Worksheet: Lazy Computation

You are given the following supplier for a lazy list of integers:

```
public static LazyList<Integer> from(int i, int step) {
  System.out.println("Hi");
  return cons(i,()->from(i+step, step));
}
```

In the following sequence of statements, write how many times will “Hi” be printed out as a side effect of executing that statement:

```
var nats = from(0, 1);       // 1
var evens = from(0, 2);      // 1
var alsoEvens = nats.filter(x -> x % 2 == 0);       // 0
var yetAnotherEvens = nats.map(x -> x * 2);         // 0
var thirdEven = evens.tail().tail().head();         // 2
var thirdEvenAgain = evens.tail().tail().head();    // 0
var fiveEvens = evens.take(5);                      // 0
var sumFiveEvens = fiveEvens.foldRight(0 , (x, y) -> x + y);       // 3
```


Why is fold evaluating an “extra” element?

```
public LazyList<T> take(int n) {
  if (n < 1) {
    return empty();
  } else {
    return cons(headVal, ()-> tail().take(n - 1));
  }
}
```

```
public LazyList<T> take(int n) {
  if (n < 1) {
    return empty();
  } else if (n == 1) {
    return cons(headVal, ()-> empty());
  } else {
    return cons(headVal, ()-> tail().take(n - 1));
  }
}
```