1. Given a concurrent system specified by a rewrite theory \( \mathcal{R} \), recall that an invariant is specified as a Boolean-valued predicate \( I : \text{State} \rightarrow \text{Bool} \) with equations protecting \( \text{BOOL} \), where \( \text{State} \) is the chosen sort of states in \( \mathcal{R} \). This can be generalized to the notion of a parametric invariant, as a Boolean-valued function \( I : \text{State} \rightarrow \text{BOOL} \). Then, in \( I(S, x_1, \ldots, x_n) \) the \( x_1 : A_1, \ldots, x_n : A_n \) are called the data parameters of the invariant.

Let now \( \text{init}(x_1 : A_1, \ldots, x_n : A_n) \) be a term of sort \( \text{State} \) whose only variables are the \( x_1 : A_1, \ldots, x_n : A_n \). We then say that the parametric invariant \( I(S, x_1, \ldots, x_n) \) holds in the initial reachability model \( \mathcal{T}_\mathcal{R} \) for the parametric family of initial states \( \text{init}(x_1 : A_1, \ldots, x_n : A_n) \) with data parameters \( x_1 : A_1, \ldots, x_n : A_n \) if and only if for each ground substitution \( \rho = \{ x_1 \mapsto u_1, \ldots, x_n \mapsto u_n \} \) of the variables \( x_1 : A_1, \ldots, x_n : A_n \) the (unparametric) invariant \( I(S, u_1, \ldots, u_n) \) holds from the (ground) initial state \( \text{init}(u_1, \ldots, u_n) \).

Consider now the following unordered communication channel between a sender and a receiver:

```maude
mod COMM-CHANN is protecting NAT . protecting QID .
  sorts Msg MsgMSet QidList State .
  subsort Msg < MsgMSet .
  subsort Qid < QidList .
  op nil : -> QidList [ctor] .
  op _;_ : QidList QidList -> QidList [ctor assoc id: nil] .
  op [_,_] : Qid Nat -> Msg [ctor] .
  op null : -> MsgMSet [ctor] .
  op _ _ : MsgMSet MsgMSet -> MsgMSet [ctor assoc comm id: null] .
  op [_:_|_|_|_] : QidList Nat MsgMSet Nat QidList -> State [ctor] .

  vars N M I J : Nat . var MS : MsgMSet . vars L L' : QidList .
  vars A B : Qid .

endm
```

where both the sender (on left) and the receiver (on right) hold buffers (lists) with lists of Qids to be sent (resp. already received), and the channel (in the middle) is a multiset of messages. We assume both counters will initially be at 0, and that the receiver’s buffer will initially hold the \( \text{nil} \) list of Qids.

Intuitively, the counters are used to ensure in order reception. That is, even though the channel is a multiset so that the messages can get out of order, we expect and desire that, at any given time during the sending and receiving process, the list of Qids in the receiver’s buffer will be arranged in the exact same order in which they were initially in the sender’s buffer, although, of course, only at the end of the sending and receiving process will the receiver hold the entire list sent by the sender.

You are asked to do the following:

- Define the above property of in-order-reception as a parametric invariant of the form:
op in-order-reception : State QidList -> Bool .

**Hint:** the use of Maude's `owise` feature can make the definition easier.

- Write out the parametric family `init(L : QidList)` of initial states for which the parametric invariant is an invariant.
- Show a screenshot of your results for verifying the invariant when `L` is instantiated to the Qid lists:
  - 'a ; 'b ; 'c
  - 'a ; 'b ; 'c ; 'd
  - 'a ; 'b ; 'c ; 'd ; 'e

2. In Exercise 1 above, the notion of **parametric invariant** has been explored for the `COMM-CHANN` example. This exercise extends these ideas to the LTL case to make you familiar with the use of **parametric state predicates** in LTL. In particular, proving a parametric invariant `I(S, x_1, \ldots, x_n)` from an initial state `init(x_1 : A_1, \ldots, x_n : A_n)` can, when the set of reachable states is finite, also be done by model checking an LTL formula of the form `\Box I(x_1, \ldots, x_n)`, were now we have defined `I` as a parametric atomic proposition of the form `I : A_1 \ldots A_n \rightarrow Prop` and have defined its meaning by giving equations using the `SATISFACTION` module.

In the slightly modified version of `COMM-CHANN` below you are asked to do two things:

- define again and prove by LTL model checking the parametric invariant `in-order-reception` for the initial states `init1, init2, and init3` below, and
- define a new parametric atomic proposition called `received` that holds when the protocol has finished if the exact same list originally held by the sender is now in the buffer of the receiver, and state and model check the instance of the LTL parametric property expressing that such a state is indeed always reached for initial states `init1, init2, and init3` below.

To make things easier for you, we provide a template indicating what you need to define. This template is available on the course webpage in the file `comm-chan-ltl.maude`.

```maude
mod COMM-CHANN is protecting NAT . protecting QID .
sorts Msg MsgMSet QidList CHState .
subsort Msg < MsgMSet .
subsort Qid < QidList .
vars N M I J : Nat . var MS : MsgMSet . vars L L' : QidList .
vars A B : Qid .
endm

load model-checker.maude

mod COMM-CHANN-PREDS is protecting COMM-CHANN . including SATISFACTION .
subsort CHState < State .
op in-order-reception : QidList -> Prop .
op received : QidList -> Prop .
```
vars N M I J : Nat . var MS : MsgMSet . vars L L' L'' : QidList .

*** add your defining equations here

demd

mod COMM-CHANN-CHECK is
  protecting COMM-CHANN-PREDs .
  including MODEL-CHECKER .
  ops init1 init2 init3 : -> CHState .
eq init1 = ['a ; 'b ; 'c : 0 | null | 0 : nil ] .
eq init2 = ['a ; 'b ; 'c ; 'd : 0 | null | 0 : nil ] .
eq init3 = ['a ; 'b ; 'c ; 'd ; 'e : 0 | null | 0 : nil ] .
edmd

*** add your LTL commands here