CMSC 15100 Introduction to Computer Science I

Homework 10

Your answers should be saved in a .rkt file and submitted via Canvas before the start of the next lecture. At the top of the file for this assignment and all future assignments include a comment,

```
;; Homework 10
;;
;; your name
;; your CNET id
;; your collaborators
```

For this homework, some parts require you to explain things in English. For these questions, write your answers in a Racket comment box, which you can create by clicking on the "Insert" tab at the top of DrRacket.

If your code doesn't compile, it may not be graded. Please double check to make sure your code runs before submitting!

Problems

1. (10 points) Nearly every function we've seen so far on lists and vectors produces a new list or vector, but leaves the original list or vector unmodified. For example, after running this code

   ```
   (: vec : (Vectorof Integer))
   (define vec (vector 1 2 3 4))
   (vector-map (lambda ([x : Integer]) (+ x 1)) vec)
   ```

   the third line will output `'# (2 3 4 5)`, but the variable `vec` will still hold the same values `'# (1 2 3 4)` if we use it in any later lines of code. Using imperative programming, we can truly modify the variable `vec`.

   Make a function `add-one!` which adds one to every element of a `(Vectorof Integer)`. The type of this function should be `(Vectorof Integer) -> Void`. To write this function, you should make a recursive helper function `add-one-helper!` which takes as input a `(Vectorof Integer)` and a `Integer` index, and adds one to all indices of the vector past the given index. The type of the helper function should be `(Vectorof Integer) Integer -> Void`.

   What is the runtime of your code? Explain.