(a) **Mutual Exclusion**: processes 1 and 2 are never simultaneously in their critical sections. Like before, use $c_1$ and $c_2$ as your atomic propositions.

$$\forall \square (\neg c_1 \lor \neg c_2)$$

(b) **Mutual Exclusion**: each process has access to its critical section infinitely often.

$$\forall \square \forall \Diamond c_1 \land \forall \square \forall \Diamond c_2$$

(c) **System Restart**: in every reachable state of the system, it is possible to return to a start state of the system. Use the proposition $start$ which is true when the system is in any of its start states.

$$\forall \square \exists \Diamond start$$

(d) **System**: if a system never deadlocks ($d$), then there is a possibility for something good ($g$) to eventually happen.

$$\forall \square \neg d \implies \exists \Diamond g$$

(e) **Traffic Light**: If the light is red and stays on, then there is always a possibility that it can eventually turn green in the future.

Exercise for you!
(f) **Traffic Light**: each red light phase is definitely preceded by a yellow light phase.

Exercise for you!

(h) **Prove or disprove**:

\[ \forall \Diamond \Phi \land \forall \Diamond \Psi = \forall (\Phi \land \Psi) \]

No. Counterexample:

Let \( \Phi = a \) and \( \Psi = b \). The execution tree below satisfies the left side but not the right.

\[ \forall \Diamond \Phi \lor \forall \Diamond \Psi = \forall (\Phi \lor \Psi) \]

No. Counterexample:

Let \( \Phi = a \) and \( \Psi = b \). The execution tree below satisfies the right side but not the left.

\[ \exists \Diamond \Phi \lor \exists \Diamond \Psi = \exists (\Phi \lor \Psi) \]
Yes, proof:

\[ TS \models \exists \diamond \Phi \lor \exists \diamond \Psi \]

by def of \( \lor \) \( \iff \) \( TS \models \exists \diamond \Phi \lor TS \models \exists \diamond \Psi \)

by def of \( \exists \) \( \iff \) \( (\exists \pi \in \text{Paths}(TS) : \pi \models \diamond \Phi) \lor (\exists \pi' \in \text{Paths}(TS) : \pi' \models \diamond \Psi) \)

by basic logic \( \iff \) \( (\exists \pi \in \text{Paths}(TS) : \pi \models \diamond \Phi \lor \diamond \Psi) \)

by basic logic \( \iff \) \( \exists \pi \in \text{Paths}(TS) : \pi \models \diamond \Phi \lor \diamond \Psi \)

by Lemma 1 \( \iff \) \( \exists \pi \in \text{Paths}(TS) : \pi \models \diamond (\Phi \lor \Psi) \)

by def of \( \exists \) \( \iff \) \( TS \models \exists \diamond (\Phi \lor \Psi) \)

**Lemma 1:** (you can inline this, but the proof is more readable this way) For all \( \pi \):

\[ \pi \models \diamond \Phi \lor \diamond \Psi \]

by def of \( \lor \) \( \iff \) \( \pi \models \diamond \Phi \lor \pi \models \diamond \Psi \)

by def of \( \diamond \) \( \iff \) \( \exists i \geq 0 : \pi[i] \models \Phi \lor \exists j \geq 0 : \pi[j] \models \Psi \)

by basic logic \( \iff \) \( \exists i \geq 0 : \pi[i] \models \Phi \lor \Psi \lor \exists j \geq 0 : \pi[j] \models \Phi \lor \Psi \)

by def of \( \diamond \) \( \iff \) \( \pi \models \diamond (\Phi \lor \Psi) \lor \diamond (\Phi \lor \Psi) \)

by basic logic \( \iff \) \( \pi \models \diamond (\Phi \lor \Psi) \)

The cleaner (non-lazy) proof of the reverse direction:

\[ TS \models \exists \diamond \Phi \lor \exists \diamond \Psi \]

by def of \( \lor \) \( \iff \) \( TS \models \exists \diamond \Phi \lor TS \models \exists \diamond \Psi \)

by def of \( \exists \) \( \iff \) \( (\exists \pi \in \text{Paths}(TS) : \pi \models \diamond \Phi) \lor (\exists \pi' \in \text{Paths}(TS) : \pi' \models \diamond \Psi) \)

by distributivity of \( \exists \) over \( \lor \) \( \iff \) \( \exists \pi \in \text{Paths}(TS) : \pi \models \diamond (\Phi \lor \Psi) \)

by Lemma 1 \( \iff \) \( \exists \pi \in \text{Paths}(TS) : \pi \models \diamond (\Phi \lor \Psi) \)

by def of \( \exists \) \( \iff \) \( TS \models \exists \diamond (\Phi \lor \Psi) \)