Principles of Programming Languages - Homework 1

Abhi Agarwal

1 Problem 1

(a)

The use of pi at line 4 is bound at which line? Line 3. A variable pi has been declared in the local scope, and so it overwrites the one declared in the global scope.

The use of pi at line 7 is bound at which line? Line 1. There's no variable pi declared within the local scope so it looks for a declaration of pi within the scope above it - the global scope (lets suppose). In the global scope it finds a variable called pi.

(b)

The use of x at line 3 is bound at which line? Line 2. The variable is passed into the function as a parameter, and is becomes visible in the local scope for line 3.

The use of x at line 6 is bound at which line? Line 5. The variable x declared on line 5 can be named anything in Scala, and is a new variable that overwrites the x declared on 2 (in the current local scope). It takes on the value of x that is passed into the function, but is a new variable.

The use of x at line 10 is bound at which line? Line 5. No other x is declared in the new scope introduced by the case statement, and so the only x visible to line 10 is on line 5.

The use of x at line 13 is bound at which line? Line 1. This function call lies in the global scope (for our purposes assume it's the global scope) - so it takes the value of x declared in that global scope, which is on line 1. There's no other x visible in the global scope, but the one declared on Line 1.

2 Problem 2

Is the body of g well-typed? Yup!

If so, give the return type of g and explain how you determined this type. The return type of g would be (Int, Int).

For this explanation, first, give the types for the names a and b.

```
Type of a is an Int. The reasoning stands from:

val (a, b) = (1, (x, 3))
    a: Int
    1: Int
    val a, b = 1, (x, 3)
    val a = 1
    val b = (x, 3)
    val _ = 1 => Int

Type of b is a Tuple of (Int, Int). The reasoning stands from:

(a, b) = (1, (x, 3))
    b: (Int, Int)
    x: Int
    val a, b = 1, (x, 3)
    val b = (x, 3)
    3: Int
    (x, 3): (Int, Int)
    val _ = (x, 3) => (Int, Int)

The return statement would either be (b, 1) or (b, a + 2).
If the return statement is (b, 1) then the return type would
be ((Int, Int), Int) since

b: (Int, Int)
1: Int
(b, 1) => ((Int, Int), Int)

If the return statement is (b, a + 2) the return type
would be ((Int, Int), Int) since:

a + 2 : Int because
    a : Int
```

```
    2 : Int
    _ + _: (Int, Int) => Int
b : (Int, Int)
(b, a + 2) => ((Int, Int), Int)

Since the return types of both of the return statements are the
same we can conclude that the return statement would be ((Int, Int), Int).
```

3 Problem 3

(d) ii.

No mutable variables? Yup!
Is your implementation tail-recursive? Yes it is!