Here is the example “message passing” object-oriented code for a bank account, from SICP Exercise 3.11:

```
(define (make-account balance)
 (define (withdraw amount)
   (if (>= balance amount)
       (begin (set! balance (- balance amount))
              balance)
       "Insufficient funds"))
 (define (deposit amount)
   (set! balance (+ balance amount))
   balance)
 (define (dispatch m)
   (cond ((eq? m 'withdraw) withdraw)
         ((eq? m 'deposit) deposit)
         (else (error "Unknown request -- MAKE-ACCOUNT" m))))
 dispatch)
```

**About environment diagrams:**

*Frames* contain bindings from symbols to values.

*Procedures* are double-circle diagrams, where
- the left circle points to the procedures parameters and body, and
- the right circle points to the frame context from where the procedure was created.

**How do I apply a procedure?**

“To apply a procedure,
- create a new frame,
  and in it bind the parameters’ symbols to their associated values,
  link that frame back to the *frame from which the procedure was created*, and
  evaluate the body of the procedure from this new frame.”
Question 1. Show the environment structure generated by:

(define acc (make-account 50))

Question 2. Extend your drawing by evaluating:

((acc 'deposit) 40)

Note that this returns 90.