Clicker Questions
September 12
Is this function tail-recursive?

(define prod (lambda (vec acc)
  (if (null? vec) acc (* (car vec) (prod (cdr vec) acc))))))

A. Yes because it uses an accumulator
B. Yes because the last thing it does is to recurse
C. No because it doesn't recurse on a tail
D. No because it modifies the result of its recursive call
Answer D: It is not tail recursive because it modifies the result of its recursive call:

\[ (*) (\text{car vec}) (\text{prod (cdr vec) acc}) \]

recursion
(define Max (lambda (lat)
  (if (null? (cdr lat))
      (car lat)
      (let ([M (Max (cdr lat))])
        (if (< (car lat) M)
            M
            (car lat)))))))

A. No because it doesn't use an accumulator.
B. Yes because it either returns M or (car lat); it isn't recursive.
C. No because it tests the value of M against (car lat).
D. Yes because it either returns M or (car lat); it doesn't modify the result of the recursion
Answer C: No because it tests the value of M against (car lat). A tail-recursive function can't do anything with the result of a recursive call but return it immediately.
If L is the list '(1 3 (5 7) (2 4 (6))) what is (cons (car L) (cdr L))? 

A. An error. 
B. L 
C. (1 3 5 7 2 4 6) 
D. (1 3 5 7 (2 4 (6)))
Answer B: For any non-null list \((\text{cons} \ (\text{car} \ L) \ (\text{cdr} \ L))\) is \(L\) itself.