Consider the program below, written in SML/NJ. This program contains five routines that operate on lists. The first, `halve`, splits a lits into two balanced halves. Two of the other routines, `balanced` and `sublist`, are predicates, that is, functions that return booleans, based on some condition.

```
fun halve nil = (nil, nil)
  | halve [a] = ([a], nil)
  | halve (a::b::cs) =
    let
      val (x, y) = halve cs
    in
      (a::x, b::y)
    end

fun len nil = 0
  | len (_::t) = 1 + len t

fun balanced L0 L1 = abs(len(L0) - Len(L1)) <= 1

fun contains e nil = false
  | contains e (x::t) = e = x orelse contains e t

fun sublist nil _ = true
  | sublist (h::t) L = contains L h andalso sublist t L
```

Figure 1: Three programs written in SML/NJ that operate on lists.

1. Prove that if `halve L = (L0, L1)`, then `balanced L0 L1`

2. Prove that if `halve L = (L0, L1)`, then `sublist L0 L1`

3. Prove that if `halve L = (L0, L1)`, then `sublist L1 L0`