Scheme süntaks

Scheme süntaks

- Programm ja definitsioonid

  ⟨*programm*⟩ ::= ⟨*definition*⟩ . . . ⟨*definition*⟩

  ⟨*definition*⟩ ::= (define ⟨*variable*⟩ ⟨*expression*⟩)

Scheme süntaks

- Programm ja definitsioonid

  ⟨*programm*⟩ ::= ⟨*definition*⟩ . . . ⟨*definition*⟩

  ⟨*definition*⟩ ::= (define ⟨*variable*⟩ ⟨*expression*⟩)

```
> (define x 1)
> x
1
> (define y (+ x 1))
> y
2
> (define x y)
> x
2
```

Scheme avaldised

Scheme avaldised

- Avaldiste süntaks

$$
\begin{array}{lll}
\langle expr \rangle & ::= & \langle literal \rangle \\
 & | & \langle variable \rangle \\
 & | & (\langle expr \rangle \ldots \langle expr \rangle)
\end{array}
$$

Scheme avaldised

- Avaldiste süntaks

  $\langle expr \rangle$ ::= $\langle literal \rangle$
  | $\langle variable \rangle$
  | ($\langle expr \rangle$ ... $\langle expr \rangle$)

- Literalid
  - Numbrid `123`, `3.1415`
  - Tõeväärtused `#t`, `#f`
  - “Tähed” `#\a`, `#\space`
  - Stringid `"See on string"`

Scheme avaldised

- Identifikaatorid `x`, `*`, `one+two=three`
  - Muutujad
  - Võtmesõnad
  - Sümbolid

Scheme avaldised

- Identifikaatorid `x`, `*`, `one+two=three`
  - Muutujad
  - Võtmesõnad
  - Sümbolid
- Protseduuride aplikatsoonid

```
> (+ 1 2 3 4)
10
> (* (/ 10 2) (- 5 3))
10
```

Scheme avaldised

- Tingimusavaldis

  $\langle expr \rangle$ ::= ...

  | (if $\langle expr \rangle$ $\langle expr \rangle$ $\langle expr \rangle$)

Scheme avaldised

- Tingimusavaldis

  $\langle expr \rangle$ ::= ...
  | (if $\langle expr \rangle$ $\langle expr \rangle$ $\langle expr \rangle$)

```
> (if #t 1 2)
1
> (if #f 1 2)
2
> (if (if (= 1 (- 2 1)) #t (/ 1 0)) 3 4)
3
```

Primitiivtüübid

Primitiivtüübid

- Numbrid — `number?`

```
> (* (- -2 -4) 10.5)
21.0
> (/ 2 3)
2/3
> (= 3 (* 2 1.5))
#t
> (< (+ (* 2 3) (- 4 0)) 10)
#f
```

Primitiivtüübid

- Numbrid — `number?`

```
> (* (- -2 -4) 10.5)
21.0
> (/ 2 3)
2/3
> (= 3 (* 2 1.5))
#t
> (< (+ (* 2 3) (- 4 0)) 10)
#f
```

- Tõeväärtused — `boolean?`

```
> (eq? (not (boolean? 7)) (number? #t))
#f
```

Primitiivtüübid

- Tähed — `char?`

```
> (char=? #\A #\a)
#f
> (char<? #\A #\a)
#t
> (char->integer #\newline)
10
> (char-numeric? #\7)
#t
> (char-alphabetic? #\7)
#f
```

Primitiivtüübid

- Stringid — `string?`

```
> (string-length "Hello, world!")
13
> (string-append "Abra" "kadabra")
"Abrakadabra"
> (string->number "1234")
1234
> (string #\A #\b #\r #\a)
"Abra"
> (string-ref "Abra" 2)
#\r
```

Primitiivtüübid

- Sümbolid — `symbol?`
- “Tsiteerimine”

  ⟨*expr*⟩ ::= …
  | (quote ⟨*datum*⟩)
  | ’⟨*datum*⟩

```
> apple
reference to undefined identifier: apple
> (quote apple)
apple
> (eq? (quote apple) 'apple)
#t
```

Andmestruktuurid

- Listid — `list?`

```
> '(1 2 3 4)
(1 2 3 4)
> '(a () ((#t 3)))
(a () ((#t 3)))
> (cons 1 '(2 3))
(1 2 3)
> (list 1 '(2 3))
(1 (2 3))
> (append '(1 (2)) '(a b) '(1))
(1 (2) a b 1)
```

Andmestruktuurid

- Operatsioone listidel

```
> (car '((1 2 3) 4 5))
(1 2 3)
> (cdr '((1 2 3) 4 5))
(4 5)
> (car (cons 'a '(b c)))
a
> (cadr '((1 2 3) 4 5))
4
> (null? '())
#t
```

Andmestruktuurid

- Paarid — `pair?`

```
> '(a . 3)
(a . 3)
> '(a . (b . (() . c)))
(a b () . c)
> (pair? '(a b c d))
#t
> '(a . (b . (c . ())))
(a b c)
> (cdr '(a . b))
b
```

Protseduurid

- Protseduuri tüüp — `procedure?`

```
> (procedure? car)
#t
> (procedure? (cadr (list 1 + 2)))
#t
> ((if (procedure? procedure?) car 'ship)
      '(plane train))
plane
> (apply + '(1 2))
3
```

Protseduurid

- Protseduuride defineerimine

⟨*expr*⟩ ::= ...
| (lambda ⟨*formals*⟩ ⟨*expr*⟩)

⟨*formals*⟩ ::= ⟨*variable*⟩
| (⟨*variable*⟩ ... ⟨*variable*⟩)

```
> (lambda (x) (+ x 1))
#<procedure>
> ((lambda (x) (+ x 1)) 3)
4
> (map (lambda (x) (+ x 1)) '(1 2 3))
(2 3 4)
```

Protseduurid

- Protseduuride defineerimine (järg)

```
> (define fahrenheit-to-celsius
    (lambda (x)
      (* (- x 32) 5/9)))
> (fahrenheit-to-celsius 212)
100
> (define compose
    (lambda (f g)
      (lambda (x)
        (f (g x)))))
> ((compose car cdr) '(a b c))
b
```

Järgmiseks korraks

- Lugeda läbi EOPL ptk. 1
  - Vektorid
  - Muutuva aarsusega protseduurid
- (TYSFD ptk. 1 – 3)
  - Ignoreerida !-lõppevaid protseduure