

Exercise 3.30

The value in an `extend-env` is an expressed value, so the value of `saved-val` in `apply-env` is an expressed value.

Exercise 3.31

```
(letrec-exp
  (p-name identifier?)
  (b-vars (list-of identifier?))
  (p-body expression?)
  (letrec-body expression?))

(call-exp
  (rator expression?)
  (rands (list-of expression?)))

(define-datatype environment environment?
  (empty-env)
  (extend-env
    (var identifier?)
    (val expval?)
    (env environment?))
  (extend-env-rec
    (p-name identifier?)
    (b-vars (list-of identifier?))
    (body expression?)
    (env environment?)))

(letrec-exp (p-name b-vars p-body letrec-body)
  (value-of letrec-body
    (extend-env-rec p-name b-vars p-body env)))

(call-exp (rator rands)
  (let ((proc (expval->proc (value-of rator env)))
        (args (map (lambda (rand) (value-of rand env))
                    rands))))
```

```
(apply-procedure proc args)))

(define apply-procedure
  (lambda (proc1 vals)
    (cases proc proc1
      (procedure (vars body saved-env)
        (value-of body
          (extend-env* vars
            vals
            saved-env)))))))
```