Assignment 2, Fall 2017

Due Monday Nov 6, 2017

(execute on Racket Scheme download, test before submitting)

1. **make5** - takes two integers, and returns a 5-digit integer constructed of the rightmost 3 digits of the first input, and the leftmost 2 digits of the second input. For example, (make5 561432 254) would return 43225. Negative signs on either input number should be ignored - that is, (make5 561432 -254) would also return 43225. If the first number has less than three, and/or the second number has less two digits, your function should return -2. Note: you may want to define some auxiliary functions.

2. **concatL** – takes two lists of strings of the same length and returns a list of that length containing strings which are the concatenation of the strings at the same position in the two list. For example, (XXX '("ab" "c" "de") '("fff" "des" "vvvv") will return ("abfff" "cdes" "devvvv"). Note: you may want to define some auxiliary functions. You may also use the built-in function "string-append" which takes two strings as arguments and returns a string which is the concatenation of the two string arguments.

3. **buildList** – Takes an integer N and a Scheme expression E (i.e. an atome or a list) and returns a new list of length N where each element is the Expression E. For example:

   (buildList 5 '()) will return the list (( ) ( ) ( ) ( ) ( ))

   (buildList 3 'A) will return (A A A)

   (buildList 2 '(a b c)) will return ((a b c) (a b c))

4. **listpicket** – takes a list and a "picket" and returns a "picketed list." The picketed list is the original list where all the elements are now surrounded by pickets (there will be only one picket between any two elements). For example:

   (listpicket 'A '(d (e f g) h (g)) will return:
   (A d A (e f g) A h A (g) A)

   Note that the sub-lists are not affected.

5. **listpicketall** – after you have done listpicket, expand it so that the pickets are also inserted in the sub-lists. For example:

    (listpicketall 'A '(a (((b c)) e))) will return
    (A a A (A (A (A b A c A) A) A e A) A)

6. **selectN -** takes as input an integer N. It then builds and returns a "select" function based on N. The "select" function that is produced (by your function selectN) would have the property that it takes as input a list, and returns the same list with the first N elements removed. For example, if selectN was called as follows:

    (selectN 3)

    a function would be produced that takes as input a list and returns the list with the first 3 elements removed from that list.
    For example, if the original call had been made as follows:

    (define Last (selectN 3))

    then the produced function C would behave as follows:

    (Last '(4 8 2 9 -1 13)) *** would return (9 -1 13)
    (Last '(-2 3 -4 8 9 1 7)) *** would return (8 9 1 7)

    Your task is just to write selectN.
    Of course, selectN should work for ANY input integer, not just 3.