```
(define generate-depth-iterator
  (lambda (l)
    (define return #f)
    (define start
      (lambda ()
        (define loop
          (lambda (l)
            (if (null? l)
                (quote neco)
                (if (pair? l)
                    (begin (loop (car l))
                           (loop (cdr l)))
                    (call/cc (lambda (new-start)
                      (begin (set! start new-start)
                             (return l))))))))
        (loop l)
        (return '())))
     (lambda ()
       (call/cc (lambda (f)
                  (set! return f)
                  (start))))))
```

Vyhodnocení: na symbol `generate-depth-iterator` se v $\mathcal{P}_g$ naváže uživatelsky definovaná procedura $\langle$`(l)`, ..., $\mathcal{P}_g\rangle$, v nasledujícím textu označovaná jako *generátor*.

```
(define p (generate-depth-iterator (a (b (c (d)) e))))
```

V prostředí $\mathcal{P}_g$ vyhodnocujeme
(define p (generate-depth-iterator (a (b (c (d)) e))))
v prostředí $\mathcal{P}_2$ vyhodnocujeme (generate-depth-iterator '(a (b (c (d)) e))).
nalezena vazba generate-depth-iterator $\longrightarrow$ *generátor* v prostředí $\mathcal{P}_g$; jde o uživatelsky definovanou proceduru.
'(a (b (c (d)) e)) se vyhodnotí na (a (b (c (d)) e)).
Aplikujeme uživatelsky definovanou proceduru *generátor* na argument ((a (b (c (d)) e))).
Vytvoří se nové prostředí $\mathcal{P}_{12}$ s vazbou: l $\longrightarrow$ (a (b (c (d)) e)),
v prostředí $\mathcal{P}_{12}$ vyhodnocujeme tělo *generátoru*.
v prostředí $\mathcal{P}_{12}$ vyhodnocujeme (define return #f): v prostředí $\mathcal{P}_{12}$ na symbol return se naváže #f.
v prostředí $\mathcal{P}_{12}$ vyhodnocujeme

```
(define start
  (lambda ()
    (define loop
      (lambda (l)
        (if (null? l)
            (quote neco)
            (if (pair? l)
                (begin (loop (car l))
                       (loop (cdr l)))
                (call/cc (lambda (new-start)
                  (begin (set! start new-start)
                         (return l)))))))))
  (loop l)
  (return '())))
```

Na symbol start se v $\mathcal{P}_{12}$ naváže uživatelsky definovaná procedura $\langle$(), ..., $\mathcal{P}_{12}\rangle$, v nasledujícím textu označovaná jako *start* α.
V prostředí $\mathcal{P}_{12}$ vyhodnocujeme

```
(lambda ()
     (call/cc (lambda (f)
                (set! return f)
                (start))))
```

výsledek: $\langle$(), (call/cc (lambda (f) (set! return f) (start))), $\mathcal{P}_{12}\rangle$
výsledek: $\langle$(), (call/cc (lambda (f) (set! return f) (start))), $\mathcal{P}_{12}\rangle$
výsledek: $\langle$(), (call/cc (lambda (f) (set! return f) (start))), $\mathcal{P}_{12}\rangle$
Na symbol p se v prostředí $\mathcal{P}_g$ naváže tento výsledek.

```
(p)
```

V prostředí $\mathcal{P}_g$ vyhodnocujeme `(p)`.
nalezena vazba `p`$\longrightarrow\langle$`()`, `(call/cc ... )`, $\mathcal{P}_{12}\rangle$ v prostředí $\mathcal{P}_g$.
Aplikujeme tuto proceduru (bez argumentů).
Vytvoří se nové prázndé prostředí $\mathcal{P}_{13}$, jehož předchůdcem je $\mathcal{P}_{12}$.
v prostředí $\mathcal{P}_{13}$ vyhodnocujeme tělo:

```
(call/cc (lambda (f) (set! return f) (start)))
```

Nalezena vazba `call/cc`$\longrightarrow$*call/cc* v prostředí $\mathcal{P}_g$; jde o proceduru – vyhodnotíme zbytek seznamu.
`(lambda (f) (set! return f) (start))` se vyhodnotí na proceduru $\langle$`(f)`, `(begin (set! return f) (st`
– *receiver*.
Vytvoří se aktuální pokračování *kontinuace* α a předá se *recieveru* (tedy aplikujeme *recieveru* na *kontinuaci* α v prostředí $\mathcal{P}_{13}$.
Vytvoří se nové prostředí $\mathcal{P}_{14}$ s vazbami: `f` $\longrightarrow$ *kontinuace* α,
v prostředí $\mathcal{P}_{14}$ vyhodnocujeme tělo *receiveru* `(begin (set! return f) (start))`
v prostředí $\mathcal{P}_{14}$ vyhodnocujeme `(set! return f)`
nalezena vazba `set!`$\longrightarrow$*set!* v prostředí $\mathcal{P}_g$; speciální forma.
nalezena vazba `f`$\longrightarrow$*kontinuace* α v prostředí $\mathcal{P}_{14}$.
Vazba `return`$\longrightarrow$`#f` v prostředí $\mathcal{P}_{12}$ je změněna na vazbu `return`$\longrightarrow$*kontinuace* α.
výsledek: *nedefinovaná hodnota*

v prostředí $\mathcal{P}_{14}$ vyhodnocujeme `(start)`
nalezena vazba `start`$\longrightarrow$ *start* α v prostředí $\mathcal{P}_{12}$; aplikujeme tuto proceduru (bez argumentů).
Vytvoří se nové prázdné prostředí $\mathcal{P}_{15}$, jehož předkem je $\mathcal{P}_{12}$:
v prostředí $\mathcal{P}_{15}$ vyhodnocujeme tělo procedury *start* α.
v prostředí $\mathcal{P}_{15}$ vyhodnocujeme

```
(define loop
  (lambda (l)
    (if (null? l)
        (quote neco)
        (if (pair? l)
            (begin (loop (car l))
                   (loop (cdr l)))
            (call/cc (lambda (new-start)
              (begin (set! start new-start)
                     (return l))))))))
```

Vytvoří se vazba symbolu `loop` na už. def. proceduru $\langle$`(l)`, ..., $\mathcal{P}_{15}\rangle$, která bude v následujícím textu nazývána *loop*.
výsledek: *nedefinovaná hodnota*

v prostředí $\mathcal{P}_{15}$ vyhodnocujeme `(loop l)`
nalezena vazba `loop`$\longrightarrow$*loop* v prostředí $\mathcal{P}_{15}$; jde o proceduru, vyhodnotíme zbytek seznamu. `l` se vyhodnotí na svou vazbu `(a (b (c (d)) e))` v prostředí $\mathcal{P}_{12}$.
Aplikujeme *loop* na tento seznam.
Vytvoří se nové prostředí $\mathcal{P}_{16}$ s vazbou `l` $\longrightarrow$ `(a (b (c (d)) e))`.
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme tělo procedury *loop*.

```
(if (null? l)
    (quote neco)
    (if (pair? l)
        (begin (loop (car l))
               (loop (cdr l)))
        (call/cc (lambda (new-start)
          (begin (set! start new-start)
                 (return l))))))
```

Jde tedy o *if*-výraz:
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme `(null? l)`
nalezena vazba `null?`⟶*null?* v prostředí $\mathcal{P}_1$. Vyhodnotí se na `#f`, vyhodnotíme tedy hlavní větev *if*-výrazu.
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme

```
(if (pair? l)
    (begin (loop (car l))
           (loop (cdr l)))
    (call/cc (lambda (new-start)
      (begin (set! start new-start)
             (return l)))))
```

Vyhodnotíme podmínku *if*-výrazu (v prostředí $\mathcal{P}_{16}$), výsledkem je `#t`. Vyhodnotíme tedy hlavní větev *if*-výrazu.
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme `(begin (loop (car l)) (loop (cdr l)))`
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme `(loop (car l))`
nalezena vazba `loop`⟶*loop* v prostředí $\mathcal{P}_{15}$, jde o proceduru; vyhodnotíme zbytek seznamu
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme `(car l)`
výsledek: `a`

Aplikujeme *loop* na argument `a`.
Vytvoří se nové prostředí $\mathcal{P}_{17}$ s vazbami: `l` ⟶ `a`,
v prostředí $\mathcal{P}_{17}$ vyhodnocujeme tělo *loop*:

```
(if (null? l)
    (quote neco)
    (if (pair? l)
        (begin (loop (car l))
               (loop (cdr l)))
        (call/cc (lambda (new-start)
          (begin (set! start new-start)
                 (return l))))))
```

Jde tedy o *if*-výraz:
v prostředí $\mathcal{P}_{17}$ vyhodnocujeme podmínku `(null? l)`
výsledek: `#f`

v prostředí $\mathcal{P}_{17}$ vyhodnocujeme

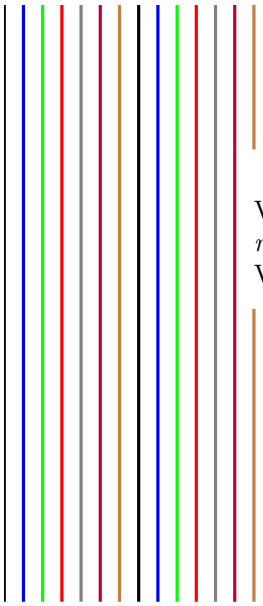
```
(if (pair? l)
    (begin (loop (car l))
           (loop (cdr l)))
    (call/cc (lambda (new-start)
      (begin (set! start new-start)
             (return l)))))
```

Jde tedy o *if*-výraz, vyhodnotíme jeho podmínku.
v prostředí $\mathcal{P}_{17}$ vyhodnocujeme `(pair? l)`
výsledek: `#f`

Vyhodnocujeme tedy alternativní větev *if*-výrazu.
v prostředí $\mathcal{P}_{17}$ vyhodnocujeme

```
(call/cc (lambda (new-start)
  (begin (set! start new-start)
         (return l))))
```

Na `call/cc` je navázána procedura *call/cc*; vyhodnotíme zbytek seznamu.
v prostředí $\mathcal{P}_{17}$ vyhodnocujeme



```
(lambda (new-start)
   (begin (set! start new-start)
          (return l)))
```

výsledek: $\langle$`(new-start)`, `(begin ... (return l))`, $\mathcal{P}_{17}\rangle$ – *receiver*.

Vytvoří se aktuální pokračování (*kontinuace* β) a předá se *recieveru*. Aplikujeme tedy *reciever* na (*kontinuaci* β).
Vytvoří se nové prostředí $\mathcal{P}_{18}$ s vazbou `new-start` $\longrightarrow$ *kontinuace* β,
v prostředí $\mathcal{P}_{18}$ vyhodnocujeme tělo *receiveru*
`(begin (set! start new-start) (return l))`
v prostředí $\mathcal{P}_{18}$ vyhodnocujeme `(set! start new-start)`
nalezena vazba symbolu `start` v prostředí $\mathcal{P}_{12}$ je změněna na *kontinuaci* β.
výsledek: *nedefinovaná hodnota*

v prostředí $\mathcal{P}_{18}$ vyhodnocujeme `(return l)`
nalezena vazba `return`$\longrightarrow$*kontinuace* α v prostředí $\mathcal{P}_{12}$.
`l` je navázáno na symbol `a` v prostředí $\mathcal{P}_{17}$.
escape s hodnotou `a`

Aktivace *kontiuace* α s hodnotou `a`

výsledek: `a`

výsledek: `a`

Slovní popis kontinuací (barvy odpovídají barvám v předcházejícím výpisu):

kontinuace α: v podstatě ukončení vyhodnocování.

- ukončení vyhodnocení výrazu `(p)` v prostředí $\mathcal{P}_g$.

kontinuace β:

- ukončení vyhodnocování výrazu v prostředí $\mathcal{P}_{17}$:

```
(if (pair? l)
    (begin (loop (car l))
           (loop (cdr l)))
    (call/cc (lambda (new-start)
      (begin (set! start new-start)
             (return l)))))
```

- ukončení vyhodnocení těla procedury *loop* v prostředí $\mathcal{P}_{17}$.
- ukončení vyhodnocení výrazu `(loop (car l))` v prostředí $\mathcal{P}_{16}$:
- **vyhodnocení výrazu `(loop (cdr l))` v prostředí** $\mathcal{P}_{16}$:
- ukončení vyhodnocení výrazu `(begin (loop (car l)) (loop (cdr l)))` v prostředí $\mathcal{P}_{16}$.
- ukončení vyhodnocení výrazu v prostředí $\mathcal{P}_{16}$:

```
(if (pair? l)
    (begin (loop (car l))
           (loop (cdr l)))
    (call/cc (lambda (new-start)
      (begin (set! start new-start)
             (return l)))))
```

- ukončení vyhodnocení těla procedury *loop* v prostředí $\mathcal{P}_{16}$.
- ukončení vyhodnocení výrazu `(loop l)` v prostředí $\mathcal{P}_{15}$.
- ukončení těla procedury *start* α v prostředí $\mathcal{P}_{15}$.
- ukončení vyhodnocení výrazu `(start)` v prostředí $\mathcal{P}_{14}$.
- ukončení vyhodnocení výrazu `(begin (set! return f) (start))` v prostředí $\mathcal{P}_{14}$.
- ukončení vyhodnocení výrazu

  ```
  (call/cc (lambda (f) (set! return f) (start)))
  ```

  v prostředí $\mathcal{P}_{13}$.
- ukončení vyhodnocení výrazu `(p)` v prostředí $\mathcal{P}_g$.

```
(p)
```

v prostředí $\mathcal{P}_2$ vyhodnocujeme `(p)`
nalezena vazba `p`$\longrightarrow\langle$`()`, `(call/cc ... )`, $\mathcal{P}_{12}\rangle$ v prostředí $\mathcal{P}_g$.
Aplikujeme tuto proceduru (bez argumentů).
Vytvoří se nové prázndé prostředí $\mathcal{P}_{13}$, jehož předchůdcem je $\mathcal{P}_{19}$.
v prostředí $\mathcal{P}_{19}$ vyhodnocujeme
`(call/cc (lambda (f) (set! return f) (start)))`
nalezena vazba `call/cc`$\longrightarrow$*call/cc* v prostředí $\mathcal{P}_g$; je to procedura.
v prostředí $\mathcal{P}_{19}$ vyhodnocujeme `(lambda (f) (set! return f) (start))`
výsledek: $\langle$`(f)`, `(begin (set! return f) (start))`, $\mathcal{P}_{19}\rangle$ – *receiver*

Vytvoří se aktuální pokračování (*kontinuace* γ) a předá se *recieveru*. Tedy aplikujeme *receiver* na *kontinuaci* γ.
Vytvoří se nové prostředí $\mathcal{P}_{20}$ s vazbou `f` $\longrightarrow$ *kontinuace* γ.
v prostředí $\mathcal{P}_{20}$ vyhodnocujeme `(begin (set! return f) (start))`
v prostředí $\mathcal{P}_{20}$ vyhodnocujeme `(set! return f)`
v prostředí $\mathcal{P}_{12}$ je změněna nalezena vazba `return`$\longrightarrow$*kontinuace* α na vazbu $\longrightarrow$*kontinuace* γ
výsledek: *nedefinovaná hodnota*

v prostředí $\mathcal{P}_{20}$ vyhodnocujeme `(start)`
nalezena vazba `start`$\longrightarrow$*kontinuace* β v prostředí $\mathcal{P}_{12}$.
escape

výsledek: *nedefinovaná hodnota*

výsledek: *nedefinovaná hodnota*

výsledek: *nedefinovaná hodnota*

výsledek: *nedefinovaná hodnota*

v prostředí $\mathcal{P}_{16}$ vyhodnocujeme `(loop (cdr l))`
nalezena vazba `loop`$\longrightarrow$*loop* v prostředí $\mathcal{P}_{15}$; jde o proceduru.
v prostředí $\mathcal{P}_{16}$ vyhodnocujeme `(cdr l)`
`l` je navázáno na seznam `(a (b (c (d)) e))` v prostředí $\mathcal{P}_{16}$. Aplikujeme na něj primitivní proceduru *cdr*.
výsledek: `((b (c (d)) e))`

Aplikujeme uživatelsky definovanou proceduru *loop* na argument `((b (c (d)) e))`.
Vytvoří se nové prostředí $\mathcal{P}_{21}$ s vazbou `l` $\longrightarrow$ `((b (c (d)) e))`,
v prostředí $\mathcal{P}_{21}$ vyhodnocujeme tělo procedury *loop*.
Jde o *if*-výraz, vyhodnotíme jeho podmínku.
v prostředí $\mathcal{P}_{21}$ vyhodnocujeme `(null? l)`
výsledek: `#f`

v prostředí $\mathcal{P}_{21}$ vyhodnocujeme

```
(if (pair? l)
    (begin (loop (car l))
           (loop (cdr l)))
    (call/cc (lambda (new-start)
      (begin (set! start new-start)
             (return l)))))
```

Je to *if*-výraz, vyhodnotíme jeho podmínku
v prostředí $\mathcal{P}_{21}$ vyhodnocujeme `(pair? l)`
výsledek: `#t`

v prostředí $\mathcal{P}_{21}$ vyhodnocujeme `(begin (loop (car l)) (loop (cdr l)))`
v prostředí $\mathcal{P}_{21}$ vyhodnocujeme `(loop (car l))`
nalezena vazba `loop`⟶*loop* v prostředí $\mathcal{P}_{15}$; je to procedura.
v prostředí $\mathcal{P}_{21}$ vyhodnocujeme `(car l)`
Aplikujeme primitivní proceduru *car* na argumenty (`((b (c (d)) e))`).
výsledek: `(b (c (d)) e)`

Aplikujeme uživatelsky definovanou proceduru *loop* na argument `(b (c (d)) e)`.
Vytvoří se nové prostředí $\mathcal{P}_{22}$ s vazbou `l` ⟶ `(b (c (d)) e)`.
v prostředí $\mathcal{P}_{22}$ vyhodnocujeme tělo procedury *loop*.
Jde o *if*-výraz, vyhodnotíme jeho podmínku.
v prostředí $\mathcal{P}_{22}$ vyhodnocujeme `(null? l)`
Aplikujeme primitivní proceduru *null?* na argumenty (`(b (c (d)) e)`).
výsledek: `#f`

v prostředí $\mathcal{P}_{22}$ vyhodnocujeme

```
(if (pair? l)
    (begin (loop (car l))
           (loop (cdr l)))
     (call/cc (lambda (new-start)
       (begin (set! start new-start)
              (return l)))))
```

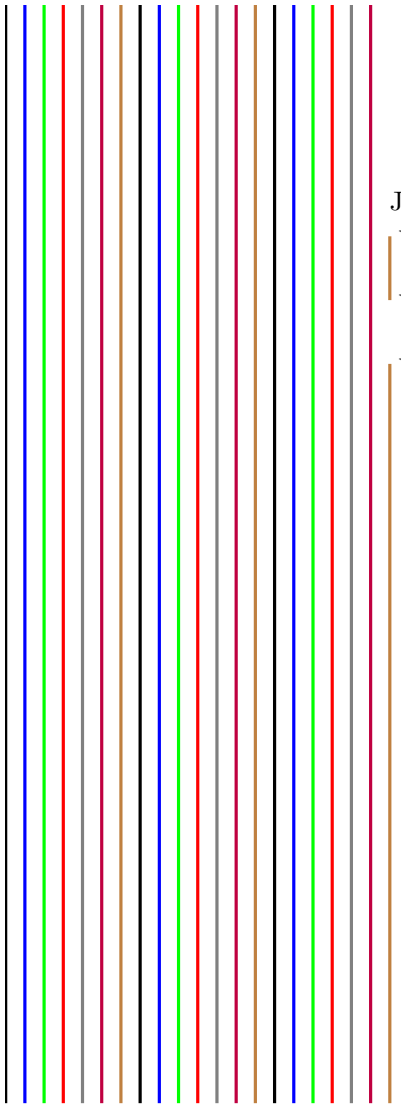
Je to *if*-výraz, vyhodnotíme jeho podmínku
v prostředí $\mathcal{P}_{22}$ vyhodnocujeme `(pair? l)`
Aplikujeme primitivní proceduru *pair?* na argumenty (`(b (c (d)) e)`).
výsledek: `#t`

v prostředí $\mathcal{P}_{22}$ vyhodnocujeme `(begin (loop (car l)) (loop (cdr l)))`
v prostředí $\mathcal{P}_{22}$ vyhodnocujeme `(loop (car l))`
nalezena vazba `loop`⟶*loop* v prostředí $\mathcal{P}_{15}$.
vyhodnotíme zda se jedná o proceduru, spec. formu, nebo makro: jde o proceduru.
v prostředí $\mathcal{P}_{22}$ vyhodnocujeme `(car l)`
Aplikujeme primitivní proceduru *car* na argumenty (`(b (c (d)) e)`).
výsledek: `b`

Aplikujeme uživatelsky definovanou proceduru *loop* na argument `b`.
Vytvoří se nové prostředí $\mathcal{P}_{23}$ s vazbou: `l` ⟶ `b`,
v prostředí $\mathcal{P}_{23}$ vyhodnocujeme tělo procedury *loop*.
Je to *if*-výraz, vyhodnotíme jeho podmínku.
v prostředí $\mathcal{P}_{23}$ vyhodnocujeme `(null? l)`
Aplikujeme primitivní proceduru *null?* na argument `b`.
výsledek: `#f`

v prostředí $\mathcal{P}_{23}$ vyhodnocujeme

```
(if (pair? l)
```



```
(begin (loop (car l))
       (loop (cdr l)))
(call/cc (lambda (new-start)
  (begin (set! start new-start)
         (return l)))))
```

Je to *if*-výraz, vyhodnotíme jeho podmínku.
v prostředí $\mathcal{P}_{23}$ vyhodnocujeme `(pair? l)`
Aplikujeme primitivní proceduru *pair?* na argumenty (`b`).
výsledek: `#f`

v prostředí $\mathcal{P}_{23}$ vyhodnocujeme
`(call/cc (lambda (new-start) (begin (set! start new-start)`
`(return l))))`
nalezena vazba `call/cc`$\longrightarrow$*call/cc* v prostředí $\mathcal{P}_g$; je to procedura.
v prostředí $\mathcal{P}_{23}$ vyhodnocujeme `(lambda (new-start) (begin (set!`
`start new-start) (return l)))`
výsledek: $\langle$`(new-start), (begin (set! start new-start) (return l)),`
– *receiver*.

Vytvoří se aktuální pokračování a předá se *recieveru*.
Aplikujeme uživatelsky definovanou proceduru *reciever* na argumenty *kontinuaci* δ.
Vytvoří se nové prostředí $\mathcal{P}_{24}$ s vazbou `new-start` $\longrightarrow$ *kontinuace* δ.
v prostředí $\mathcal{P}_{24}$ vyhodnocujeme `(begin (set! start new-start)`
`(return l))`
v prostředí $\mathcal{P}_{24}$ vyhodnocujeme `(set! start new-start)`
V prostředí $\mathcal{P}_{12}$ je změněna vazba `start`$\longrightarrow$*kontinuace* β na `start`$\longrightarrow$*kontinuace* δ.
výsledek: *nedefinovaná hodnota*

v prostředí $\mathcal{P}_{24}$ vyhodnocujeme `(return l)`
nalezena vazba `return`$\longrightarrow$*kontinuace* γ v prostředí $\mathcal{P}_{12}$.
nalezena vazba `l`$\longrightarrow$`b` v prostředí $\mathcal{P}_{23}$.
escape s hodnotou `b`

Aktivace *kontiuace* γ s hodnotou `b`

výsledek: `b`

výsledek: `b`