Section 1: Multiple Choice. Choose exactly one answer.

Exercise 1: Formal Methods (2 credits)
Select the right answer to the following question (choose exactly one answer).
Which of the following best describes the reasons for using formal verification in a software project?

1. Formal verification is a straightforward and cost-effective approach to software quality
2. Formal verification allows to determine the particular test cases that need to be executed to guarantee program correctness
3. Formal verification allows to prove that the software performs according to a previously agreed specification
4. It is easier to explain the benefits of formal verification to management than it is to explain the benefits of ad hoc testing

Answer: 3

Exercise 2: FOL Specification (2+2 credits)
For each of the questions below, select the correct translation into first-order logic (choose exactly one answer).
Use the predicates $O(x)$ - $x$ is fully operational, $L(x)$ - $x$ needs emergency landing, $R(x)$ - $x$ is running, $B(x, y)$ - $x$ belongs to $y$, to translate the following sentences into first-order logic. Which translation is correct?

1. An aircraft is fully operational if all its engines are running.
   (a) $\forall a. (\forall e. B(e, a) \land R(e)) \Rightarrow O(a)$
   (b) $\forall a. (\forall e. B(a, e) \land R(e)) \Rightarrow O(a)$
   (c) $\forall a. (\exists e. B(a, e) \Rightarrow R(e)) \Rightarrow O(a)$
   (d) $\forall a. (\forall e. B(e, a) \Rightarrow R(e)) \Rightarrow O(a)$

   Answer: 1d

2. An aircraft needs emergency landing unless all its engines are running.
   (a) $\forall a. (\forall e. B(e, a) \Rightarrow R(e)) \Rightarrow L(a)$
   (b) $\forall a. (\forall e. B(e, a) \Rightarrow R(e)) \Rightarrow L(a)$
   (c) $\forall a. L(a) \Rightarrow (\forall e. B(a, e) \Rightarrow R(e))$
   (d) $\forall a. (\neg (\forall e. B(e, a) \Rightarrow R(e))) \lor L(a)$
Exercise 3

Structural Induction

Given the following function definition:

\[ (++ : [a] -> [a] -> [a] \]
\[ [\cdot] ++ ys = ys \quad \text{-- A1} \]
\[ (x:xs) ++ ys = x : (xs ++ ys) \quad \text{-- A2} \]

Suppose you have been asked to prove

\[ xs ++ (ys ++ zs) = (xs ++ ys) ++ zs \]

What do you need to prove for the base case and step case respectively? Select exactly one option.

1. Base case: \[ [] ++ (ys ++ zs) = ([] ++ ys) ++ zs \]
   Step case: Assume \[ as ++ (ys ++ zs) = (as ++ ys) ++ zs \]
   Prove \[ (a:as) ++ (ys ++ zs) = ((a:as) ++ ys) ++ zs \]

2. Base case: \[ xs ++ ([] ++ zs) = (xs ++ []) ++ zs \]
   Step case: Assume \[ as ++ (ys ++ zs) = (as ++ ys) ++ zs \]
   Prove \[ (a:as) ++ (ys ++ zs) = ((a:as) ++ ys) ++ zs \]

3. Base case: \[ [] ++ (ys ++ zs) = ([] ++ ys) ++ zs \]
   Step case: Assume \[ xs ++ (bs ++ zs) = (xs ++ bs) ++ zs \]
   Prove \[ xs ++ ((b:bs) ++ zs) = (xs ++ (b:bs)) ++ zs \]

4. Base case: \[ [] ++ ([] ++ []) = ([] ++ []) ++ [] \]
   Step case: Assume \[ as ++ (ys ++ zs) = (as ++ ys) ++ zs \]
   Prove \[ (a:as) ++ (ys ++ zs) = ((a:as) ++ ys) ++ zs \]

Answer: 1

Exercise 4

Hoare Logic

Consider the following Hoare triples:

(I) \{ x = 2 \} x := y + 1 \{ y = 1 \}

(II) \{ y = y + 1 \} x := y + 1 \{ y = x \}

Which of these triples is/are valid? Select exactly one answer.

(a) I only
(b) II only
(c) Both
(d) Neither

Solution
b.

Exercise 5

Hoare Logic

Consider the following Hoare triples:

(I) \{ true \} x := y + 1 \{ false \}

(II) \{ true \} x := y + 1 \{ z = 1 \}

Which of these triples is/are valid? Select exactly one answer.

Answer: b
(a) I only  (b) II only  (c) Both  (d) Neither

Solution
d.

Exercise 6  Hoare Logic  (2 credits)
Consider the following Hoare triples:

(I) \{ x = y \} \text{ if } (x = 0) \text{ then } x := y + 1 \text{ else } z := y + 1 \{ (x = y + 1) \lor (z = x + 1) \}

(II) \{ x = y \} \text{ if } (x = 0) \text{ then } x := y + 1 \text{ else } z := y + 1 \{ (z = 1) \rightarrow (x = 1) \}

Which of these triples is/are valid? Select exactly one answer.

(a) I only  (b) II only  (c) Both  (d) Neither

Solution  c.

Exercise 7  Non-Empty Sets in $\mathbb{Z}$  (2 credits)
Consider the following $\mathbb{Z}$ schema that specifies a non-empty set of integers and the operations of adding and removing elements.

$$
\begin{array}{c}
\text{NonEmpty} \\
S : \mathbb{P} \mathbb{Z} \\
S \neq \emptyset
\end{array}
\quad
\begin{array}{c}
\text{Add} \\
\Delta \text{NonEmpty} \\
i? : \mathbb{Z} \\
S = S' \cup \{i\}
\end{array}
\quad
\begin{array}{c}
\text{Remove} \\
\Delta \text{NonEmpty} \\
i? : \mathbb{Z} \\
S = S' \setminus \{i?\}
\end{array}
$$

Which of the above operation schemas are robust? Select exactly one answer.

(a) Add only  (b) Remove only  (c) Both  (d) Neither

Solution. c. Note if you read the constraint as $S' = S \cup \{i\}$ and $S' = S \setminus \{i\}$ then only $\text{Add}$ is robust. We have awarded full marks for both answers.

Exercise 8  Sets not containing zero  (2 credits)
Now consider a variant of the above schema that asserts that 0 is not contained in a set of integers.

$$
\begin{array}{c}
\text{NoZero} \\
S : \mathbb{P} \mathbb{Z} \\
0 \notin S
\end{array}
\quad
\begin{array}{c}
\text{Add} \\
\Delta \text{NoZero} \\
i? : \mathbb{Z} \\
S = S' \cup \{i?\}
\end{array}
\quad
\begin{array}{c}
\text{Remove} \\
\Delta \text{NoZero} \\
i? : \mathbb{Z} \\
S = S' \setminus \{i?\}
\end{array}
$$

Which of the above operation schemas are robust? Select exactly one answer.

(a) Add only  (b) Remove only  (c) Both  (d) Neither

Solution. c. Note if you read the constraint as $S' = S \cup \{i\}$ and $S' = S \setminus \{i\}$ then only $\text{Remove}$ is robust. We have awarded full marks for both answers.

Section 2: Free-Form Questions

Exercise 9  Natural Deduction  (10 credits)
Prove the correctness of the following derived rules using natural deduction. Do not use algebraic laws, or any of the derived rules obtained in lectures.

\[
\frac{r \Rightarrow s}{\neg(r \land \neg s)} \quad \frac{(\forall x. S(x)) \lor (\forall y. T(y))}{\forall z. (S(z) \lor T(z))}
\]

where $z$ does not occur free in $S$ or $T$ in the rule on the right.
The first rule.

1. \( r \Rightarrow s \)
2. \( r \land \neg s \)
3. \( r \quad \land\text{-E, } 2 \)
4. \( \neg s \quad \land\text{-E, } 2 \)
5. \( s \quad \rightarrow\text{-E, } 1, 3 \)
6. \( s \land \neg s \quad \land\text{-I, } 5, 4 \)
7. \( \neg(r \land \neg s) \quad \neg\text{-I, } 2–6 \)

The second rule.

1. \( (\forall x.S(x)) \lor (\forall y.T(y)) \)
2. \( \forall x.S(x) \)
3. \( a \quad S(a) \quad \forall\text{-E, } 2 \)
4. \( S(a) \lor T(a) \quad \forall\text{-I, } 3 \)
5. \( \forall z.(S(z) \lor T(z)) \quad \forall\text{-I, } 4 \)
6. \( \forall y.T(y) \)
7. \( a \quad T(a) \quad \forall\text{-E, } 2 \)
8. \( S(a) \lor T(a) \quad \forall\text{-I, } 7 \)
9. \( \forall z.(S(z) \lor T(z)) \quad \forall\text{-I, } 8 \)
10. \( \forall z.(S(z) \lor T(z)) \quad \forall\text{-E, } 1, 2–5, 6–9 \)

Exercise 10 (6 credits)

Give a proof of the following Hoare triple on the left using the while-rule for total correctness on the right.

\[
[x > 0] \text{ while } (x \neq 0) \text{ do } x := x - 1 [\text{true}]
\]

\[
\begin{array}{c}
P \land b \Rightarrow E \geq 0 \\
[P \land b \land (E = n)] S [P \land (E < n)]
\end{array}
\]

\[
[P] \text{ while } b \text{ do } S [P \land \neg b]
\]

Hint: The invariant \( P \) required by the while-rule is not necessarily the precondition of the Hoare triple in the question.

Solution

(In this proof \( P \equiv x \geq 0 \) and \( E \equiv x \).)

1. \( x \geq 0 \land x \neq 0 \Rightarrow x \geq 0 \quad \text{(Trivial Fact)} \)
2. \( [x - 1 \geq 0 \land x - 1 < n] x := x - 1 [x \geq 0 \land x < n] \quad \text{(Assignment)} \)
3. \( [x \geq 0 \land x \neq 0 \land x = n] x := x - 1 [x \geq 0 \land x < n] \quad \text{(2, Precondition Strengthening)} \)
4. \( [x \geq 0] \text{ while } (x \neq 0) \text{ do } x := x - 1 [x \geq 0 \land \neg(x \neq 0)] \quad (1, 3, \text{While}) \)
5. \( [x \geq 0] \text{ while } (x \neq 0) \text{ do } x := x - 1 [\text{true}] \quad (4, \text{Postcondition Weakening}) \)
6. \( [x > 0] \text{ while } (x \neq 0) \text{ do } x := x - 1 [\text{true}] \quad (5, \text{Precondition Strengthening}) \)

Marking guide

If the proof is not completely correct, give 1 point for each correct and relevant step, and cap the total mark at 5.
It's acceptable to merge steps 5 and 6 into a single step. If that happens, regard it as two steps for the purpose of giving partial credit.

In step 4, it's acceptable to write \( x = 0 \) in lieu of \( \neg(x \neq 0) \), or in lieu of the whole postcondition.

**Exercise 11**

**Scaling of Integer Sets**

(2+2 credits)

Consider the \( \mathbb{Z} \)-schema of non-empty sets of integers and the following two incomplete operation specifications below.

<table>
<thead>
<tr>
<th>( \text{NonEmpty} )</th>
<th>( \Delta \text{NonEmpty} )</th>
<th>( \Xi \text{NonEmpty} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S : \mathbb{P} \mathbb{Z} )</td>
<td>( s? : \mathbb{Z} )</td>
<td>( o! : \mathbb{B} )</td>
</tr>
<tr>
<td>( S \neq \emptyset )</td>
<td>( \forall i : \mathbb{Z} \bullet i \in S' \iff )</td>
<td>( o! \iff )</td>
</tr>
</tbody>
</table>

where \( \mathbb{B} = \{\text{true, false}\} \) is the set of booleans.

1. Complete the constraint in \( \text{Scale} \) so that \( \text{Scale} \) specifies the scaling of a set \( S \) of integers by an integer \( i \), i.e. such that \( S' \) contains precisely all products \( s? \cdot i \) where \( i \in S \).

**Solution.**

One way to complete the constraint is:

\[
\forall i : \mathbb{Z} \bullet i \in S' \iff \exists n \in S \bullet i = n \cdot s?
\]

2. Complete the constraint in \( \text{Bounded} \) so that the value of the output variable \( o! \) is true if and only if the set \( S \) is bounded, i.e. there are integers \( l \) and \( h \) such that all elements \( i \in S \) of \( S \) lie between \( l \) and \( h \).

One way to complete the constraint is:

\[
o! \iff \phi \land \psi
\]

where

\[
\phi \equiv \exists l : \mathbb{Z} \bullet \forall n : \mathbb{Z} \bullet n \in S \rightarrow l \leq n
\]

and

\[
\psi \equiv \exists h : \mathbb{Z} \bullet \forall n : \mathbb{Z} \bullet n \in S \rightarrow n \leq h
\]