CS115 – Lab 9: Advanced Recursion

Spring 2020

Question 1: Eyes in the Dark

Exercise

Use recursion to write a function `(double L)` that consumes a `(listof Any)` and returns a list containing two copies of each item from `L`.
For example, `(double (list "." "o" "0")) => (list "." "." "o" "o" "0" "0")`

Question 2: What's That Function?

Exercise

Rewrite the following `(unknown-fn1 L)` using recursion.

```
(define (unknown-fn1 L)
  (foldr string-append ""
         (map (lambda(x)
                (substring x (- (string-length x) 1)))
              (filter string? L))))
```

Question 3: Insertion Sort

Exercise

Use recursion to write a function `(insert-strlen item L)` that consumes a `Str` and a `(listof Str)`. The function requires that `L` is already sorted by increasing string length then alphabetic order.
The function returns list containing all the values in `L`, with `item` inserted so it remains sorted in this way.
For example,
`(insert-strlen "my" (list "a" "by" "tr" "huh")) => (list "a" "by" "my" "tr" "huh")`

Exercise

Using `insert-strlen` as a helper function, use recursion to write a function `(sort-length S)` that consumes a `(listof Str)` and returns a `(listof Str)` containing the same values as `S`, but in increasing order of string length (and alphabetic order when of equal length).
For example: `(sort-length (list "z" "aa" "hhh" "gg" "dddd")) => (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 ith 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.