void appendV1 (QueueOfT& r, QueueOfT& g) // Using r for receiver, g for giver

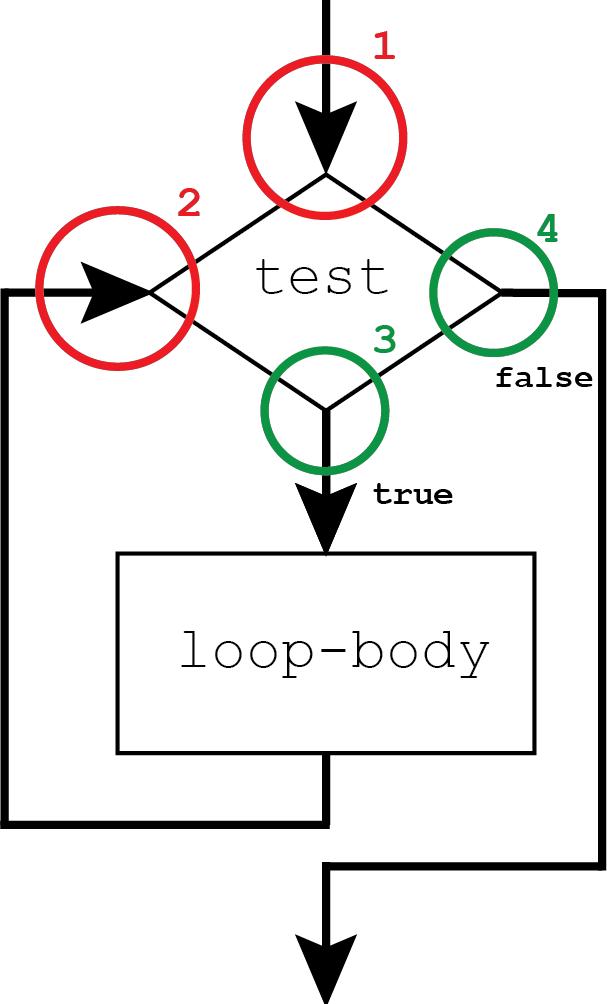
//! updates r

//! clears g

//! requires: true

//! ensures: r = #r \* #g

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S** | **Code** | **Assume** | | **Confirm** |
| 0 |  | true | | r0 \* g0 = r0 \* g0 |
|  | while(g.length() > 0) {  //! updates g, r  //! maintains  //! r \* g = #r \* #g  //! decreases |g| |  | |  |
| 1 |  | |g1| > 0 ^  r1 \* g1 = r0 \* g0 | |  |
|  | T y; |  | |  |
| 2 |  | T.Init(y2) | Unchanged r, g | g2 /= <> |
|  | g.dequeue(y); |  | |  |
| 3 |  | g3 = g2[1,|g2|) ^  <y3> = g2[0,1) | Unchanged r |  |
|  | r.enqueue(y); |  | |  |
| 4 |  | T.Init(y4) ^ r4 = r3 \* <y3> | Unchanged g | |g4| < |g1| ^  r4 \* g4 = r0 \* g0 |
|  | } |  | |  |
| 5 |  | ~(|g5| > 0) ^  r5 \* g5 = r0 \* g0 | | r5 = r0 \* g0 ^  g5 = <> |
|  |  |  | |  |
| 7 |  |  | |  |
|  |  |  | |  |

Loop invariant related reasoning:

* Must confirm it holds at and



* Get to assume it holds at and
* Allows us to reason about the loop as if it were a single statement