Exercise 3.38

```
;; expressions
(cond-exp
 (clauses (list-of (list-of expression?))))

;; translation-of
(cond-exp (clauses)
          (cond-exp (translate-clauses clauses senv)))

(define translate-clauses
  (lambda (clauses senv)
    (let ((predicates (map car clauses))
          (consequents (map cadr clauses)))
      (map list
           (map (lambda (predicate)
                  (translation-of predicate senv))
                predicates)
           (map (lambda (consequent)
                  (translation-of consequent senv))
                consequents)))))

;; value-of
(cond-exp (clauses)
          (value-of-cond clauses nameless-env))

(define value-of-cond
  (lambda (clauses nameless-env)
    (if (null? clauses)
        (report-no-cond-value)
        (let* ((clause (car clauses))
               (predicate (car clause))
               (consequent (cadr clause))
```

```
          (pval (value-of predicate nameless-env)))
      (if (expval->bool pval)
          (value-of consequent nameless-env)
          (value-of-cond (cdr clauses) nameless-env))))))
```