CSE 331
Software Design & Implementation

Summer 2024

Section Cipher – Functional Programming II

Administrivia

- HW Cipher released Thursday evening, due the following Wednesday at **11pm**
  - Please! Start early and be prepared for a challenge!
  - Give yourself time to come to OH and ask questions on Ed
  - Working on the same issue for hours when you're stuck won't help, ask for help!

- Remember to check the autograder to make sure you pass your own tests!
- Remember to look at your feedback, you may not have lost points on a question, but could still have helpful comments

Defining Function By Cases – Review

- Sometimes we want to define functions by cases
  - **Ex:** define $f(\mathrm{n})$ where $\mathrm{n} : \mathbb{Z}$

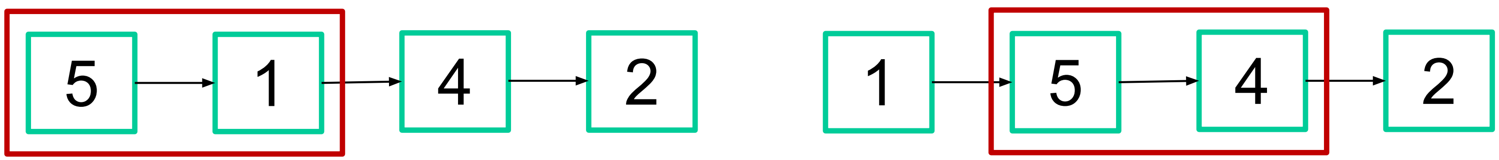
$$
\begin{array}{lll}
\textbf{func}\ \mathrm{f(n)} & := 2\mathrm{n} + 1 & \quad \textbf{if } \mathrm{n} \geq 0 \\
\mathrm{f(n)} & := 0 & \quad \textbf{if } \mathrm{n} < 0
\end{array}
$$

  - To use the definition $f(\mathrm{n})$, we need to know if $\mathrm{n} > 0$ or not
  - This new code structure requires a new proof structure

    (We will review proof by cases later in Q4)

Question 1

pseudo-sort: takes a list of numbers as an argument, “looks at the first two numbers in the list, orders the pair to place the smaller of the two in the front, and then continues through the following pairs in the list after the first element“



5 > 1, switch

Now compare 5 & 4 and so on…

(a) Write a formal definition using recursion

(b) Show by example that pseudo-sort does not actually sort the list

Structural Induction – Review

- Let **P(S)** be the claim
- To Prove P(S) holds for any list S, we need to prove two implications: base case and inductive case
  - **Base Case:** prove P(nil)
    - Use any know facts and definitions
  - **Inductive Hypothesis:** assume **P(L)** is true for a L: List
    - Use this in the inductive step ONLY ⤵️
  - **Inductive Case:** prove **P(cons(x, L))** for any x : *Z,* L : List
    - Direct proof
    - Use know facts and definitions and Inductive Hypothesis
- Assuming we know P(S), if we prove P(cons(x, L)), we then prove recursively that P(S) holds for any List

Structural Induction - 331 Format

The following is the structural induction format we recommend for using in your homework (the staff solution also follows this format)

1) **Introduction** - define P(L) to be what we are trying to prove
2) **Base Case** - show P(nil) holds
3) **Inductive Hypothesis** - assume P(L) is true for an arbitrary list
4) **Inductive Step** - show P(cons(a, L)) holds
5) **Conclusion** - “We have shown that P(L) holds for any list”

Note: You do not have to follow this format but your solution **MUST** include all the information above

Question 2

```
const s = sum(L);
...
return 2 * s;  // = sum(twice(L))
```

Prove this code is correct by showing that **sum(twice(S)) = 2 sum(S)** holds for any list S by structural induction.

$$
\begin{aligned}
&\textbf{func } \mathsf{sum}(\mathsf{nil}) &&:= 0 \\
&\qquad \mathsf{sum}(\mathsf{cons}(a, L)) &&:= a + \mathsf{sum}(L) \qquad \text{for any } a : \mathbb{Z} \text{ and } L : \mathsf{List}
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{func } \mathsf{twice}(\mathsf{nil}) &&:= \mathsf{nil} \\
&\qquad \mathsf{twice}(\mathsf{cons}(a, L)) &&:= \mathsf{cons}(2a, \mathsf{twice}(L)) \qquad \text{for any } a : \mathbb{Z} \text{ and } L : \mathsf{List}
\end{aligned}
$$

Question 3

$$\text{sum}(\text{twice-evens}(L)) + \text{sum}(\text{twice-odds}(L)) = 3\,\text{sum}(L)$$

Prove that this holds for any list S by structural induction.

$$
\begin{aligned}
\textbf{func}\ & \text{twice-evens}(\text{nil}) && := \text{nil} && \\
& \text{twice-evens}(\text{cons}(a, \text{nil})) && := \text{cons}(2a, \text{nil}) && \text{for any } a : \mathbb{Z} \\
& \text{twice-evens}(\text{cons}(a, \text{cons}(b, L))) && := \text{cons}(2a, \text{cons}(b, \text{twice-evens}(L))) && \text{for any } a, b : \mathbb{Z} \text{ and } L : \text{List}
\end{aligned}
$$

$$
\begin{aligned}
\textbf{func}\ & \text{twice-odds}(\text{nil}) && := \text{nil} && \\
& \text{twice-odds}(\text{cons}(a, \text{nil})) && := \text{cons}(a, \text{nil}) && \text{for any } a : \mathbb{Z} \\
& \text{twice-odds}(\text{cons}(a, \text{cons}(b, L))) && := \text{cons}(a, \text{cons}(2b, \text{twice-odds}(L))) && \text{for any } a, b : \mathbb{Z} \text{ and } L : \text{List}
\end{aligned}
$$

Proof By Cases – Review

- Split a proof into cases:
  - **Ex:** $\mathrm{a} = \mathrm{True}$ and $\mathrm{a} = \mathrm{False}$ or $\mathrm{n} >= 0$ and $\mathrm{n} < 0$
  - These cases needs to be *exhaustive*

- **Ex:** func $\mathrm{f(n)} := 2\mathrm{n} + 1$ if $\mathrm{n} \geq 0$
  $\mathrm{f(n)} := 0$ if $\mathrm{n} < 0$
  Prove that $\mathrm{f(n)} \geq \mathrm{n}$ **for any** $\mathrm{n} : \mathbb{Z}$

Case n ≥ 0:

$f(\mathrm{n})$	$= 2\mathrm{n} + 1$	def of f (since $n \geq 0$)
	$> \mathrm{n}$	since $n \geq 0$

Case n < 0:

$f(\mathrm{n})$	$= 0$	def of f (since $n < 0$)
	$\geq \mathrm{n}$	since $n < 0$

Since these 2 cases are *exhaustive*,
$f(\mathrm{n}) >= \mathrm{n}$
holds in general

Question 4

$$
\begin{aligned}
\textbf{func } \mathsf{swap}(\mathsf{nil}) &:= \mathsf{nil} \\
\mathsf{swap}(\mathsf{cons}(a, \mathsf{nil})) &:= \mathsf{cons}(a, \mathsf{nil}) && \text{for any } a : \mathbb{Z} \\
\mathsf{swap}(\mathsf{cons}(a, \mathsf{cons}(b, L))) &:= \mathsf{cons}(b, \mathsf{cons}(a, \mathsf{swap}(L))) \\
& \qquad \text{for any } a, b : \mathbb{Z} \text{ and } L : \mathsf{List}
\end{aligned}
$$

Prove by cases that swap(cons(a, L)) $\neq$ nil for any integer a : $\mathbb{Z}$ and list L.