SOEN 331: Formal Methods for Software Engineering Various problems with solutions

Dr. Constantinos Constantinides, P.Eng.

Department of Computer Science and Software Engineering
Concordia University Montreal, Canada

November 25, 2021

Part 1: Propositional logic

Problem 1

- You are shown a set of four cards placed on a table, each of which has a **shape** on one side and a **number** on the other side.
- The visible faces of the cards show the shapes **rectangle**, and **circle**, and the numbers **4**, and **7**.
- Which card(s) must you turn over in order to test the truth of the proposition that "If a card has a circle on one side, then it has an odd number on the other side"?

Problem 1: Solution

- Recall the arrangement: **rectangle**, **circle**, **4**, **7**.
- First, we summarize the proposition as **circle** $\rightarrow$ **odd**.
- By Modus Ponens we must turn **circle** and expect an odd number.

$$p \rightarrow q,\ p,\ \therefore q$$

- By Modus Tollens we must turn **4** and expect a non-circle.

$$p \rightarrow q,\ \neg q,\ \therefore \neg p$$

Problem 1: Solution /cont.

- Recall the arrangement: **rectangle**, **circle**, **4**, **7**.
- Recall the proposition: **circle** $\rightarrow$ **odd**.
- It would be wrong to select **square** and expect an even number. This is an invalidating pattern called "inverse error."

$$p \rightarrow q,\ \neg p,\ \therefore \neg q$$

- It would also be wrong to select **7** and expect a circle. This is an invalidating pattern called "converse error."

$$p \rightarrow q,\ q,\ \therefore p$$

Problem 2

- “If every prime number is a multiple of 4, and every multiple of 4 is an even number, then every prime number is even.”

- Is this an argument? Is it valid? Is it sound?

Problem 2: Solution

- "If every prime number is a multiple of 4, and every multiple of 4 is an even number, then every prime number is even."
- This is a valid argument form called "hypothetical syllogism" (or "transitivity').
- Formally this is expressed as $p \to q, q \to r \ \therefore p \to r$.
- It is not sound since not all premises are true.

Problem 3

- In a paper titled "Computing Machinery and Intelligence" published in 1950, **Alan Turing** wrote:

  "*If each man had a definite set of rules of conduct by which he regulated his life he would be no better than a machine. But there are no such rules, so men cannot be machines*."

- Is this a valid argument?

Problem 3: Solution

- Premise **If** each man had a definite set of rules of conduct by which he regulated his life, [**Then**] he would be no better than a machine. ($p \rightarrow q$)
- Premise There is no set of rules of conduct. ($\neg p$)
- Conclusion Therefore, men cannot be machines. ($\neg q$)

- This is a non-validating pattern called "Denying the antecedent" (or "inverse error").
- In the article, Turing actually states that this is an example of an invalid argument.

Part 2: Predicate logic

Problem 1

- Let $p(x)$ denote the statement "x is a politician", and $q(x)$ denote the statement "x is crooked." Formalize the following sentences:
- There is an honest politician.
- No politician is honest.
- All politicians are honest.
- No politician is crooked.
- Some politicians are crooked.

Problem 1: Solution

- Let $p(x)$ denote the statement "x is a politician", and $q(x)$ denote the statement "x is crooked." Formalize the following sentences:

- There is an honest politician: $\exists x(p(x) \land \neg q(x))$.

- No politician is honest: $\forall x(p(x) \rightarrow q(x))$.

- All politicians are honest: $\forall x(p(x) \rightarrow \neg q(x))$.

- No politician is crooked: $\forall x(p(x) \rightarrow \neg q(x))$.

- Some politicians are crooked: $\exists x(p(x) \land q(x))$.

Problem 2

- Consider the sentence "All that glitters is gold."
- Translate the predicate in formal logic for predicates $glitters(x)$ meaning "x glitters" and $gold(x)$ meaning "x is gold."

Problem 2: Solution

Translate the predicate in formal logic for predicates $glitters(x)$ meaning "x glitters" and $gold(x)$ meaning "x is gold."

$$\forall x(glitters(x) \rightarrow gold(x))$$

Problem 3

Use the two predicates *glitters*(x) and *gold*(x) to build the 4 types of categorical propositions in formal logic, clearly indicating each form.

Problem 3: Solution

Use the two predicates *glitters*(x) and *gold*(x) to build the 4 types of categorical propositions in formal logic, clearly indicating each form.

A $\forall x\,(glitters(x) \rightarrow gold(x))$

E $\forall x\,(glitters(x) \rightarrow \neg gold(x))$

I $\exists x\,(glitters(x) \wedge gold(x))$

O $\exists x\,(glitters(x) \wedge \neg gold(x))$

Problem 4

- Given the sentence: *The city is now empty / No human soul remains*.
- We can identify two predicates *soul*(x) that reads "x is human soul" and *remains*(x) which reads "x remains."
- Build categorical propositions in English, identify their forms, and translate them into formal logic.

Problem 4: Solution

A All human souls remain.
$\forall x(\textit{soul}(x) \rightarrow \textit{remains}(x))$.

E No human souls remain.
$\forall x\,(\textit{soul}(x) \rightarrow \neg \textit{remains}(x))$

I Some human souls remain.
$\exists x\,(\textit{soul}(x) \wedge \textit{remains}(x))$

O Some human souls do not remain.
$\exists x\,(\textit{soul}(x) \wedge \neg \textit{remains}(x))$

Problem 5

From the previous problem, identify pairs that are

- contradictories,
- contraries,
- subcontraries, and
- pairs that support subalteration.

Problem 5: Solution

Contradictories:

- "All human souls remain" and "Some human souls do not remain."
- "No human souls remain" and "Some human souls remain."

Problem 5: Solution /cont.

Contraries:

- "All human souls remain" and "No human souls remain."

Problem 5: Solution /cont.

Subontraries:

- "Some human souls remain" and "Some human souls do not remain."

Problem 5: Solution /cont.

Subalteration:

- "Some human souls remain" is a subaltern to "All human souls remain."
- "Some human souls do not remain" is a subaltern to "No human souls remain."

Part 3: Binary relations

Problem 1

The binary relation "is reachable from" on the set of nodes of a directed mathematical multigraph, can be best described as follows:

Reflexive?	
Irreflexive?	
Symmetric?	
Asymmetric?	
Antisymmetric?	
Transitive?	

Problem 1: Solution

The binary relation "is reachable from" on the set of nodes of a directed mathematical multigraph, can be best described as follows:

Reflexive	$\forall\, a \in A : aRa$	✓
Irreflexive	$\forall\, a \in A : \neg(aRa)$	×
Symmetric	$\forall\, a, b \in A : aRb \to bRa$	×
Asymmetric	$\forall\, a, b \in A : aRb \to \neg bRa$	×
Antisymmetric	$\forall\, a, b, c \in A : (aRb \wedge bRa) \to a = b$	×
Transitive	$\forall\, a, b, c \in A : (aRb \wedge bRc) \to aRc$	✓

Problem 2

- Consider the binary relation “x divides y” on the set of natural numbers.
- Discuss reflexivity, symmetry, antisymmetry and transitivity.

Problem 2: Solution

- Consider the binary relation "x divides y" on the set of natural numbers.
- **Reflexivity** is defined as $\forall\, a \in A : aRa$.
- Since x is always a divisor of itself, the relation is reflexive.
- **Symmetry** is defined as $\forall\, a, b \in A : aRb \rightarrow bRa$.
- The relation is not symmetric since not for every x being a divisor of y it is also the case that y is also a divisor of x.

Problem 2: Solution /cont.

- Consider the binary relation "x divides y" on the set of natural numbers.
- **Transitivity** is defined as $\forall\, a, b, c \in A : (aRb \land bRc) \rightarrow aRc$.
- The relation is transitive since if x is a divisor of y and if y is a divisor of z, then x is also a divisor of z.

Problem 3

- If R is the relation "is sister of" and S is the relation "is the mother of" on the set of all people, describe the relations
- $R \circ S$, and
- $S \circ S$.

Problem 3: Solution

- R: "is sister of", and S: "is the mother of" on the set of all people.
- Recall that for $R \subseteq X \times Y$, and $S \subseteq Y \times Z$, then

  $$R \circ S = \{(x, z) \in X \times Z \mid (\exists\, y \in Y : (x, y) \in R \land (y, z) \in S\}$$

- $R \circ S$: If a is the sister of b, and b is the mother of c, then a is a sister of the mother of c. In other words, a is an aunt of c. Thus, $R \circ S$ captures the relation "is an aunt of."
- $S \circ S$ captures the relation "is a grandmother of."

Problem 4

- Let $A = \{0, 1, 2, 3\}$ and relation R over A defined as follows:

$$R = \{(0,0), (0,1), (0,3), (1,0), (1,1), (2,2), (3,0), (3,3)\}$$

- Is R an equivalence relation, a partial order, or neither?

Problem 4: Solution

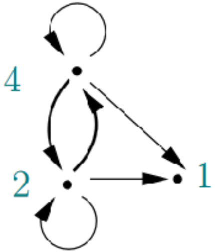
- Let $A = \{0, 1, 2, 3\}$ and a relation R over A to be defined as follows:

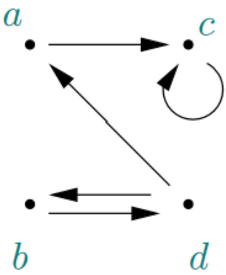
$$R = \{(0,0), (0,1), (0,3), (1,0), (1,1), (2,2), (3,0), (3,3)\}$$

- The relation is reflexive, symmetric and not transitive.
- It is, thus, neither an equivalence relation, nor a partial order.

Problem 5

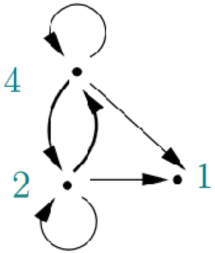
- For each of the relations shown below, list its ordered pairs and determine all their properties.

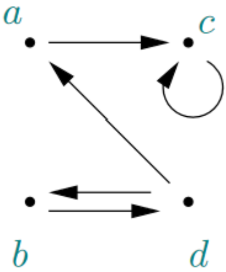




Problem 5: Solution

- For the relations shown below





- The ordered pairs are
  1. $\{(2,1),(2,2),(2,4),(4,1),(4,2),(4,4)\}$
  2. $\{(a,c),(b,d),(c,c),(d,a),(d,b)\}$

Problem 5: Solution /cont.

		(a)	(b)
Reflexive	$\forall\, a \in A : aRa$	$\times$	$\times$
Irreflexive	$\forall\, a \in A : \neg(aRa)$	$\times$	$\times$
Symmetric	$\forall\, a, b \in A : aRb \rightarrow bRa$	$\times$	$\times$
Asymmetric	$\forall\, a, b \in A : aRb \rightarrow \neg bRa$	$\times$	$\times$
Antisymmetric	$\forall\, a, b, c \in A : (aRb \wedge bRa) \rightarrow a = b$	$\times$	$\times$
Transitive	$\forall\, a, b, c \in A : (aRb \wedge bRc) \rightarrow aRc$	$\checkmark$	$\times$

Part 4: Functions

Problem 1

- Given sets $A = \{a, b, c\}$, and $B = \{1, 2, 3\}$ decide which of the functions is injective, surjective or bijective.
- $\{a \mapsto 1, b \mapsto 1, c \mapsto 3\}$
- $\{a \mapsto 1, b \mapsto 3, c \mapsto 2\}$

Problem 1: Solution

- Given sets $A = \{a, b, c\}$, and $B = \{1, 2, 3\}$ decide which of the functions is injective, surjective or bijective.
- $\{a \mapsto 1, b \mapsto 1, c \mapsto 3\}$: Not injective since the value 1 is repeated. Not surjective since the element 2 in the codomain is not a value of the function.
- $\{a \mapsto 1, b \mapsto 3, c \mapsto 2\}$: Injective since values are not repeated and surjective since the range and codomain coincide. By definition it is also bijective.

Problem 2

- Given sets $A = \{a, b, c\}$, and $B = \{1, 2\}$ decide whether the following function is injective, surjective or bijective.

- $\{a \mapsto 1, b \mapsto 1, c \mapsto 2\}$

Problem 2: Solution

- Given sets $A = \{a, b, c\}$, and $B = \{1, 2\}$ decide whether the following function is injective, surjective or bijective.
- $\{a \mapsto 1, b \mapsto 1, c \mapsto 2\}$: Not injective since the value 1 is repeated. It is surjective since the range and the codomain are the same.

Problem 3

- Given sets $A = \{a, b\}$, and $B = \{1, 2, 3\}$ decide whether the following function is injective, surjective or bijective.

- $\{a \mapsto 3, b \mapsto 1\}$

Problem 3: Solution

- Given sets $A = \{a, b\}$, and $B = \{1, 2, 3\}$ decide whether the following function is injective, surjective or bijective.

- $\{a \mapsto 3, b \mapsto 1\}$: Injective but not surjective.

Problem 4

- Given *Odd* and *Even* that represent the sets of odd and even natural numbers respectively.
- Consider function $f : Odd \to Even$ defined by $f(x) = x - 1$.
- Is f injective, surjective, bijective?
- Does there exist an inverse function?

Problem 4: Solution

- The function $f(x) = x - 1$ is a bijection.
- Its inverse function is $f^{-1} = x + 1$.

Functions problems 5 - 9

The following applies to Problems 5 - 9.

Binary relations are sets of pairs and can model lookup tables.

$$
\begin{array}{|l}
\mathit{Employees} : \mathit{NAME} \leftrightarrow \mathit{Phone} \\
\hline
\mathit{Employees} = \\
\quad \{ \\
\qquad (\mathit{frank} \mapsto 0110), \\
\qquad (\mathit{philip} \mapsto 0113), \\
\qquad (\mathit{aki} \mapsto 4019), \\
\qquad (\mathit{doug} \mapsto 4107), \\
\qquad \ldots \\
\quad \}
\end{array}
$$

Problem 5

What is the result of

$$\{\mathit{frank}, \mathit{doug}\} \lhd \mathit{Employees} =$$

Problem 5: Solution

Domain restriction selects pairs based on their first element:

$$
\begin{array}{l}
\{\mathit{frank}, \mathit{doug}\} \lhd \mathit{Employees} = \\
\quad \{ \\
\qquad (\mathit{doug} \mapsto 4107), \\
\qquad (\mathit{frank} \mapsto 0110) \\
\quad \}
\end{array}
$$

Problem 6

What is the result of

$$Employees \rhd \{4000..4999\} =$$

Problem 6: Solution

Range restriction selects pairs based on their second element:

$$
\begin{array}{l}
\mathit{Employees} \rhd \{4000..4999\} = \\
\quad \{ \\
\quad\quad (\mathit{aki} \mapsto 4019), \\
\quad\quad (\mathit{doug} \mapsto 4107) \\
\quad \}
\end{array}
$$

Problem 7

What is the result of

$$Employees' = Employees \oplus \{frank \mapsto 0178\} =$$

Problem 7: Solution

Overriding can model database updates:

$$
\begin{array}{l}
\mathit{Employees}' = \mathit{Employees} \oplus \{\mathit{frank} \mapsto 0178\} = \\
\quad \{ \\
\qquad (\mathit{frank} \mapsto 0178), \\
\qquad (\mathit{philip} \mapsto 0113), \\
\qquad (\mathit{aki} \mapsto 4019), \\
\qquad (\mathit{doug} \mapsto 4107), \\
\qquad \ldots \\
\quad \}
\end{array}
$$

Problem 8

What is the result of

$$Employees' = \{frank\} \lhd Employees =$$

Problem 8: Solution

The domain subtraction of a function or binary relation R by a set A, denoted as

$$A \ntriangleleft R$$

removes all elements of A from the domain of the function:

$$
\begin{array}{l}
\mathit{Employees}' = \{\mathit{frank}\} \ntriangleleft \mathit{Employees} = \\
\quad \{ \\
\qquad (\mathit{philip} \mapsto 0113), \\
\qquad (\mathit{aki} \mapsto 4019), \\
\qquad (\mathit{doug} \mapsto 4107), \\
\qquad \ldots \\
\quad \}
\end{array}
$$

Problem 9

What is the result of

$$Employees' = Employees \rhd \{4107\} =$$

Problem 9: Solution

The range subtraction of a function of binary relation R by a set B, denoted as

$$R \rhd B$$

removes all elements of B from the codomain of the function:

$$\begin{aligned}
&\mathit{Employees}' = \mathit{Employees} \rhd \{4107\} = \\
&\quad \{ \\
&\quad\quad (\mathit{philip} \mapsto 0113), \\
&\quad\quad (\mathit{aki} \mapsto 4019), \\
&\quad\quad \ldots \\
&\quad \}
\end{aligned}$$

Part 5: Temporal logic

Problem 1

Translate the following formula into English:

$$\exists\, x : \mathit{Person} \mid (\mathit{philosopher}(x) \land \Diamond \mathit{king}(x))$$

Problem 1: Solution

Translate the following formula into English:

$$\exists x : \mathit{Person} \mid (\mathit{philosopher}(x) \land \Diamond \mathit{king}(x))$$

The formula translates to:

"There is someone who is now a philosopher and will be a king at some time."

Problem 2

Translate the following formula into English:

$$\exists\, x : \mathit{Person} \mid \Diamond(\mathit{philosopher}(x) \land \mathit{king}(x))$$

Problem 2: Solution

Translate the following formula into English:

$$\exists\, x : \mathit{Person} \mid \Diamond(\mathit{philosopher}(x) \land \mathit{king}(x))$$

The formula translates to:

"There now exists someone who will at some time be both a philosopher and a king."

Problem 3

For a routine S, express the partial correctness triple in linear temporal logic.

Problem 3: Solution

The triple reads

$$\{P\}\, S\, \{Q\}$$

and it can be expressed in LTL as

$$P \rightarrow \Box(halts(S) \rightarrow Q)$$

which reads:
"If P holds at the initial state, then it is always the case that whenever S halts, then Q holds." (i.e. if the routine halts at any moment in time, then the postcondition will hold).

Problem 4

Describe and visualize the following formula:

$\Box(\phi \wedge \bigcirc\psi)$

Problem 4: Solution

$\Box(\phi \wedge \bigcirc\psi)$

$(\phi \wedge \bigcirc\psi)$ is globally true:

ϕ is true at every moment and ψ is also true at every next moment.

Problem 5

Describe and visualize the following formula:

$\phi \mathcal{U} (\phi \mathcal{U} \psi)$

Problem 5: Solution

$\phi \mathcal{U} (\phi \mathcal{U} \psi)$

- ϕ is true up and until (but not including) the moment when $(\phi \mathcal{U} \psi)$ becomes true.
- The parenthesized formula reads: ϕ is true up and until (but not including) the moment when ψ becomes true.

Problem 6

Describe and visualize the following formula:

$\phi \wedge \bigcirc(\psi \,\mathcal{U}\, (\psi \,\mathcal{U}\, \tau))$

Problem 6: Solution

$\phi \wedge \bigcirc(\psi \mathcal{U} (\psi \mathcal{U} \tau))$

- Both ϕ and $\bigcirc(\psi \mathcal{U} (\psi \mathcal{U} \tau))$ are true at time i.
- $(\psi \mathcal{U} (\psi \mathcal{U} \tau))$ is true at time $i + 1$:
  ψ is true up and until (but not including) the moment when $(\psi \mathcal{U} \tau)$ becomes true.

Problem 7

$\Box(\phi \vee \psi)$

Problem 7: Solution

$$\Box(\phi \vee \psi)$$

The parenthesized formula is globally true (i.e. at every moment).

Problem 8

$$\bigcirc\Box(\phi \oplus \psi)$$

Problem 8: Solution

$$\bigcirc\Box(\phi \oplus \psi)$$

The entire formula is true at time i.

This means that $\Box(\phi \oplus \psi)$ is true at time $i + 1$.

The parenthesized formula becomes true at time $i + 1$ and remains true subsequently.

Problem 9

$\bigcirc^2 \Diamond \Box \phi$

Problem 9: Solution

$$\bigcirc^2 \Diamond \Box \phi$$

Is the formula well-formed?

Yes, because it is interpreted as $\bigcirc^2(\Diamond(\Box(\phi)))$.

or

$\bigcirc^2(\textit{formula})$.

Problem 9: Solution /cont.

$$\bigcirc^2 \Diamond \Box \phi$$

The entire formula is true at time i.

This means that $\Diamond \Box \phi$ is true at time $i + 2$.

This implies that at some moment in time starting from $i + 2$, ϕ becomes true and stays true subsequently.

Problem 10

$\neg \Diamond \Box \phi$

Problem 10: Solution

$$\neg\Diamond\Box\phi$$

The entire formula is true at time $= i$.

The formula is interpreted as "It is not the case that at some moment in time ϕ becomes true and stays true subsequently."

The interpretation can be rephrased as "Once ϕ becomes true, it will not remain true for ever."

Problem 11

Translate the following requirements into temporal logic expressions:

1. Once a request is made, it will remain registered at least until the request is answered.
2. Once a process requests a resource, it will be inactive until it receives a go message.

Problem 11: Solution

Translate the following requirements into temporal logic expressions:

1. Once a request is made, it will remain registered at least until the request is answered.

$$\Box(\mathit{request\_made} \rightarrow (\mathit{request\_registered}\,\mathcal{U}\ \mathit{request\_answered}))$$

2. Once a process requests a resource, it will be inactive until it receives a go message.

$$\Box(\mathit{sent}(\mathit{request}) \rightarrow (\mathit{inactive}(P)\,\mathcal{U}\ \mathit{received}(\mathit{go})))$$

Problem 12

Consider the following requirements for a library system:

1. Administrators operate in self-exclusion for a write operation, i.e. only one administrator may be modifying the library catalog at any moment in time:
2. Administrators and clients operate in mutual exclusion:

Problem 12: Solution

1. Administrators operate in self-exclusion for a write operation, i.e. only one administrator may be modifying the library catalog at any moment in time:

$$\forall\, i, j : \mathit{Administrator}$$
$$(\Box\neg(\mathit{modification}(\mathit{admin}_i) \wedge \mathit{modification}(\mathit{admin}_j)))$$
$$\mathit{for}\ i \neq j.$$

2. Administrators and clients operate in mutual exclusion:

$$\Box(\neg \mathit{active}(\mathit{admin}) \vee \neg \mathit{active}(\mathit{client}))$$

Problem 13

In the previous example (for requirement 2), would

$$\Box(active(admin) \oplus active(client))$$

be a correct expression to capture mutual exclusion?

Problem 13: Solution

No, because this would enforce that always one must be active, whereas we need to allow the possibility for a moment in time that none is active.

Problem 14

Translate the following into formal logic:

"Once red, the light cannot become green immediately."

Problem 14: Solution

Translate the following into formal logic:

"Once red, the light cannot become green immediately."

$$\Box(\mathit{red} \rightarrow \neg \bigcirc \mathit{green})$$

Problem 15

Translate the following into formal logic:

"Once red, the light always becomes green eventually after being yellow for some time."

(Hint: Enforce some duration for both red and yellow before reaching green.)

Problem 15: Solution

Translate the following into formal logic:

"Once red, the light always becomes green eventually after being yellow for some time."

$$\Box(\mathit{red} \rightarrow \bigcirc(\mathit{red}\,\mathcal{U}\,(\mathit{yellow} \wedge \bigcirc(\mathit{yellow}\,\mathcal{U}\,\mathit{green}))))$$

Problem 16

Given the following statements that constitute system requirements:

1. $\Box(\mathit{request} \rightarrow \Diamond \mathit{acknowledgment})$
2. $\Box(\mathit{acknowledgment} \rightarrow \bigcirc \mathit{processing})$
3. $\Box(\mathit{processing} \rightarrow \Diamond\Box \mathit{done})$

How would you translate the following statement in English

$$\Box \mathit{request} \wedge \Box \neg \mathit{done}$$

and what can you infer about it?

Problem 16: Solution

The statement translates to “The system continuously sends a request but never sees it completed” and it is inconsistent with the set of requirements.

Problem 17

Translate the following into formal statements and indicate the corresponding property (fairness, liveness, safety):

- "Printing for processes `a` and `b` can never occur simultaneously."
- "Eventually, printing will be allowed for some process."
- "If a process makes a print request infinitely often, then printing for that process will occur infinitely often."

Problem 17: Solution

- (**Safety**) "Printing for processes `a` and `b` can never occur simultaneously."

  $$\Box\neg(\mathit{printing}(a) \wedge \mathit{printing}(b))$$

- (**Liveness**) "Eventually, printing will be allowed for some process."

  $$\Diamond(\exists\, x : \mathit{Process} \mid \mathit{printing}(x))$$

- (**Fairness**) "If a process makes a print request infinitely often, then printing for that process will occur infinitely often."

  $$\forall\, y : \mathit{Process}\ (\Box\Diamond \mathit{print\_request}(y) \to \Box\Diamond \mathit{printing}(y))$$

Problem 18

Let p and q be propositions. The expression $p \rightarrow \Diamond q$ maps to which property below:

Guarantee	**Response**	**Precedence**

Problem 18: Solution

Let p and q be propositions. The expression $p \rightarrow \Diamond q$ maps to which property below:

Guarantee	**Response**	**Precedence**
$\times$	$\checkmark$	$\times$

Problem 19

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"Once red, the light will never become immediately green"			

Problem 19: Solution

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"Once red, the light will never become immediately green"	✓	×	×

Problem 20

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"If the light is red infinitely often, it should be yellow infinitely often."			

Problem 20: Solution

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"If the light is red infinitely often, it should be yellow infinitely often."	$\times$	$\times$	$\checkmark$

Problem 21

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"Once red, the light becomes green eventually.'			

Problem 21: Solution

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"Once red, the light becomes green eventually.'	×	✓	×

Problem 22

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"The resource must not be simultaneously accessed by a writer and a reader.'			

Problem 22: Solution

The requirement below corresponds to which property:

	Safety	**Liveness**	**Fairness**
"The resource must not be simultaneously accessed by a writer and a reader.'	✓	×	×

Problem 23

Select the appropriate temporal logic formula for "The boiler controller system is deadlock free."

$\Box\Diamond\neg deadlock$	$\Diamond\neg deadlock$	$\Box\neg deadlock$	$\Diamond\Box\neg deadlock$

Problem 23: Solution

Select the appropriate temporal logic formula for "The boiler controller system is deadlock free."

$\Box\Diamond\neg \mathit{deadlock}$	$\Diamond\neg \mathit{deadlock}$	$\Box\neg \mathit{deadlock}$	$\Diamond\Box\neg \mathit{deadlock}$
×	×	✓	×

Problem 24

What pattern of behavior does the following temporal formula specify?

$$\square \begin{bmatrix} \textbf{start} \rightarrow \mathit{req\_jack} \\ \textbf{start} \rightarrow \neg \mathit{wait} \\ \mathit{req\_jack} \rightarrow \bigcirc \mathit{wait} \\ \mathit{req\_jack} \rightarrow \bigcirc \neg \mathit{req\_jack} \\ \mathit{wait} \rightarrow \bigcirc \mathit{req\_jack} \\ \mathit{wait} \rightarrow \bigcirc \neg \mathit{wait} \end{bmatrix}$$

Problem 24: Solution

The program reproduces indefinitely the sequence

$$\langle (\mathit{req\_jack} \wedge \neg \mathit{wait}), (\neg \mathit{req\_jack} \wedge \mathit{wait}) \rangle$$

Part 6: Algebraic specifications

Problem 1

- For the Set ADT whose operations are shown below

$$
\begin{aligned}
&\text{newset} \to \text{Set};\\
&\text{add} : \text{Set} \times \text{Element} \to \text{Set};\\
&\text{remove} : \text{Set} \times \text{Element} \to \text{Set};\\
&\text{size} : \text{Set} \to \mathbb{N}_0;\\
&\text{isempty} : \text{Set} \to \text{Boolean};\\
&\text{ismember} : \text{Set} \times \text{Element} \to \text{Boolean};
\end{aligned}
$$

- and for variables `s: Set; x, y: Element`, produce axioms to demonstrate the following properties of operation `add`:

1. Commutativity, and
2. Idempotence.

Problem 1 /cont.

- **Commutativity** is a property of a binary operation whereby changing the order of the operands does not change the result.

- **Idempotence** is the property of certain operations whereby they can be applied multiple times without changing the result beyond the initial application.

Problem 1: Solution

- For the Set ADT

$$
\begin{aligned}
&\text{newset} \to \text{Set};\\
&\text{add} : \text{Set} \times \text{Element} \to \text{Set};\\
&\text{remove} : \text{Set} \times \text{Element} \to \text{Set};\\
&\text{size} : \text{Set} \to \mathbb{N}_0;\\
&\text{isempty} : \text{Set} \to \text{Boolean};\\
&\text{ismember} : \text{Set} \times \text{Element} \to \text{Boolean};
\end{aligned}
$$

- The commutativity property of operation `add` can be captured by
  `add(add(s, x), y) = add(add(s, y), x);`

- and its idempotence property can be captured by
  `add(add(s, x), x) = add(s, x);`