CSC310 Programming Languages

Homework 4 (Due: Monday, March 25, 2024)

Answer following questions:

1. **Exercises 3.14** (p171) (Make sure to do for both **static scoping** and **dynamic scoping**)

3.14 Consider the following pseudo-code:

```
x : integer – – global
procedure set_x(n : integer)
    x := n
procedure print_x()
    write_integer(x)
procedure first()
    set_x(1)
    print_x()
procedure second()
    x : integer
    set_x(2)
    print_x()
set_x(0)
first()
print_x()
second()
print_x()
```

What does this program print if the language uses static scoping? What does it print with dynamic scoping? Why?

Static Scoping:

global x:	0 → 1 → (print #1) → (print #2) → 2 → (print #3) → (print #4)
second() x:	(never assigned a value)

Dynamic Scoping:

global x:	0 → 1 → (print #1) → (print #2) → (print #4)
second() x:	2 → (print #3)

If the program uses static scoping, then it will print "**1, 1, 2, 2**". However, if the program uses dynamic scoping, then it will print "**1, 1, 2, 1**". This is because for static scoping, the program will always check where a variable was declared using the static structure of the program text, which can be determined during compile time. For dynamic scoping, the program will always check where a variable was declared using the order in which procedures were called, with recent called procedures being checked first for their variable declarations.

So in this instance, with static scoping the function call `set_x(2)` in `procedure second` actually refers to the globally defined `x`, because of where the `set_x(n)` procedure was *defined*. This is in contrast to dynamic scoping, in which the function call to `set_x(n)` in `procedure second` actually refers to the `x` defined in procedure `second`, because that is the most recently called definition of `x`.