Exercise 1.

1. state after `let r = ref 5 and s = ref 3 and t = r`:
2. state after subsequently executing `incr r`:
3. state after subsequently executing `incr t`:

Exercise 2. **in-place list reversal**

State before the loop:

State after the loop:

Loop invariant:

Exercise 3. **length of mutable list using a while loop**

State before the loop:

State after the loop:

Picture describing the state during the loop:

Try to state a loop invariant. What do you need?

Exercise 4.

generalize MList to define $p \leadsto \mathsf{MlistSeg}\, q\, L$, where L denotes the list of items in the list segment from p (inclusive) to q (exclusive):

$p \leadsto \mathsf{MlistSeg}\, q\, L \quad \equiv$

Exercise 5. **length of mutable list using a while loop and** MlistSeg

Loop invariant: $\exists q, L_1, L_2. \ \ldots$

Instantiate q, L_1, L_2 before the loop:

Instantiate q, L_1, L_2 after the loop:

Exercise 6. define the representation predicate $p \rightsquigarrow \mathsf{Queue}\, L$.

Exercise 7. define the representation predicate $p \rightsquigarrow \mathsf{Mtree}\, T$.

Exercise 8. define $p \rightsquigarrow \mathsf{MtreeDepth}\, n\, T$ by generalizing $p \rightsquigarrow \mathsf{Mtree}\, T$.

Exercise 9. give an alternative definition of "$p \rightsquigarrow \mathsf{MtreeDepth}\, n\, T$", this time by reusing the definition of $p \rightsquigarrow \mathsf{Mtree}\, T$ without modification.

Exercise 10. define a predicate $p \rightsquigarrow \mathsf{MtreeComplete}\, T$ for describing a mutable complete binary tree, of some unspecified depth.

Exercise 11. define a predicate $p \rightsquigarrow \mathsf{MsearchTree}\, E$ for describing a mutable binary search tree storing the set of elements E.

Exercise 12. specify the primitive operations on references.

```
(ref v)
(!r)
(r := v)
```

Exercise 13. Give specifications for:

```
(Array.get i p)
(Array.set i p v)
(Array.length p)
(Array.create n v)
```

Exercise 14. What is the *natural* specification of function `myref`? What is missing from our current interpretation of triple?