1.

a.
```
(define Helper (lambda (a lst)
        (if (null? lst) (list)
        (if (= a (car lst)) (car lst)
        (if (null? (cdr lst)) (list)
        (Helper a (cdr lst)))))))
    (define Find (lambda (a lst)
        (if (null? lst) (list)
        (if (null? (Helper a (car lst))) (Find a (cdr lst))
        (if (= a (FHelper a (car lst))) (car lst)
        (if (null? (cdr lst)) (list)
        (Find a (cdr lst))))))))
```

b.
```
(define Helper (lambda (l, r)                    // takes 2 lists l and r
        (if (null? l) r
        (cons (car l) (Helper (cdr l) r)))))
    (define Concatenate (lambda (lst)
        (if (null? lst) (list)
        (Helper (car lst) (Concatenate (cdr lst))))))
```

2.
```
(define IndexFind (lambda (a lst)
        (if (null? lst) (list)
        (if (= 0 a) (car lst)
        (IndexFind (cdr lst) (- a 1))))))
(define Length (lambda (lst)
        (if (null? lst) 0
        (+ 1 (Length (cdr lst))))))
(define Random (lambda (lst)
        (if (null? lst) (list)
        (IndexFind lst (- (Length lst) 1)))))
(define ShuffleHelp (lambda (a lst)
        (if (null? lst) (list)
        (if (= a (car lst)) (cdr lst)
        (cons (car lst) (ShuffleHelp a (cdr lst)))))))
(define Shuffle (lambda (lst)
        (if (null? lst) (list) (let ((a (Random lst)))
        (cons a (Shuffle (ShuffleHelp a lst)))))))
```