Questions | Reading 1: Static Checking | 6.005.1x Courseware

openlearninglibrary.mit.edu/courses/course-v1:MITx+6.005.1x+3T2016/courseware/Readings_Videos/01-Static-Checking

1. Course , current location
2. Progress

Questions

snapshot diagrams and final

2/2 points (graded)

Suppose we want to add `final` to both variable declarations in the `hailstoneSequence` method, as shown:

```
public static List<Integer> hailstoneSequence(final int n) {
      final List<Integer> list = new ArrayList<Integer>();
      while (n != 1) {
          list.add(n);
          if (n % 2 == 0) {
              n = n / 2;
          } else {
              n = 3 * n + 1;
          }
      }
      list.add(n);
      return list;
  }
```

Which of the following are true statements about putting `final` on `n`?

correct

Which of the following are true statements about putting `final` on `list`?

correct

Explanation

`final` can't be used on `n` because `n` needs to be reassigned in the body of the method. But `final` can indeed be used on `list`.

`final` can be used on both parameters and local variables. When used on a parameter, `final` means that the parameter is assigned when the method is called, and then can't be reassigned during the body of the method. When used on a local variable, `final` means that the variable can't be reassigned after its first assignment, until the variable's scope ends.

`final` can be used on variables of any type -- not just immutable types like int, but also mutable types like `List`. If a `final` variable points to a mutable object, then the variable cannot be reassigned, but the object it points to can still be mutated, say by calling `add()` on a `List`.

Some problems have options such as save, reset, hints, or show answer. These options follow the Submit button.

Answers are displayed within the problem