (k+1)^2 + \dots + 100^2$ за $k \leq 100$
- ② $\int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$
- ③ $x + e^x + e^{e^x} + e^{e^{e^x}} + \dots$ докато поредното събираме е $\leq 10^{1000}$

```
(define (sum1 k)
  (if (> k 100) 0 (+ (* k k) (sum1 (+ k 1)))))
```

$$\int_a^b f(x)dx \approx \Delta x f(a) + \int_a^{a+\Delta x} f(x)dx$$

Задачи за сумиране

Задача: Да се пресметнат следните суми:

$$\textcircled{1} \quad k^2 + (k+1)^2 + \dots + 100^2 \text{ за } k \leq 100$$

$$\textcircled{2} \quad \int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$$

$$\textcircled{3} \quad x + e^x + e^{e^x} + e^{e^{e^x}} + \dots \text{ докато поредното събирамео е } \leq 10^{1000}$$

```
(define (sum1 k)
  (if (> k 100) 0 (+ (* k k) (sum1 (+ k 1)))))
```

```
(define (sum2 a b f dx)
  (if (> a b) 0 (+ (* dx (f a)) (sum2 (+ a dx) b f dx))))
```

$$u \leq l$$

$$y := e^x$$

Задачи за сумиране

Задача: Да се пресметнат следните суми:

$$\textcircled{1} \quad k^2 + (k+1)^2 + \dots + 100^2 \text{ за } k \leq 100$$

$$\textcircled{2} \quad \int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$$

$$\textcircled{3} \quad x + e^x + e^{e^x} + e^{e^{e^x}} + \dots \text{ докато поредното събираме } e \leq 10^{1000}$$

```
(define (sum1 k)
  (if (> k 100) 0 (+ (* k k) (sum1 (+ k 1)))))
```

```
(define (sum2 a b f dx)
  (if (> a b) 0 (+ (* dx (f a)) (sum2 (+ a dx) b f dx))))
```

```
(define (sum3 x)
  (if (> x (expt 10 1000)) 0 (+ x (sum3 (exp x)))))
```

Задачи за сумиране

Задача: Да се пресметнат следните суми:

- ① $k^2 + (k+1)^2 + \dots + 100^2$ за $k \leq 100$
- ② $\int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$
- ③ $x + e^x + e^{e^x} + e^{e^{e^x}} + \dots$ докато поредното събираме $e \leq 10^{1000}$

```
(define (sum1 k)
  (if (> k 100) 0 (+ (* k k) (sum1 (+ k 1)))))
```

```
(define (sum2 a b f dx)
  (if (> a b) 0 (+ (* dx (f a)) (sum2 (+ a dx) b f dx))))
```

```
(define (sum3 x)
  (if (> x (expt 10 1000)) 0 (+ x (sum3 (exp x)))))
```

Задачи за сумиране

Задача: Да се пресметнат следните суми:

$$\textcircled{1} \quad k^2 + (k+1)^2 + \dots + 100^2 \text{ за } k \leq 100$$

$$\textcircled{2} \quad \int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$$

$$\textcircled{3} \quad x + e^x + e^{e^x} + e^{e^{e^x}} + \dots \text{ докато поредното събираме } e \leq 10^{1000}$$

```
(define (sum1 k)
  (if (> k 100) 0 (+ (* k k) (sum1 (+ k 1)))))
```

```
(define (sum2 a b f dx)
  (if (> a b) 0 (+ (* dx (f a)) (sum2 (+ a dx) b f dx))))
```

```
(define (sum3 x)
  (if (> x (expt 10 1000)) 0 (+ x (sum3 (exp x)))))
```

Задачи за сумиране

Задача: Да се пресметнат следните суми:

- ① $k^2 + (k+1)^2 + \dots + 100^2$ за $k \leq 100$
- ② $\int_a^b f(x)dx \approx \Delta x [f(a) + f(a + \Delta x) + f(a + 2\Delta x) + \dots + f(b)]$
- ③ $x + e^x + e^{e^x} + e^{e^{e^x}} + \dots$ докато поредното събираме $e \leq 10^{1000}$

```
(define (sum1 k)
  (if (> k 100) 0 (+ (* k k) (sum1 (+ k 1)))))
```

```
(define (sum2 a b f dx)
  (if (> a b) 0 (+ (* dx (f a)) (sum2 (+ a dx) b f dx))))
```

```
(define (sum3 x)
  (if (> x (expt 10 1000)) 0 (+ x (sum3 (exp x)))))
```

Обобщена функция за сумиране

Да се напише функция от по-висок ред `sum`, която пресмята сумата:

$$\sum_{\substack{i=a \\ i \rightarrow \text{next}(i)}}^b \text{term}(i).$$

Обобщена функция за сумиране

Да се напише функция от по-висок ред sum, която пресмята сумата:

$$\sum_{\substack{i=a \\ i \rightarrow \text{next}(i)}}^b \text{term}(i) = \text{term}(a) + \sum_{i=\text{next}(a)}^b \text{term}(i)$$

```
(define (sum a b term next)
  (if (> a b) 0 (+ (term a) (sum (next a) b term next))))
```

Приложения на sum

Решение на задачите за суми чрез sum:

$$\sum_{i=k}^{100} i^2$$

Приложения на sum

Решение на задачите за суми чрез sum:

$$\sum_{i=k}^{100} i^2$$

```
(define (square x) (* x x))
(define (1+ x) (+ x 1))
(define (sum1 k) (sum k 100 square 1+))
```

Приложения на sum

Решение на задачите за суми чрез sum:

$$\sum_{i=k}^{100} i^2$$

```
(define (square x) (* x x))
(define (1+ x) (+ x 1))
(define (sum1 k) (sum k 100 square 1+))
```

$$\sum_{\substack{i=a \\ i \rightarrow i + \Delta x}}^b \Delta x f(i)$$

Приложения на sum

Решение на задачите за суми чрез sum: (+ x 2)

$$\sum_{i=k}^{100} i^2$$

```
(define (square x) (* x x))
(define (1+ x) (+ x 1))
(define (sum1 k) (sum k 100 square 1+))
```

$$\sum_{\substack{i=a \\ i \rightarrow i+\Delta x}}^b \Delta x f(i)$$

```
(define (sum2 a b f dx)
  (define (term x) (* dx (f x)))
  (define (next x) (+ x dx)))
  (sum a b term next))
```

Приложения на sum

Решение на задачите за суми чрез sum:

$$\sum_{i=k}^{100} i^2$$

```
(define (square x) (* x x))
(define (1+ x) (+ x 1))
(define (sum1 k) (sum k 100 square 1+))
```

$$\Delta x \sum_{\substack{i=a \\ i \rightarrow i+\Delta x}}^b f(i)$$

```
(define (sum2 a b f dx)
  (define (next x) (+ x dx))
  (* dx (sum a b f next)))
```

Приложения на sum

Решение на задачите за суми чрез sum:

$$\sum_{i=k}^{100} i^2$$

```
(define (square x) (* x x))
(define (1+ x) (+ x 1))
(define (sum1 k) (sum k 100 square 1+))
```

$$\Delta x \sum_{\substack{i=a \\ i \rightarrow i+\Delta x}}^b f(i)$$

```
(define (sum2 a b f dx)
  (define (next x) (+ x dx))
  (* dx (sum a b f next)))
```

$$\sum_{\substack{i=x \\ i \rightarrow e^i}}^{10^{1000}} i$$

Приложения на sum

Решение на задачите за суми чрез sum:

$$\sum_{i=k}^{100} i^2$$

```
(define (square x) (* x x))
(define (1+ x) (+ x 1))
(define (sum1 k) (sum k 100 square 1+))
```

$$\Delta x \sum_{\substack{i=a \\ i \rightarrow i+\Delta x}}^b f(i)$$

```
(define (sum2 a b f dx)
  (define (next x) (+ x dx))
  (* dx (sum a b f next)))
```

$$\sum_{\substack{i=x \\ i \rightarrow e^i}}^{10^{1000}} i$$

```
(define (sum3 x)
  (sum x (expt 10 1000) id exp))
```

Обобщена функция за произведение

Да се напише функция от по-висок ред `product`, която пресмята:

$$\prod_{\substack{i=a \\ i \rightarrow \text{next}(i)}}^b \text{term}(i).$$

Обобщена функция за произведение

Да се напише функция от по-висок ред `product`, която пресмята:

$$\prod_{\substack{i=a \\ i \rightarrow \text{next}(i)}}^b \text{term}(i).$$

```
(define (prod a b term next)
  (if (> a b) 1 (* (term a) (prod (next a) b term next))))
```

Обобщена функция за произведение

Да се напише функция от по-висок ред `product`, която пресмята:

$$\prod_{\substack{i=a \\ i \rightarrow \text{next}(i)}}^b \text{term}(i).$$

```
(define (prod a b term next)
  (if (> a b) 1 (* (term a) (prod (next a) b term next))))  
  
(define (sum a b term next)
  (if (> a b) 0 (+ (term a) (sum (next a) b term next))))
```

Обобщена функция за произведение

Да се напише функция от по-висок ред `product`, която пресмята:

$$\prod_{\substack{i=a \\ i \rightarrow \text{next}(i)}}^b \text{term}(i).$$

```
(define (prod a b term next)
  (if (> a b) [1] (* (term a) (prod (next a) b term next)))))

(define (sum a b term next)
  (if (> a b) [0] (+ (term a) (sum (next a) b term next))))
```

Обобщена функция за натрупване

Да се напише функция, която пресмята

$$\text{term}(a) \oplus \left(\text{term}(\text{next}(a)) \oplus \left(\dots \oplus (\text{term}(b) \oplus \perp) \dots \right) \right),$$

където \oplus е двуместна операция,
а \perp е нейната „нулева стойност“, т.е. $x \oplus \perp = x$.

b
term(i)
i = a
i -> next(i)

Обобщена функция за натрупване

Да се напише функция, която пресмята

$$\text{term}(a) \oplus \left(\text{term}(\text{next}(a)) \oplus \left(\dots \oplus (\text{term}(b) \oplus \perp) \dots \right) \right),$$

където \oplus е двуместна операция,
а \perp е нейната „нулева стойност“, т.е. $x \oplus \perp = x$.

```
(define (accumulate op nv a b term next)
  (if (> a b) nv
      (op (term a) (accumulate op nv (next a) b term next))))
```

Обобщена функция за натрупване

Да се напише функция, която пресмята

$$\text{term}(a) \oplus \left(\text{term}(\text{next}(a)) \oplus \left(\dots \oplus (\text{term}(b) \oplus \perp) \dots \right) \right),$$

където \oplus е двуместна операция,
а \perp е нейната „нулева стойност“, т.е. $x \oplus \perp = x$.

```
(define (accumulate op nv a b term next)
  (if (> a b) nv
      (op (term a) (accumulate op nv (next a) b term next))))
```

```
(define (sum a b term next) (accumulate + 0 a b term next))
(define (product a b term next) (accumulate * 1 a b term next))
```

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- `(lambda ({<параметър>}) <тяло>)`

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- `(lambda ({<параметър>}) <тяло>)`
- Оценява се до функционален обект със съответните параметри и тяло

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- `(lambda ({<параметър>}) <тяло>)`
- Оценява се до функционален обект със съответните параметри и тяло
- Анонимната функция пази указател към средата, в която е оценена

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- `(lambda ({<параметър>}) <тяло>)`
- Оценява се до функционален обект със съответните параметри и тяло
- Анонимната функция пази указател към средата, в която е оценена
- Примери:

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- `(lambda ({<параметър>}) <тяло>)`
- Оценява се до функционален обект със съответните параметри и тяло
- Анонимната функция пази указател към средата, в която е оценена
- Примери:
 - `(lambda (x) (+ x 3))` → #<procedure>

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- (`lambda` ({<параметър>}) <тяло>)
- Оценява се до функционален обект със съответните параметри и тяло
- **Анонимната функция пази указател към средата, в която е оценена**
- Примери:
 - (`lambda` (x) (+ x 3)) → #<procedure>
 - ((`lambda` (x) (+ x 3)) 5) → 8

Анонимни функции

Можем да конструираме параметрите на функциите от по-висок ред „на място“, без да им даваме имена!

- (`lambda` ({<параметър>}) <тяло>)
- Оценява се до функционален обект със съответните параметри и тяло
- Анонимната функция пази указател към средата, в която е оценена
- Примери:
 - (`lambda` (x) (+ x 3)) → #<procedure>
 - ((`lambda` (x) (+ x 3)) 5) → 8
 - (`define` (<име> <параметри>) <тяло>)
↔
`(define` <име> (`lambda` (<параметри>) <тяло>))

Примери

```
(define (integral a b f dx)
  (* dx (accumulate + 0 a b f (lambda (x) (+ x dx)))))
```

Примери

```
(define (integral a b f dx)
  (* dx (accumulate + 0 a b f (lambda (x) (+ x dx)))))
```

Задача: Как можем да реализираме с accumulate:

- $n!$

A handwritten diagram illustrating the calculation of $n!$ using the accumulate function. It shows a red fraction bar with n above it and a red dot below it. To its right is a yellow oval containing a red dot above a red letter i . Below these, the equation $i = 1$ is written in red.

Примери

```
(define (integral a b f dx)
  (* dx (accumulate + 0 a b f (lambda (x) (+ x dx)))))
```

Задача: Как можем да реализираме с accumulate:

- $n!$
- x^n

$$\prod_{i=1}^n \text{ } \textcircled{X}$$

Примери

```
(define (integral a b f dx)
  (* dx (accumulate + 0 a b f (lambda (x) (+ x dx)))))
```

Задача: Как можем да реализираме с accumulate:

- $n!$
- x^n
- $\sum_{i=0}^n \frac{x^i}{i!}$

$$\begin{aligned}
 e^x &\approx 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots \\
 &= 1 + x \left(1 + \frac{x}{2} \left(1 + \frac{x}{3} \left[1 + \dots \right] \right) \right)
 \end{aligned}$$

Примери

```
(define (integral a b f dx)
  (* dx (accumulate + 0 a b f (lambda (x) (+ x dx)))))
```

Задача: Как можем да реализираме с accumulate:

- $n!$
- x^n
- $\sum_{i=0}^n \frac{x^i}{i!}$
- $\exists x \in [a; b] p(x)$

$$\begin{array}{c}
 p(a) \vee p(a+1) \vee \dots \vee p(b) \\
 \bigvee_{i=a}^b p(i)
 \end{array}$$