Lab 1

Yuxiang Wang

Principles of Programming Languages

CSCI 3155-Fall 2017

Question 2:
(a) Consider the following Scala code.

```
1    val pi = 3.14
2    def circumference(r: Double): Double = {
3      val pi = 3.14159
4      2.0 * pi * r
5    }
6    def area(r: Double): Double =
7      pi * r * r
```

The use of pi at line 4 is bound at which line?

Answer: The use of pi at line 4 is bound at line 3. Because there is a new definition of pi inside function circumference in that scope, so overwriting the original definition of pi in that scope and pi at line 3 and line 4 are both in this scope.

The use of pi at line 7 is bound at which line?

Answer: The use of pi at line 7 is bound at line 1. Because pi at line 7 is out of the scope of function circumference, and there is no definition within area function, so the pi at line 7 need use the original definition of pi at line 1.

(b)

```
1    val x = 3
2    def f(x: Int): Int =
3        x match {
4            case 0 => 0
5            case x => {
6                val y = x + 1
7                ({
8                    val x = y + 1
9                    y
10               } * f(x - 1))
11           }
12       }
13   val y = x + f(x)
```

Lab 1

The use of x at line 3 is bound at which line?

Answer: The use of x at line 3 is bound on line 2. Because x is a parameter, it is passed into function and then used .

The use of x at line 6 is bound at which line?

Answer: The use of x at line 6 is bound on line 5. Because the line 5 is a match case of the match statement, and x at line 6 and line 5 within the same case. So they are in the same scope.

The use of x at line 10 is bound at which line?

Answer: The use of x at line 10 is bound by line 5. Because they're in the same match statement and same scope. The bracket on line 11 is the end of line 5.

The use of x at line 13 is bound at which line?

Answer: The use of x at line 13 is bound on line 1. Because x at line 13 outside of definition, that mean x at line 13 outside of the function. So the x at line 13 just can use the value from line 1.

Question 3:
Scala Basics: Typing. In the following, I have left off the return type of function g. The body of is well-typed if we can come up with a valid return type. Is the body of g well-typed?

```
1    def g(x: Int) = {
2        val (a, b) = (1, (x, 3))
3        if (x = 0) (b, 1) else (b, a + 2)
4    }
```

Answer: The body of g is well-typed and the return type is ((Int, Int) Int). Because no matter the value of x, this function returns either (b,1) or (b, a+2). And the type of 'a' is Int, type of 'b' is (Int, Int), so (b,1) and (b,a+2) have the same type((Int, Int)Int).

```
def g(x: Int): ((Int, Int), Int) because
  (b, 1): ((Int, Int) Int) because
      b: (Int, Int) because
          (x, 3): (Int, Int) because
              x: Int
              3: Int
      1: Int

  (b, a + 2): ((Int, Int) Int) because
```

Lab 1

---

b: (Int, Int) because
    (x, 3): (Int, Int) because
        x: Int
        3: Int
a + 2: int because
    a: Int
    2: Int