Exercise 3.42.

Ben Bitdiddle suggests that it's a waste of time to create a new serialized procedure in response to every `withdraw` and `deposit` message. He says that `make-account` could be changed so that the calls to `protected` are done outside the `dispatch` procedure. That is, an account would return the same serialized procedure (which was created at the same time as the account) each time it is asked for a withdrawal procedure.

```
(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)
  (let ((protected (make-serializer)))
    (let ((protected-withdraw (protected withdraw))
          (protected-deposit (protected deposit)))
      (define (dispatch m)
        (cond ((eq? m 'withdraw) protected-withdraw)
              ((eq? m 'deposit) protected-deposit)
              ((eq? m 'balance) balance)
              (else (error "Unknown request -- MAKE-ACCOUNT"
                           m))))
      dispatch)))
```

Is this a safe change to make? In particular, is there any difference in what concurrency is allowed by these two versions of `make-account`?

Answer.

This is a safe change to make. Notice our original intention of introducing serialization is to ensure that only one execution of a procedure in each serailized set is permitted to happen at a time. It puts no restriction on how a procedure is created. The `balance` of an account always stay upgraded even if it is asked to call the same serialized procedure. Hence, there is nothing different in what concurrency is allowed by these two version of `make-account`.