Problem Set 4
COMP301 FALL 2023
Week 4: 27.10.2023 - 30.10.2023

Instructions:

- Submit your answers to the Blackboard PS4 assignment until October 28 Saturday, at 23.59.
- Please submit only **one single PDF file**, where all of your codes for each of the parts are included.
- Name your submission file as *id_ username_ ps4.pdf*
  (Example: *00000_ edemirbas17_ ps4.pdf*).

Problem 1: In the lecture you have seen a procedural implementation of the environment. Below, there is a procedural implementation of the list. Fill in the blank part.

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

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

(define (car-list lst)
  (lst #t))

(define (cdr-list lst)
  (lst #f))

; Tests:
(define x
  (prepend-list 13
    (prepend-list 3
      (prepend-list 6
        (prepend-list 7
          (empty-list))))))

(car-list x) ; returns 13
(car-list (cdr-list x)) ; returns 3
(car-list
  (cdr-list
    (cdr-list
      (cdr-list (cdr-list x))))) ; returns "end of list"
```

Problem 2:

a) find-min. Given a non-nested list, implement a procedure "find-min" that returns the minimum element of the list. If the list is empty, the procedure returns `#f`.

```
(find-min '(1 2 3 4 98 9)) ; returns 1.
(find-min '(-1010 -3 -45 -67 -97)) ; returns -1010.
```

***b) remove-n-times**.* Given a list, an element and a number, implement a procedure "remove-n-times" that removes the element from the list n times. If the element does not occurs n times in the list, it removes all occurrences.

```
(remove-n-times 'a '(a b a a b a) 2) ; returns (b a b a).
(remove-n-times 'a '(a b a) 3) ; returns (b).
```

Problem 3: Given a nested list and an input, implement a procedure named "count-occurrence-nested" that counts the occurrence of the given element in the nested list.

```
(count-occurrence-nested '(a b (a b (a b)) (a b)) 'a) ; returns 4
(count-occurrence-nested '(a (b a) (a b) (a b c)) 'b) ; returns 3
```