Reasoning Table for goToTheRear

Use: `T.Init(x)` as a predicate to state that variable x has initial value for its type T

```
void goToTheRear(QueueOfT& q);
   //! updates q
   //! requires   |q| > 0
   //! ensures    q = #q[1,|#q|) * #q[0,1)
```

State	Code	Assume	Confirm
0		*A0*: `|q0| > 0`	*C0*: `true` This *true* is requires clause from Type T's constructor
	`T y;`	/ / / / / / / / / / / / / / / / / / / /	/ / / / / / / / / / / / / / / / / / / /
1		*A1*: `T.Init(y1) ^ q1 = q0`	*C1*: `q1 /= <>`
	`q.dequeue(y);`	/ / / / / / / / / / / / / / / / / / / /	/ / / / / / / / / / / / / / / / / / / /
2		*A2*: `<y2> is prefix of q1 ^ q2 = q1[1,|q1|)`	*C2*: `true` This *true* is requires clause from enqueue
	`q.enqueue(y);`	/ / / / / / / / / / / / / / / / / / / /	/ / / / / / / / / / / / / / / / / / / /
3		*A3*: `q3 = q2 * <y2> ^ T.Init(y3)`	*C3*: `q3 = q0[1,|q0|) * q0[0,1)`

VCs written using *A0*, *A1*, *A2*, and *A3* cell labels

VC Format: *antecedent* → *consequent*

VC0: `A0 → true`
VC1: `(A0 ^ A1) → q1 /= <>`
VC2: `(A0 ^ A1 ^ A2) → true`
VC3: `(A0 ^ A1 ^ A2 ^ A3) → q3 = q0[1,|q0|) * q0[0,1)`

Where:

```
A0 = |q0| > 0
A1 = T.Init(y1) ^ q1 = q0
A2 = <y2> is prefix of q1 ^ q2 = q1[1,|q1|)
A3 = q3 = q2 * <y2> ^ T.Init(y3)
```