CS115 – Lab 7: Recursion

Spring 2020

Question 1: Locating a Value

Exercise

Write a function `(list-pos L item)` that consumes a `(listof Str)` and a `Str` and returns the position of the first occurrence of `item` in `L`.

`L` is non-empty and `item` is guaranteed to be in it. The first item is in position 0.

For example: `(list-pos (list "Odlaw" "Wenda" "Waldo" "Woof") "Waldo") => 2`

! Use recursion! Do not use **`map`**, **`foldr`**, or **`filter`**.

Question 2: Copying Strings

Exercise

Write a function `(copy s n)` that returns the `Str` created by appending `n` copies of `s`.
For example: `(copy "that" 3) => "thatthatthat"`

! Use recursion! Do not use **`map`**, **`foldr`**, or **`filter`**.

Question 3: Building Stairs

Exercise

Write a function `(stair n)` that returns a `(listof Str)` of length `n` where the first item is `"X"`, and every subsequent item contains one more `"X"`.
For example,

```
(check-expect (stair 4)
              (list "X"
                    "XX"
                    "XXX"
                    "XXXX"))
```

Hint You may use `copy` as a helper function.

! Use recursion! Do not use **`map`**, **`foldr`**, **`filter`**, or **`range`**.

Question 4: Hide & Seek

Exercise

Create a function `(next-list L target)` that consumes a `(listof Any)` and an `Any`, and returns the item in the list that appears after `target`.
If `target` is not present in `L` or is the last item in `L`, the function returns `#false`.
For example: `(next-list (list 1 2 3) 2) => 3` `(next-list (list 1 2 3) 7) => #false`
`(next-list (list 1 2 3) 3) => #false`
If `target` appears more than once in `L`, consider the first occurrence.
`(next-list (list 2 45 7 8 7 3 6 7 9) 7) => 8`

! Use recursion! Do not use `map`, `foldr`, or `filter`.