MC28 Intra-term Exam 2

Serial #:

This exam is an open-book, open-notes, individual-work exam. Please write your name only on this page. Do not turn the page until instructed, in order that everyone may have the same time. Then, be sure to look at all problems before deciding which one to do first. Some problems are easier than others, so plan your time accordingly. You have 50 minutes to work.

Write the answer to each problem on the page on which that problem appears. You may use the back of that same sheet if you need additional space. You may also request additional paper.

Name: _____

Problem	Page	Possible	Score
1	2	12	
2	3	12	
3	4	12	
Total		36	

1. **[12 Points]** Consider the problem of deleting all but the first element from a queue, or alternatively the problem of deleting all but the last element. Both can be done using only the operations of the queue ADT, but would take time proportional to the number of elements in the queue, since the elements would have to be deleted one by one and then the single element that is to survive reinserted.

   Alternatively, constant time implementations of these two operations are possible, provided we take advantage of the specific representation of queues rather than using only the ADT operations. Write constant-time versions of `delete-all-but-first!` and `delete-all-but-last!` that work with the book's queues (pp. 208-213). You may assume that the queue being operated on is non-empty in each case.

2. **[12 Points]** Normal Gack-land people can only `go` in directions in which there are exits, such as `north`, `south`, `east`, `west`, `up`, and `down`. Create a new subclass of `person`, `homing-person`, that can always `go` in one additional "direction," namely `home`, regardless of where it is. That is, if `homer` is a `homing-person` and we `(ask homer 'go 'home)`, he will move to the place where he was created. When `ask`ed to `go` in any other direction, he responds the same way as other people. Here's an example:

```
(define homer (make-homing-person 'homer computer-lab 10)) ;homer is a nerd

(install-person homer)

(ask homer 'go 'east)
homer moves from computer-lab to lounge

(ask homer 'go 'south)
homer moves from lounge to offices
At offices : homer says -- Hi barbara

; oh no! the witch!

(ask homer 'go 'home)
homer moves from offices to computer-lab
At computer-lab : homer says -- Hi karl
```

Your job is to write `make-homing-person`.

3. [**12 Points**] The following procedure is designed to eliminate an entire time segment (with all its actions) from an agenda. Fill in the blanks appropriately.

```
(define delete-time-from-agenda!
  (lambda (time agenda)
    (if (= time (current-time agenda))
        (error "deletion of current time not allowed")
        (delete-time-from-segments! time (segments agenda)))))

(define delete-time-from-segments!
  (lambda (time segments)
    ;; assumes that segments is non-empty, and that
    ;; time isn't equal to the first segment's time
    (if (< time (segment-time (car segments)))
        (error "requested time not in agenda")
        (let ((rest (cdr segments)))
          (if (null? rest)
              (error "requested time not in agenda")

              (if _____________________________________

                  _____________________________________
                  (delete-time-from-segments! time rest)))))))
```