Paradigmata programování 2 – test1

1. Popište, jak probíhá vyhodnocení seznamu $(E_1\, E_2\, \ldots\, E_n)$, v případě, že se E_1 vyhodnotí na makro. (3 body)

2. Vyhodnoťte následující symbolický výraz a zakreslete hierarchii prostředí, která při tomto vyhodnocování vznikne. (5 bodů)

```
(let ((c 0))
  ((lambda (x y z)
     (if (null? y) (mcons x (mcons z '()))
         (mcons x (let iter ((w y))
                    (set! c (+ c 1))
                    (if (null? (mcdr w))
                        (begin
                          (set-mcdr! w (mcons z '()))
                          y)
                        (iter (mcdr w)))))))
   2
   (mcons 1 (mcons 3 ()))
   c))
```

3. Do následujícího kódu, který je implementací fronty v objektově orientovaném stylu, dopište lokální proceduru `write` pro vložení zadaného prvku na konec fronty. (6 bodů)

```
(define make-queue
  (lambda ()
    (let ((first '())
          (last '()))

      (define empty?
        (lambda ()
          (and (null? first) (null? last))))

      (define read
        (lambda ()
          (cond ((empty?) (error "Queue is empty. "))
                ((equal? first last) (let ((out (mcar first)))
                                       (set! first '())
                                       (set! last '())
                                       out))
                (else (let ((out (mcar first)))
                        (set! first (mcdr first))
                        out)))))

      (define write
        ...

      (lambda (message . args)
        (cond ((equal? message 'empty?) (empty?))
              ((equal? message 'read) (read))
              ((equal? message 'write) (write (car args)))
              (else (error "Unknown message!")))))))
```

4. V jazyce Scheme vytvořte makro `dosquares`, které na daný symbol naváže postupně druhé mocniny čísel z daného rozsahu a vyhodnotí v prostředí s touto vazbou zadané symbolické výrazy. (6 bodů)

   Příklady použití:

```
(dosquares n 5 10 (display n) (display " "))
(newline)
(let ((sum 0))
  (dosquares n 1 10 (set! sum (+ sum n)))
  sum)
```

   Výstupy příkladů:

```
25 36 49 64 81 100
385
```