Function Composition

CS 5010 Program Design Paradigms
“Bootcamp”
Learning Objectives

• At the end of this lesson, the student should be able to define short functions by composing existing functions.
Introduction

• In this lesson, we will learn how to use Function Composition to write function definitions.
Programs are sets of Functions

- We organize our programs as sets of *functions*.
- A function takes an argument (or arguments) and returns a result.
- The signature says what kind of data the argument and result are.
- A purpose statement describes how the result depends on the argument.
Function Composition

• Sometimes, the answer we want can be explained as a combination of simpler computations.

• An example is the f2c function from the textbook, where the simpler computations were just arithmetic.

• Another example is the area-of-ring function from the textbook.
Function Compositions as Diagrams

• We can think of a function composition as a wiring diagram. The arguments to the function flow in, and then they are steered (piped) through some functions.

• In the next few slides, we’ll see some of the forms that this can take.

• For each example, we’ll show a wiring diagram and the corresponding Racket expression.
A function call as a wiring diagram

\[ f(u, v) \]
A simple function composition

\[ (\text{define } (\text{my-fcn } u \ v) \ (g \ (f \ u \ v))) \]
Another function composition

\[(\text{define } (\text{my-fn } x \ y) \ (f \ (g \ x) \ (h \ y)))\]
This is the pattern we used in `area-of-ring`.

(define (area-of-ring inner outer)
  (- (area-of-circle outer)
      (area-of-circle inner)))
You can pipe a single argument through two functions.

\[
\text{(define (my-fcn x))}
\text{ (f (g x) (h x))}
\]
Simple conditionals are also permitted as part of a function composition.

\[(\text{if} \ (p \ x) \ (g \ x) \ (h \ x))\]
A function composition can take any wiring diagram as its form, *so long as the wiring diagram is small*.

\[(f \ (g \ x) \ y)\]
Definition of a function composition

\[ fc ::= \text{variable} \]

\[ ::= (\text{function} \ fc \ fc \ fc \ \ldots) \]

\[ ::= (\text{if} \ (\text{pred} \ \text{var} \ \text{var} \ \text{var} \ \ldots) \]

\[ fc \]

\[ fc \]

But: no conditionals here.
Keep it short!

• Function composition is for very short definitions only.
• If you have complicated junk in your function, you must have put it there for a reason. Turn it into a separate function so you can explain it and test it.
Here’s a pair of examples. The left one contains all the code in the program while the right one separates the code into separate functions and combines them using function composition. Which do you think is clearer? Which looks easier to debug? Which would you like to have to defend in front of a TA?
When do you need to introduce help functions?

• If a function has pieces that can be given meaningful signatures and purpose statements, then break it up and use function composition.

• Then apply the design recipe to design the pieces.
Summary

• In this lesson, you’ve learned
  – How to use Function Composition to write a function definition.
  – When a function definition needs to be simplified by using help functions.
Next Steps

• If you have questions or comments about this lesson, post them on the discussion board.