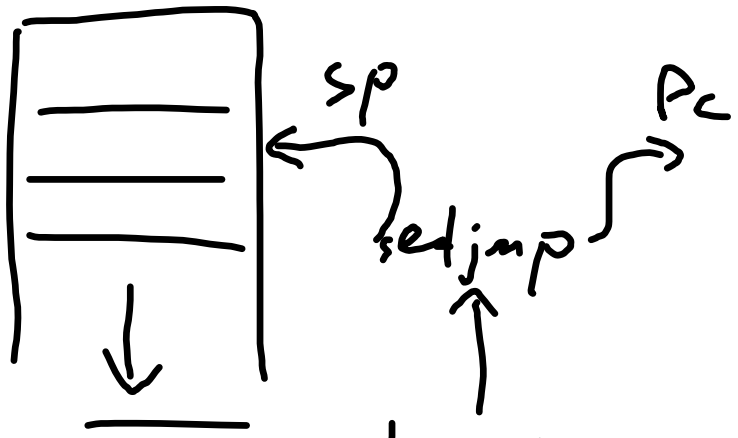
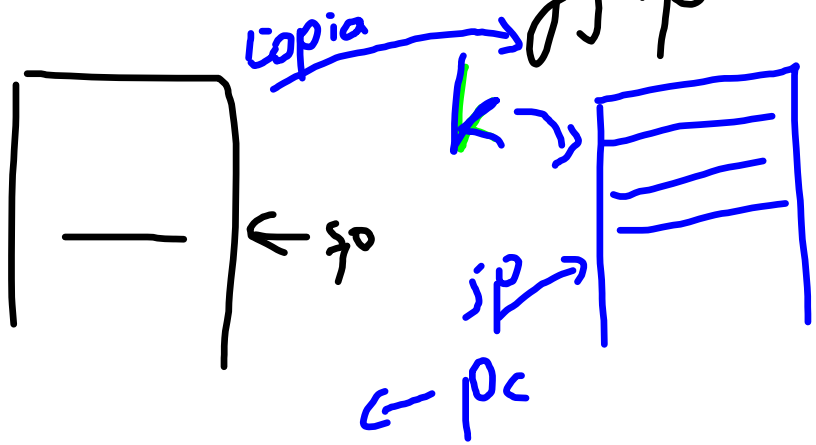
Continuaciones:

→ setjmp C:



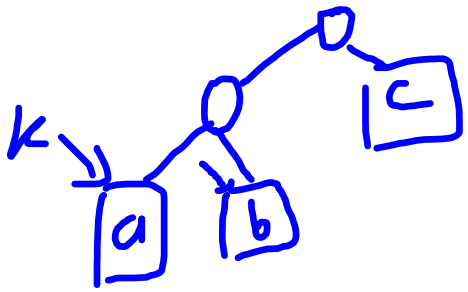
→ continuación:



↳ ej:

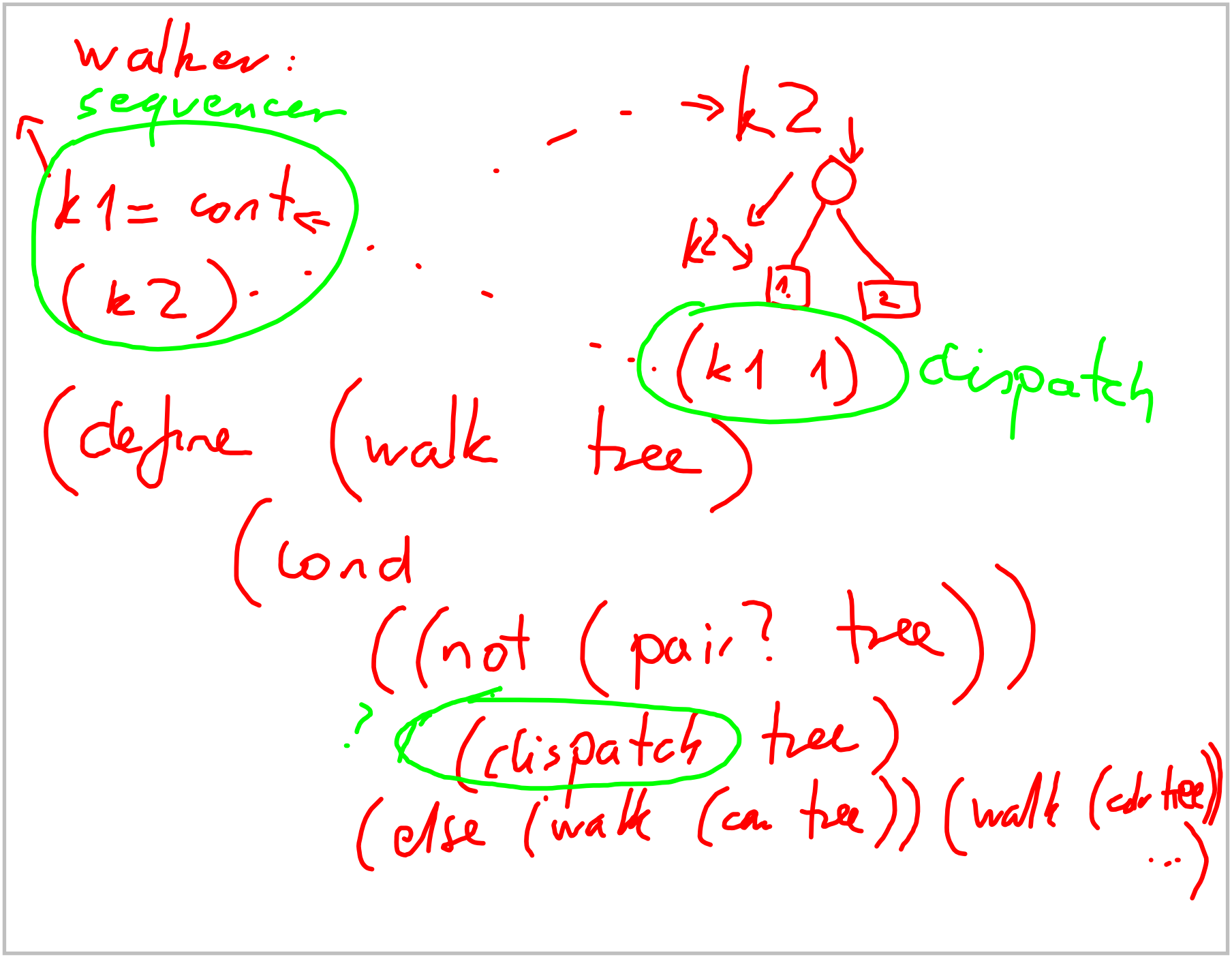
```
(define walker (make-walker
                '((a . b) . c)
                'eot ))
```

(walker) → a
(walker) → b
(walker) → c
(walker) → eot
?



walker:

1) Salvar k1 del llamador
2) Tomar k2 del walker
3) Salvar k2 en la prox hoja
4) (k1 hoja)


walker:
sequencer
k1 = cont
(k2)
k2
k2
1
2
(k1 1)
dispatch
(define (walk tree)
(cond
((not (pair? tree))
(dispatch tree))
(else (walk (car tree)) (walk (cdr tree))
...)

```
(define (dispatch leaf)
  (call/cc (lambda (k)
             (set! k2 k)
             (k1 leaf))))
```

← return si k2 es invocado

no retorna!