CSE 326 Problem Set 1 Due: October 13, 1997

1. The classic way to evaluate a polynomial is Horner's Rule. Horner's Rule can be stated recursively as follows:

   Let $p(x) = a_0 + a_1 * x + a_2 * x^2 + \cdots + a_k * x^k$ be a polynomial, where $*$ means "times." To evaluate the polynomial $p(x)$ at c, first let y be the result of evaluating the polynomial $a_1 + a_2 * x + a_3 * x^2 + \cdots + a_k * x^{k-1}$ at c, then evaluate the final result as $a_0 + c * y$.

   - Briefly argue that the process described above correctly evaluates the polynomial $p(x)$ at c.
   - Give a recursive pseudocode function which executes Horner's Rule. Assume that the coefficients of the polynomial are stored in a linked list with a_0 stored at the head of the list. Hint: your function should have arguments p and c where p is a pointer to a list of coefficients and c is the value at which the polynomial is to be evaluated.
   - Design an iterative algorithm and its pseudocode for Horner's Rule. Your iterative algorithm should be a simple for loop. In this case the polynomial should be stored in an array A with $A[1] = a_0, A[2] = a_1$, and so on.
   - What difficulty, if any, is there in executing Horner's Rule iteratively when the coefficients are stored in a linked list?

2. In this problem you should carefully define a list oriented recursive mergesort algorithm. Assume the input data is found in an unsorted linked list. Mergesort should return (destructively) a sorted list. Mergesort first splits the list into two lists of approximately equal size, recursively mergesorts the two lists, then merges the two sorted lists into a single sorted list. You can assume you have already defined a *split* function that returns a *duo* whose *first* field points to a list containing half the elements of the given list, and whose *second* field points to a list containing the remaining elements.

   - First, carefully define *merge*, which takes two sorted lists as arguments and combines them into a single sorted list.
   - Second, define *mergesort* which takes an unsorted list as argument and returns a sorted list containing the same elements.

   Try to make your pseudocode as simple and elegant as possible.