CS 164 Programming Languages and Compilers **Spring 2019**

Discussion Week of 4/15: Midterm 2 Review

Instructor: Paul N. Hilfinger GSIs: Nikhil Athreya, Vivant Sakore

1 Type Unification

1. Compute the types of the following functions:

```
fun
  f g x = if (g x) then (h g x) else (f g x)
and
  h i y = if (i y) then (f i y) else (h i y)
```

2. Is it possible to consider each of their types separately? In other words, just knowing that h is a function which takes two arguments and returns something, can we deduce the correct type for f and vice versa?

2 Operational Semantics

Suppose we have already defined the operational semantics of basic arithmetic expressions involving the integers, variables, and booleans as follows (respectively):

$$\frac{...}{\Gamma \vdash e_1 \;:\; n,\, \Gamma}$$

and

$$\frac{...}{\Gamma \;\vdash\; b_1 \;:\; b,\, \Gamma}$$

Γ here represents our mapping between variables and their values. We use the convention e for arithemtic expressions and b for boolean expressions. Note that as of now, expressions do not modify the state of our program. Write the operational semantics for the following types of constructs:

1. Assignment to an arithmetic expression as follows: $x := e$. Note an assignment returns void.

2. A sequence of statements s_1 and s_2 as follows: s_1; s_2. Note a sequence returns void.

3. An if statement as follows: if b then s_1 else s_2. Note an if statement must only execute its corresponding branch.

3 Runtime support for functions

1. Consider the following program:

```
  def f0 (x):
      def f1 (y):
          def f2 (z):
              return x * y * z
          def h1 (g):
              return g(3)
          print(h1(f2))
      f1(2)

if __name__ == "__main__":
    f0(5)
```

   What will the above code print? Give the stack representation of function calls.

2. What is continuation? Can it be handled by just using stack?