PS4 Mehmet Murat Budak 78940

Problem 1

```
(define (empty-list)
  (lambda (mode)
    (display "end of list")
  )
)


(define (prepend-list a lst)
  (lambda (mode)
    (if mode a lst)
  )
)


(define (car-list lst)
  (lst #t)
)
(define (cdr-list lst)
  (lst #f)
)
```

Problem 2

Iterative Implementation

```
(define (find-min-helper lst num)
  (if (pair? lst)
    (find-min-helper (cdr lst) (min num (car lst)))
    num ;If not a pair return num
  )
)

(define (find-min lst)
  (if (pair? lst)
    (find-min-helper (cdr lst) (car lst))
    #f
  )
)
```

Problem 3

```
(define (remove-n-times elem lst num)
  (cond ((null? lst) '())
        ((eq? (car lst) elem)
         (if (= num 0)
             lst
             (remove-n-times elem (cdr lst) (- num 1))))
        (else (cons (car lst) (remove-n-times elem (cdr lst) num)))))
```

Problem 4

```
(define count-occurrence-nested
    (lambda (lst char)
        (if (eq? lst '())
            0
            (if (list? (car lst))
                (+ (count-occurrence-nested (cdr lst) char) (count-occurrence-nested (car ls
                (if (eq? (car lst) char)
                    (+ 1 (count-occurrence-nested (cdr lst) char))
                    (count-occurrence-nested (cdr lst) char)
                )
            )
        )
    )
)
```