CSCC24 2025 Summer Assignment 2

Due: July 3 11:59PM
This assignment is worth 10% of the course grade.

In this assignment, we investigate some properties of lazy and/or self-referencing data structures.

This assignment is mainly theory and calculations. Please hand in your answer in a text file A2.txt.

"Scan from Left"

The standard library has a function `scanl`. For concreteness, we work on the following example (it's `scanl (+)` expanded):

```
sums acc xs = acc : go acc xs
  where
    go acc []     = []
    go acc (x:xs) = sums (acc + x) xs
```

(a) What does it do? [2 marks]

Show some algebra steps to show why `sums 0 [x,y,z]` is $[0, 0+x, 0+x+y, 0+x+y+z]$. This part just requires basic algebra; evaluation order does not matter. The purpose is just to get the hang of what the function is about.

(b) The nth Element [10 marks]

The following function gives the nth item (base-0 indexing) of a list, assuming it's long enough.

```
get 0 (x:_) = x
get n (_:xs) = get (n-1) xs
```

Show the lazy evaluation steps of `get 2 (sums 0 (10:20:[]))` until you get the numeric answer.

And in general, how much space (up to Θ) does it take to evaluate `get n (sums 0 xs)` (if `xs` is long enough)?

(c) Saving Space [10 marks]

The standard library provides a variation `scanl'` that uses `seq` to save space. Here is what `scanl' (+)` expands to:

```
sums' acc xs = seq acc (acc : go acc xs)
  where
    go acc []     = []
    go acc (x:xs) = sums' (acc + x) xs
```

Show the lazy evaluation steps of `get 2 (sums' 0 (10:20:[]))` until you get the numeric answer.

And in general, how much space (up to Θ) does it take to evaluate `get n (sums' 0 xs)` (if `xs` is long enough)?

(d) [5 marks]

You may wonder why `sums` was not implemented in the following obvious way, eliminating the `go` helper:

```
sumsSimp acc [] = acc : []
sumsSimp acc (x:xs) = acc : sumsSimp (acc + x) xs
```

There is a small difference, and it can be shown by the following example. Use the method of successive approximations to calculate and guess the values of xs and ys.

```
xs = sumsSimp 1 xs
ys = sums 1 ys
```