Exercises for MPRI Separation Logic course 4

\[ p \leadsto \text{Mlistof } R \ L \equiv \text{match } L \text{ with } \]
\[ | \text{nil } \Rightarrow [p = \text{null}] \]
\[ | X :: L' \Rightarrow \exists x'. \ p \leadsto \{ \text{hd} = x; \ \text{tl} = p'\} \]
\[ \star p' \leadsto \text{Mlistof } R \ L' \]
\[ \star x \leadsto R \ X \]

**Exercise:** since \((p : \text{loc})\) and \((x : \text{Val})\) and \((X : A)\) for some \(A\), what is the type of \(R\)? What is the type of \(\text{Mlistof}\)?

\[ p \leadsto \text{MList } L \equiv \text{match } L \text{ with } \]
\[ | \text{nil } \Rightarrow [p = \text{null}] \]
\[ | x :: L' \Rightarrow \exists x'. \ p \leadsto \{ \text{hd} = x; \ \text{tl} = p'\} \]
\[ \star p' \leadsto \text{MList } L' \]

**Exercise:** define the identity representation predicate \(\text{Id}\) such that
\[ p \leadsto \text{Mlistof Id } L = p \leadsto \text{MList } L \]

**Exercise:** specify functions over queues using a higher-order representation predicate written \(p \leadsto \text{Queueof } R \ L\).

Shorthand: just write “Q R” instead of “Queueof R”.

```latex
\{ \} \ (create()) \{ \}
\{ \} \ (push \ x \ p) \{ \}
\{ \} \ (pop \ p) \{ \}
\{ \} \ (concat \ p \ p') \{ \}
```
Exercise: specify a function \texttt{copy f p} that duplicables a mutable queue specified using \texttt{Queueof}, where \textit{f} is a function to duplicate items.

\[
(\forall x X. \{ \{ f x \} \}) \Rightarrow \{ p \leadsto \texttt{Queueof R L} \}
\]

\[
(\text{copy f p}) \Rightarrow \{ \lambda p'. p \leadsto \texttt{Queueof R L} \star \ p' \leadsto \texttt{Queueof R L} \}
\]

\[
p \leadsto \texttt{MCellof R_{1} V_{1} R_{2} V_{2}} \equiv \exists v_{1} v_{2}. \ p \leadsto \{ \text{hd}=v_{1}; \text{tl}=v_{2} \} \\
\star \ v_{1} \leadsto \texttt{R_{1} V_{1}} \\
\star \ v_{2} \leadsto \texttt{R_{2} V_{2}}
\]

Exercise: rewrite the specification of \texttt{Mlistof} using \texttt{MCellof}.

\[
p \leadsto \texttt{Mlistof R L} \equiv \text{match } L \text{ with} \\
| \texttt{nil} \Rightarrow [p = \text{null}] \\
| X :: L' \Rightarrow
\]

Exercise: rewrite the specification of \texttt{Narytreeof} using \texttt{Nodeof}.

\[
p \leadsto \texttt{Narytreeof R T} \equiv \\
\text{match } T \text{ with} \\
| \texttt{Leaf} \Rightarrow [p = \text{null}] \\
| \texttt{Node X L} \Rightarrow
\]

Exercise: complete the specification of \texttt{Bagof} using \texttt{Nodeof}.

Hint: chunks are described by the predicate \texttt{p' \leadsto Chunkof R E'}.

\[
p \leadsto \texttt{Bagof R T} \equiv \\
\text{match } T \text{ with} \\
| \texttt{Empty} \Rightarrow [p = \text{null}] \\
| \texttt{Layer E' T'} \Rightarrow
\]
**Exercise:** specify the function `miter`, using an invariant of the form $JKK'$, describing the state before and the state after the iteration.

$$
\forall f p RLJ. \ (\forall x X . \ {x \sim R X} \ \{x \sim R X\})
$$

$$
(f\ x)
\{\lambda_\_\} \\
\Rightarrow \ {p \sim \text{Mlistof} RL} \ * \ \{\text{miter} f\ p\} \\
\{\lambda_\_\} \\
$$

```ocaml
let incr_all p = 
  miter (fun x -> incr x) p

let example_p = 
  { hd = ref 5; tl = { hd = ref 3; tl = null } }
```

Exercise: using the representation predicates `Ref` and `Mlistof`, specify the function `(fun x -> incr x)` and `incr_all`.

```ocaml
{ } (incr\ x)\ \{\lambda_\_\} \\
{ } (incr_all\ p)\ \{\lambda_\_\} \\
```

Exercise: Describe the state at the front of each lines (except 5 and 6). Explicit the instantiation of the existential in the invariant.

```ocaml
1  let r = ref 0
2  let s = ref n
3  let p = create_lock()
```
4
5   let concurrent_step () =
6       let () = acquire_lock p in
7       incr r;
8       decr s;
9       release_lock p

**Exercise:** state a conversion rule relating \( p \leadsto \text{Cellsof } R M \) with a predicate of the form \( p \leadsto \text{Cellsof Id } M' \).

Hint: \( (R : A \to a \to \text{Hprop}) \) and \( (M : \text{map int } A) \) and \( (M' : \text{map int } a) \).

\[ p \leadsto \text{Cellsof } R M = \]