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. You can assume there are predicates (atom? x) and (number? x); you don’t need to write those. You can also assume there is a function (length L) that returns the number of entries at the top level of list L. So (length '(a b c)) is 3 and (length '(a (b c))) is 2. Any other helper functions you should write yourself.

1) Write procedure (insert n a lat) that inserts atom a at position n in lat and returns the resulting list. If n is greater than or equal to the length of lat, insert atom a at the end of lat. So (insert 0 'x '(a b c d e)) returns (x a b c d e), (insert 3 'x '(a b c d e)) returns (a b c x d e), and (insert 25 'x '(a b c d e)) returns (a b c d e x)
2) Write procedure \textbf{(replace a b L)} that replaces every instance of atom \textit{a} with atom \textit{b} in the general list \textit{L}. So \textbf{(replace ‘a ‘x ‘(a b (c (a d) a (b c))}) returns \textbf{(x b (c (x d) x (b c))})
3) What will this expression evaluate to? If you are sure your answer is right that is all I need, but if your answer is wrong it would help to have an explanation.

(let ([a 3] [b 5])
  (let ([f (lambda (x) (+ x a))])
    (let ([a 10] [b 20])
      (f b))))
4) Explain step-by-step how the following expression will be evaluated in the top-level environment.

(let ([s 7] [f (lambda (x y) (+ x y))])
  (f 2 s))
5) This problem concerns a data structure \textit{Grades} that represents the grades (numbers between 0 and 100) that students receive on an exam or assignment. The constructor for this structure is
\begin{verbatim}
(define MakeGrades (lambda (list-of-students list-of-grades)
    (list 'grades list-of-students list-of-grades)))
\end{verbatim}
For example \((\text{MakeGrades} \ (\text{harry ron hermione}) \ (90 65 100))\) returns the triple \((\text{'grades 'harry ron hermione '90 65 100})\)

a) Give a function \((\text{LookupGrade(person g})\) which returns the grade saved for the given person in a grades-structure formed by the constructor above. If the person is not found in this structure you should return 0. For example, if we say
\begin{verbatim}
(define G1 (MakeGrades '(harry ron hermione) '(90 65 100)))
\end{verbatim}
\((\text{LookupGrade 'ron G1})\) returns 65 and \((\text{LookupGrade 'nevville G1})\) returns 0.

b) Give a function \((\text{AverageGrade person g1 g2 g3 ... gn})\) that takes a name and any number of grades-structures and returns the person’s average grade. Note that there is a standard function \((\text{length L})\) that gives the number of entries of list \(L\); you can use this to find the number of grades for the average. If we say
\begin{verbatim}
(define G1 (MakeGrades '(harry ron hermione) '(90 65 100)))
(define G2 (MakeGrades '(harry ron hermione) '(95 61 100)))
\end{verbatim}
then \((\text{AverageGrade 'ron G1 G2})\) returns 63.
6) Write a procedure \textbf{(Sum lat)}. As you might expect, this returns the sum of the numbers in \textit{lat}. However, if \textit{lat} contains the atom ‘squares’ \textit{(Sum} \textit{lat)} returns the sum of the squares of the numbers in \textit{lat}. For example \textit{(Sum \{1 2 3}\}) returns 10, and \textit{(Sum \{1 2 3 squares} 4\}) returns 30. You can assume that any entry in \textit{lat} other than the atom ‘squares will be a number.
7) Use foldl or foldr to write a function \((\text{minMax lat})\) that takes a flat list of numbers and returns a pair containing the smallest and largest values in that list. For example \((\text{minMax '(5 2 8 6 3 6 1)})\) returns \((1 8)\). You can assume all of the numbers in \(\text{lat}\) are between 0 and 100.
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.