`nth-element`

Exercise 1.6

If we check whether $n = 0$ before determining that *lst* is not empty, then we might take the car or cdr of an empty list in the recursive call.

Exercise 1.7

```
(define informative-report-list-too-short
  (lambda (lst n)
    (eopl:error ’nth-element
                "~s does not have ~s elements.~%"
                lst
                (+ n 1))))

(define nth-element
  (lambda (lst n)
    (define iter
      (lambda (l m)
        (if (null? l)
            (informative-report-list-too-short lst n)
            (if (zero? m)
                (car l)
                (iter (cdr l) (- m 1))))))
    (iter lst n)))
```

`remove-first`

Exercise 1.8

remove-first : $Sym \times Listof(Sym) \to Listof(Sym)$
usage: `(remove-first` *s los*`)` `returns a list with the elements of` *los* `arranged in the same order, except that all of the elements before and including the first occurrence of` *s* `are removed.`

Exercise 1.9

```
(define remove
  (lambda (s los)
    (if (null? los)
        '()
        (if (eqv? (car los) s)
            (remove s (cdr los))
            (cons (car los) (remove s (cdr los)))))))
```

`occurs-free?`