Question 1: Eyes in the Dark

Use recursion to write a function \( (\text{double } L) \) that consumes a \( (\text{listof Any}) \) and returns a list containing two copies of each item from \( L \).

For example,

\[
\text{(double (list ." "o" "O"))} \Rightarrow \text{(list ." ." ."O" ."O" )}
\]

Question 2: What’s That Function?

Rewrite the following \( (\text{unknown-fn1 } L) \) using recursion.

\[
\text{(define (unknown-fn1 } L) \\
\text{(foldr string-append "")} \\
\text{(map (lambda (x) } \\
\text{(substring x (- (string-length x) 1))))} \\
\text{(filter string? } L))))
\]

Question 3: Insertion Sort

Use recursion to write a function \( (\text{insert-strlen item } L) \) that consumes a \( \text{Str} \) and a \( (\text{listof Str}) \). If \( L \) is already sorted by increasing string length, the function returns list containing all the values in \( L \), with \( \text{item} \) inserted so it remains sorted in this way.

Strings of equal length should be in alphabetic order.

For example,

\[
(\text{insert-strlen "my" (list a" by" "tr" "huh")}) \Rightarrow (\text{list a" by" my" tr" huh")}
\]

Using insert-vowel as a helper function, use recursion to write a function \( (\text{sort-length } S) \) that consumes a \( (\text{listof Str}) \) and returns a \( (\text{listof Str}) \) containing the same values as \( S \), but in increasing order of string length (and alphabetic order when of equal length).

For example:

\[
(\text{sort-length (list z" aa" hhh" gg" dddd")}) \Rightarrow (\text{list z" aa" gg" hhh" dddd")}
\]

Do not use sort, instead implement insertion sort yourself.

Question 4: Stairs
Exercise

Use recursion to write a function (make-stairs n) that returns a (listof (listof Nat)) of length n, where the i\text{th} value consists of i copies of i.

For example,

(make-stairs 5) =>
(list
(list 1)
(list 2 2)
(list 3 3 3)
(list 4 4 4 4)
(list 5 5 5 5 5))

Hint

Start by writing a function to create one row.