CS135 Tutorial 05

Goals

- Consume and alter a sorted list of numbers (`remove-one-of-each & dedup`).
- Look at part of an assignment from previous term (`longest-sequential-sequence`).
- Appropriate tests.

remove-one-of-each

Write a function `remove-one-of-each` that consumes a sorted list of Num and produces a sorted list of Num where one occurrence of each Num found in the consumed list of Num has been removed.

For example:

```
(check-expect (remove-one-of-each
(list 2 2 3 3 3 3 4 5 6 7 7 7))
(list 2 3 3 3 7 7))
```

remove-one-of-each: Purpose & Contract

```
;; (remove-one-of-each slon) produces [slon] where one occurrence of each of
the numbers has been removed

;; remove-one-of-each: (listof Num) -> (listof Num)
;; requires: The input list of Num is sorted
```

remove-one-of-each: Examples

```
;; (remove-one-of-each slon) produces a list of …

;; Examples:
(check-expect (remove-one-of-each empty) empty) ;; the most basic case
(check-expect (remove-one-of-each (list 3)) empty) ;; one element
(check-expect (remove-one-of-each (list 3 3 3 3)) (list 3 3 3)) ;; one
element repeated
(check-expect (remove-one-of-each (list 3 4 5 6)) empty) ;; one of each
element
(check-expect (remove-one-of-each (list 3 3 4 5 6 6)) (list 3 6))

;; remove-one-of-each: (listof Num) ->…
```

dedup

Write a function dedup that consumes a sorted list of numbers and produces the same list with each element appearing only once.

For example:

```
(check-expect (remove-one-of-each
(list 2 2 3 3 3 3 4 5 6 7 7 7))
(list 2 3 4 5 6 7 ))
```

Hint: This is very similar to `remove-one-of-each`

Longest sequential substring

Write a function, `longest-seq-substr`, that consumes a `(listof Nat)` and produces the longest substring of the list where all the elements are in sequential order.

Definitions:

- sequence: an ordered list `(list 1 2 3 40 5 62)`
- subsequence: what's left of a sequence after deleting elements; order is maintained. `(list 2 40 5)`
- substring: a subsequence where all members were consecutive in the original sequence `(list 2 3 40)`

Example:

```
(check-expect (longest-seq-substr (list 10 11 1 2 3 100))
              (list 1 2 3))
```

Longest sequential substring

Write a function, `longest-seq-substr`, that consumes a `(listof Nat)` and produces the longest substring of the list where all the elements are in sequential order.

Examples:

```
(check-expect (longest-seq-substr empty) empty)          ; empty
(check-expect (longest-seq-substr (list 10)) (list 10))   ; just one
(check-expect (longest-seq-substr (list 1 2 3 10 11 100)) ; longest at start
              (list 1 2 3))
(check-expect (longest-seq-substr (list 10 11 1 2 3))     ; longest at end
              (list 1 2 3))
(check-expect (longest-seq-substr (list 10 11 1 2 3 100)) ; longest in middle
              (list 1 2 3))
```