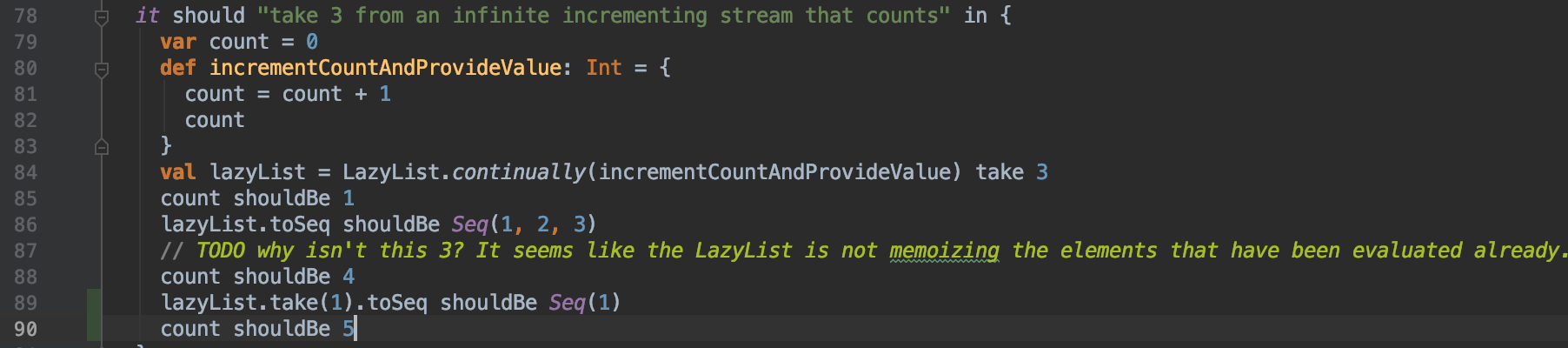
1)

(a) what is the chief way by which LazyList differs from Stream (the built-in Scala class that does the same thing). Don't mention the methods that LazyList does or doesn't implement--I want to know what is the structural difference.

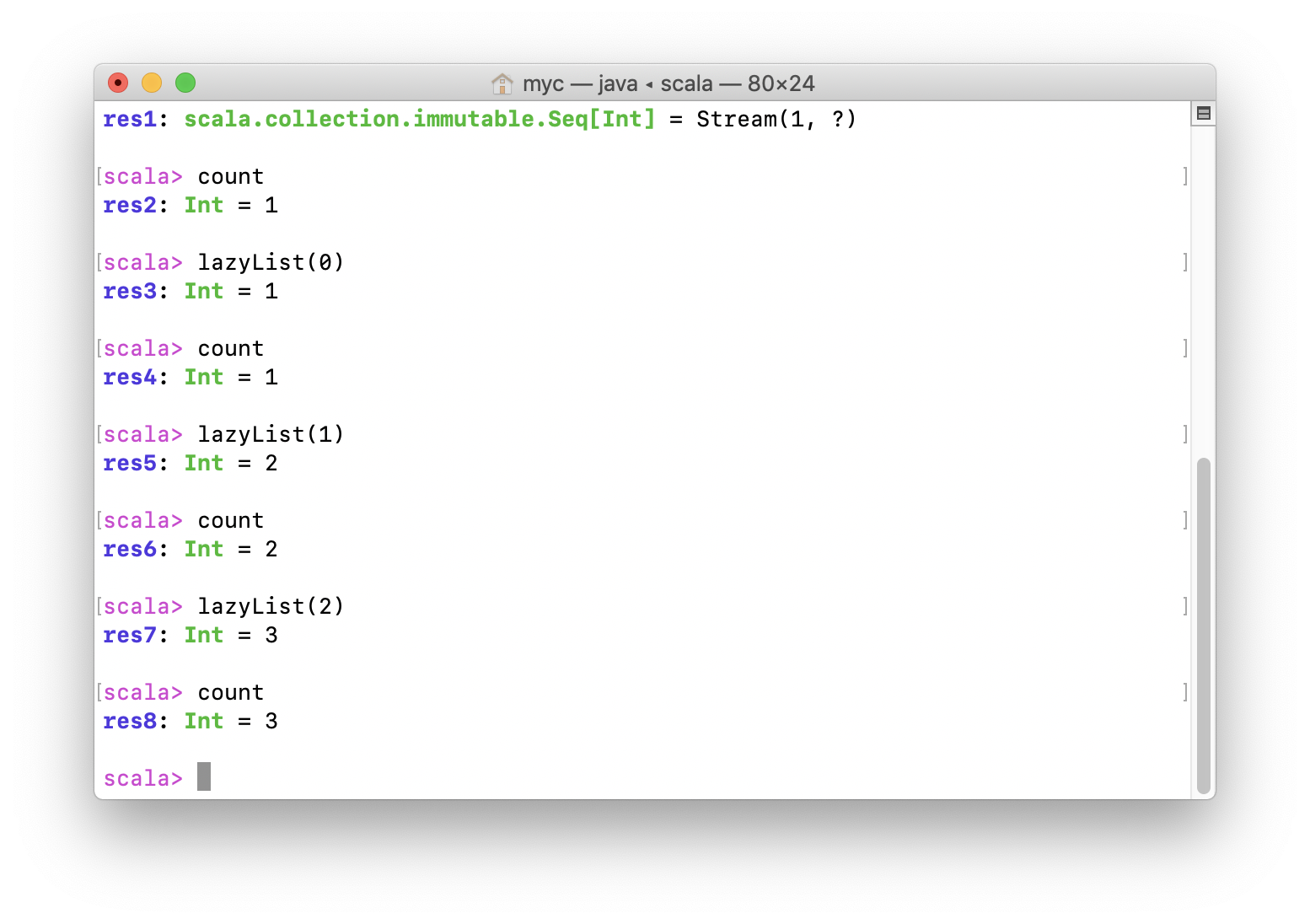
LazyList cannot memorize the element that has been evaluated , and LazyList.toSeq will make all element evaluated. In Stream, toSeq function will not evaluated all element and will memorize element that has been evaluated.

(b) Why do you think there is this difference?



run the code above in scalatest, print count we will get count=4 at last instead of 3 in Stream. Besides, After call toSeq function we will directly get count= 4 while Stream still get 1. The main reason is the LazyList cannot memorize the value has been evaluated, it can be proved by code at line 89 and 90. As lazyList, it will first  evaluates its arguments and then it uses them. So the incrementCountAndProvideValue function will be call once again, even the element has been evaluated.



In Stream, After call toSeq we still get count=1 as result.

To make difference, we make lazyList get evaluated by sequence, finally get 3 instead of 4. In this way, we can prove Stream can memorize element that evaluated.

2)Explain what the following code actually does and why is it needed? def tail = lazyTail()

recursively set tail for LazyList, it makes lazyList become lazy, only when lazyTail() invoked, the tail value in lazylist will be evaluated.

3)List all of the recursive calls that you can find in LazyList (give line numbers).

28,39,91,109,385,369,358,126

4)List all of the mutable variables and mutable collections that you can find in LazyList (give line numbers).

Collection : LazyList , EmptyList

5)What is the purpose of the zip method?

Zip two lazyList together and return a new listlike objects

6)Why is there no length (or size) method for LazyList?

Because the numbers of sequence(size) will be actually needed is determined by ourselves, so as lazyList we don’t need to apply size method. The actual size of lazyList will evaluated when needed. In other words, we cannot get specific size of lazylist before it is evaluated