Directions: There are 7 numbered problems worth 14 points each; you get 2 points for free. You don’t need to write helper functions via letrecs; you can define them at the top level (and that might make your code easier to read). You can assume there is a predicate (atom? x; you don’t need to write that. Any other helper functions you should write yourself. In the following questions argument lat is always a flat list, such as ‘(a b c) and argument L is a general list, such as ‘( a (b c (d e)) f). There is a place on the last page for you to sign the Honor Pledge.

1. Write procedure (get n lat) which returns the element of lat at index n. If n is greater than or equal to the length of lat, get should return the last element of lat. So (get 0 ‘(a b c d e)) returns a, (get 3 ‘(a b c d e)) returns d, and (get 100 ‘(a b c d e)) returns e.
2. Use foldr or foldl to write \( \text{(replace old new lat)} \) which replaces each instance of atom \( old \) with atom \( new \) in \( lat \), a flat list of atoms.
3. Write any way you wish procedure (replace* old new L) which replaces each instance of old with new in the general list L.
4. Write (Alternates lat) which returns a pair of lists, the first with the even-indexed elements of lat and the second with the odd-indexed elements. The two returned lists should have elements in the same order as the original lat. For example, (Alternates `(a b c d e f g)) returns `((a c e g) (b d f)).
5. Use map and apply to write \((\text{Count a L})\) which returns the number of times atom \(a\) occurs in general list \(L\).
6. Let’s say that a *count list* for a list L is a list of pairs, where the first element of each pair is one of the atoms of L and the second element of the pair is how often that atom occurs in L. So a count list for ‘(a b a c d d a) is ( (a 3) (b 1) (c 1) (d 2) ). Write function (CountList lat) that returns a count list for flat list lat. Your solution can list the elements of lat in any order you wish.
7. Explain step-by-step how the following expression will be evaluated in the top-level environment.

\[
( \text{let } ([a \ 5]) \ (\text{lambda } (x) \ (+ \ x \ a))) \ 6)
\]

Note that it consists of a let expression:

\[
(\text{let } ([a \ 5])
\text{\ 
(\text{lambda } (x) \ (+ \ x \ a)))}
\]

and a call to value of this let expression with argument 6. In particular you should say what the let expression evaluates to, and how this value is called with argument 6.

You can use this page as extra room for any problem. If you want me to grade something here be clear about which problem it refers to.
Please write and sign the honor pledge when you are finished with the exam.