INSTRUCTOR: Aaron Williams

DURATION: 3 Hours

INSTRUCTIONS

- Questions are to be answered on the examination paper.
- THIS EXAMINATION HAS 12 PAGES. Students must count the number of pages in this examination paper before beginning to write, and report any discrepancy immediately to the invigilator.
- The examination is closed book, and you may not use any printed or electronic materials, cell phones, PDAs, laptops, or calculators.

<table>
<thead>
<tr>
<th>Question</th>
<th>Total marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>

|          | 80          |
1. (6 marks) Below is a reduced ordered binary decision diagram. Write down the BDD that results from applying the \( \text{RESTRICT}(B, b, \text{False}) \) algorithm. That is, write down the BDD for \( A|_{b=\text{False}} \), where \( A \) is the formula that \( B \) represents.

\[ B = \]

2. (6 marks) When \( B \) is a reduced ordered BDD, is it always true that \( \text{RESTRICT}(B, p, \alpha) \) results in a reduced ordered BDD, for any proposition \( p \) and \( \alpha \in \{\text{True, False}\} \)? If it is true, then justify your answer with an informal argument (a formal proof is not required). Otherwise, show a reduced ordered BDD and a restriction of it that is not a reduced ordered BDD.

\[ B = \]

becomes

when \( \text{RESTRICT}(B, b, \text{True}) \) is done.

However, this is not a reduced BDD since it is equivalent to

Note: Another nice answer is \( \text{RESTRICT}(B, d, \text{False}) \) using the BDD \( B \) from the previous question.
3. (6 marks) Write Prolog code that removes every copy of a specified value from a list. In other words, implement `removeX(A,X,B)` so that `B` is the list that results from removing every copy of `X` from list `A`, whenever `X` and `A` are given as constants. For example, the query `removeX([1 2 3 4 3 2 1],3,B)` should succeed with the unique binding `B = [1 2 4 2 1].`

```
removeX([], X, [X]).
removeX([H|T], X, B) :-
    H is X ->
        removeX(T, X, B),
    removeX(T, X, B1),
    append([H], B1, B).
```

4. (6 marks) This question tests your knowledge of unification in Prolog. Decide whether each of the following queries succeeds or fails. If the query succeeds, then provide a suitable list of bindings. Otherwise, simply state “no”.

a) `a = b.`
   
   no

b) `3 = X.`
   
   X = 3

c) `a(c,X,X) = a(Y,Y,b).`
   
   no

d) `X = Y, a(Z) = a(Y), X = hello.`
   
   X = hello
   Y = hello
   Z = hello

e) `a(X) = a(b,c).`
   
   no

f) `a(b,X) = a(b,c(Y,e)), X = c(hello, e).`
   
   X = c(hello, e)
   Y = hello
5. For every formula $A$, exactly one of the following three statements is true:

S1) $A$ is unsatisfiable
S2) $A$ is unsatisifiable
S3) $A$ is satisfiable and $A$ is falsifiable.

Clearly state which statement is true for the given formula and then prove your result using semantic tableau.

a) (4 marks) $((p \lor q) \rightarrow r) \lor (p \land \neg r)$

$\neg \Sigma$$\neg \Sigma$$\neg \Sigma$

Therefore the formula is unsatisfiable.

---

**Common Error:** You cannot prove that a formula $A$ is unsatisfiable by using a semantic tableau for $A$. _example_ $p \lor q$

$p$  
$q$  

This does not prove $p \lor q$ is unsatisfiable and in fact it is falsifiable.

Therefore, $p \lor q$ is satisfiable (but not unsatisfiable).
b) (6 marks) \((\forall x p(x) \land \exists x \neg p(x)) \lor \exists x p(x)\)

\[ (\forall x p(x) \land \exists x \neg p(x)) \lor \exists x p(x) \]

\[
\begin{array}{c}
\forall x p(x) \land \exists x \neg p(x) \\
\forall x p(x), \exists x \neg p(x) \\
\forall x p(x), \neg p(a) \\
p(a), \neg p(a) \\
\neg(p(a), \neg p(a)) \\
\forall \neg \neg(p(a), \neg p(a)) \\
\end{array}
\]

Therefore, the formula is satisfiable.

\[
\begin{array}{c}
\exists x p(x) \\
\exists x \neg p(x) \\
\exists x \neg p(x), \exists x p(x) \\
\exists x \neg p(x), \exists x p(x) \\
\exists x \neg p(x), \exists x p(x), \forall \neg \neg(p(a), \neg p(a)) \\
\end{array}
\]

But \((\forall x p(x) \lor \exists x \neg p(x)) \lor \exists x p(x)\) is not falsifiable.

Therefore, the formula is falsifiable.
6. For every pair of formulas $A$ and $B$, exactly one of the following four statements is true:

P1) $A \iff B$ and $B \iff A$

P2) $A \iff B$ and $B \not\iff A$

P3) $A \not\iff B$ and $B \iff A$

P4) $A \not\iff B$ and $B \not\iff A$

Clearly state which statement is true for the given pair of formulas and then prove your result. You may use any technique to determine your answer, however, your proofs must be of the following format.

- To prove $A \iff B$ (or $B \iff A$) show $A \equiv \ldots \equiv B$ where each formula follows logically from the previous one.
- To prove $A \not\iff B$ (or $B \not\iff A$) specify an appropriate assignment or interpretation.

In the case of P1), you may prove both directions at once by showing $A \equiv \ldots \equiv B$.

a) (4 marks) $A = (p \oplus q) \oplus r$ and $B = (p \leftrightarrow q) \leftrightarrow r$

\[
\begin{align*}
A = B \text{ and } B \equiv A \text{ (in other words, } A &\equiv B) \\
(p \oplus q) \oplus r &\equiv \neg \neg ((p \oplus q) \oplus r) \\
&\equiv \neg \neg ((p \oplus q) \oplus r) \\
&\equiv \neg \neg ((p \oplus q) \leftrightarrow r) \\
&\equiv ((p \oplus q) \lor r) \lor ((p \oplus q) \land r) \\
&\equiv (p \leftrightarrow q) \lor r \\
\end{align*}
\]

Note: you can use a truth table to determine this answer

\[
\begin{array}{c|c|c|c|c|c}
  p & q & r & (p \oplus q) \oplus r & (p \leftrightarrow q) \leftrightarrow r \\
  \hline
  T & T & T & T & T \\
  T & T & F & F & F \\
  T & F & T & F & F \\
  T & F & F & T & T \\
  F & T & T & F & F \\
  F & T & F & T & T \\
  F & F & T & F & F \\
  F & F & F & T & F \\
\end{array}
\]
b) (6 marks) \( A = \forall x A(x) \lor \exists x B(x) \) and \( B = \forall x (A(x) \lor B(x)) \)

P3) \( A \neq B \) and \( B = A \)

First let's prove \( A \neq B \)

Consider the interpretation \( (\mathbb{Z}^+, \text{is Even}, \text{is Prime} 3, \emptyset) \).

Since there exists a prime integer, \( A \) is true in this interpretation.

Since 9 is an integer that is neither even nor prime, \( B \) is false in this interpretation.

Therefore, \( A \neq B \).

Next let's prove \( B = A \)

\[ \forall x (A(x) \lor B(x)) \equiv \forall x (\neg A(x) \rightarrow B(x)) \]
\[ \equiv \exists x \neg A(x) \rightarrow \exists x B(x) \]
\[ \equiv \forall x \neg A(x) \rightarrow \exists x B(x) \]
\[ \equiv \neg \forall x A(x) \lor \exists x B(x) \]
\[ \equiv \forall x A(x) \lor \exists x B(x) \].
7. Compute the weakest precondition. Show your work and simplify your answer as much as possible. In particular, your answer should not include arithmetic symbols or unnecessary conditions.

a) (4 marks) \( \text{wp}(x := x*y, x := x+y, x = y) \)

\[
\begin{align*}
\text{wp}(x := x*y, \ \text{wp}(x := x+y, x = y)) \\
\text{wp}(x := x*y, \ x+y = y) \\
(x*y + y = y) \\
(x*y = 0) \\
(x = 0 \lor y = 0)
\end{align*}
\]

b) (4 marks) \( \text{wp}(\text{if } x > y \text{ then } x := x+2 \text{ else } y := y/2 \text{ end, } x = y) \)

\[
\begin{align*}
(x > y) \Rightarrow \text{wp}(x := x+2, x = y) \land ((x \leq y) \Rightarrow \text{wp}(y := y/2, x = y)) \\
(x > y) \Rightarrow x + 2 = y \land (x \leq y) \Rightarrow x = y/2 \\
(x > y) \Rightarrow 3x = y \land \neg (x > y) \Rightarrow 2x = y \\
3x = y
\end{align*}
\]

8. (Space is provided on the next two pages to answer this question.) Consider the following two programs, and determine if they are partially correct with respect to their pre- and post-conditions.

\[
\begin{array}{l}
\{n > 0\} \\
f := n \\
i := 2 \\
\text{while } i <> n \text{ do} \\
f := f \cdot i \ \\
i := i + 1 \\
\text{end while} \\
\{f = n!\}
\end{array}
\begin{array}{l}
\{n > 0\} \\
a := 0 \\
b := 1 \\
i := 1 \\
\text{while } i < n \text{ do} \\
b := b + a \\
a := b - a \\
i := i + 1 \\
\text{end while} \\
\{b = F(n)\}
\end{array}
\]

If a program is partially correct then prove it by inserting appropriate mid-conditions. Otherwise, simply provide an appropriate input and output pair that shows the program is not partially correct. Your answers should be written on the next two pages.

\[
\text{Invariant: } f = \prod_{j=1}^{i-1} j = n \cdot (i-1)!
\]

\[
\text{Invariant: } b = F(i) \land a = F(i-1) \land i \leq n
\]
a) (6 marks) The purpose of this program is to compute $n!$, the $n$th factorial, where $n! = \prod_{i=1}^{n} i$ for all $n > 0$.

\[
\{ n > 0 \} \\
\{ n := n \} \\
\{ n := n \prod_{j=1}^{n-1} j \} \\
f := n \\
\{ f := n \prod_{j=1}^{i} j \} \\
i := 2
\]

while $i < n$ do

\[
\{ f := n \prod_{j=1}^{i-1} j \land i \neq n \} \\
\{ f := n \prod_{j=1}^{i} j \} \\
i := i + 1
\]

end while

\[
\{ f := n \prod_{j=1}^{i-1} j \land i \neq n \} \\
\{ f := n \prod_{j=1}^{i} j \} \\
i := i + 1
\]

end while

\[
\{ f := n \prod_{j=1}^{i-1} j \} \\
\{ f := n \prod_{j=1}^{i} j \} \\
\{ f := n \prod_{j=1}^{i} j \} \\
\{ i = n \} \\
\{ f := n \prod_{j=1}^{i} j \} \\
\{ f := n \prod_{j=1}^{i} j \} \\
\{ f = n! \}
\]

Loop invariant: $f := n \prod_{j=1}^{i-1} j$

this form makes for a nicer looking solution (ops).

Trace for $n = 5$

\[
\begin{array}{c|c|c|c}
\text{loop} & \text{counter} & \text{value} & \text{product} \\
\hline
1 & 5 & 2 & 5 \\
2 & 5 & 3 & 5 \cdot 2 \\
3 & 5 & 4 & 5 \cdot 2 \cdot 3 \\
4 & 5 & 5 & 5 \cdot 2 \cdot 3 \cdot 4 \\
5 & 5 & 6 & 5 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \\
\end{array}
\]

So the program is correct for $n = 5$
b) (6 marks) The purpose of this program is to compute \( F(n) \), the \( n \)th Fibonacci number, where \( F(0) = 0, F(1) = 1 \), and \( F(n) = F(n-1) + F(n-2) \) for all \( n > 2 \).

\[
\{ n > 0 \} \\
\{ n \geq 0 \} \\
\{ 1 = F(1) \land 0 = F(0) \land 0 \leq n \} \\
a := 0 \\
\{ 1 = F(1) \land a = F(0) \land 0 \leq n \} \\
b := 1 \\
\{ b = F(1) \land a = F(0) \land 0 \leq n \} \\
i := 1 \\
\begin{align*}
\{ b = F(i) \land a = F(i-1) \land i \leq n \} \\
\text{while } i < n \text{ do} \\
\{ b = F(i) \land a = F(i-1) \land i \leq n \} \\
\{ a = F(i-1) \land b = F(i) \land i+1 \leq n \} \\
\{ m = F(i+1) \cdot F(i) \land b = F(i) \land i+1 \leq n \} \\
\{ a = F(i+1) - b \land b = F(i) \land i+1 \leq n \} \\
\{ b + a = F(i+1) \land b + a = F(i) \land i+1 \leq n \} \\
b := b + a \\
\{ b = F(i+1) \land b - a = F(i) \land i+1 \leq n \} \\
a := b - a \\
\{ b = F(i+1) \land a = F(i) \land i+1 \leq n \} \\
i := i + 1 \\
\end{align*}
\{ b = F(i) \land a = F(i-1) \land i \leq n \} \\
\{ b = F(i) \land a = F(i) \land i \leq n \land i < n \} \\
\{ b = F(i) \land i = n \} \\
\{ b = F(n) \}
9. (8 marks) State a loop invariant that would allow you to prove that the following program is partially correct, with respect to the given pre- and post-conditions. You do not need to prove that the program is partially correct. Note: The symbol \( x | y \) means divides, so \( x | y \) means that there exists an integer \( k \) such that \( y = kx \).

\[
\begin{align*}
\{ x > 0 \land y > 0 \} \\
& r := x \\
& d := 0 \\
while & \ r \geq y \ do \\
& r := r - y \\
& d := d + 1 \\
end \ while \\
& \{ r = 0 \iff y | x \} \\
\end{align*}
\]

\[
\begin{array}{c|c}
 r & d \\
\hline
 6 & 0 \\
 3 & 1 \\
 0 & 2 \\
\end{array}
\]

\[
\begin{array}{c|c}
 r & d \\
\hline
 13 & 0 \\
 9 & 1 \\
 5 & 2 \\
 1 & 3 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
 r & d & r \iff 0 \land y | x \\
\hline
 0 & 0 & \text{true} \\
 6 & 0 & \text{false} \\
 3 & 1 & \text{false} \\
 0 & 2 & \text{true} \\
\end{array}
\]

**Invariant:** \( r = x - dy \wedge r \geq 0 \)

\[
\begin{array}{c|c|c|c}
 i & j & \text{Variant Value} \\
\hline
 0 & 0 & 10 - 0 - 0 = 10 \\
 1 & 0 & 10 - 1 - 0 = 9 \\
 1 & 1 & 10 - 1 - 1 = 8 \\
 2 & 0 & 10 - 3 - 0 = 7 \\
 2 & 1 & 10 - 3 - 1 = 6 \\
 2 & 2 & 10 - 3 - 2 = 5 \\
 3 & 0 & 10 - 6 - 0 = 4 \\
 3 & 1 & 10 - 6 - 1 = 3 \\
 3 & 2 & 10 - 6 - 2 = 2 \\
 3 & 3 & 10 - 6 - 3 = 1 \\
 4 & 0 & 10 - 10 - 0 = 0 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c}
 i & j & \text{Variant Value} & \text{non-negative} & \text{integer-valued} \\
\hline
 0 & 0 & 10 - 0 - 0 = 10 & \text{non-negative} & \text{decreasing} \\
 1 & 0 & 10 - 1 - 0 = 9 & \text{non-negative} & \text{decreasing} \\
 1 & 1 & 10 - 1 - 1 = 8 & \text{non-negative} & \text{decreasing} \\
 2 & 0 & 10 - 3 - 0 = 7 & \text{non-negative} & \text{decreasing} \\
 2 & 1 & 10 - 3 - 1 = 6 & \text{non-negative} & \text{decreasing} \\
 2 & 2 & 10 - 3 - 2 = 5 & \text{non-negative} & \text{decreasing} \\
 3 & 0 & 10 - 6 - 0 = 4 & \text{non-negative} & \text{decreasing} \\
 3 & 1 & 10 - 6 - 1 = 3 & \text{non-negative} & \text{decreasing} \\
 3 & 2 & 10 - 6 - 2 = 2 & \text{non-negative} & \text{decreasing} \\
 3 & 3 & 10 - 6 - 3 = 1 & \text{non-negative} & \text{decreasing} \\
 4 & 0 & 10 - 10 - 0 = 0 & \text{non-negative} & \text{decreasing} \\
\end{array}
\]

**Invariant:** \( \frac{n \cdot (n+1)}{2} - \frac{i \cdot (i+1)}{2} - j \)

**Note:** There are simpler variants that work, such as \( n^2 - i^2 - j \).
(Hodja thought you might want an extra page.)

**WHY DID HE RIDE ON THE DONKEY BACKWARDS?**

One day when Hodja was going to the mosque with his students, he decided to ride on his donkey backwards. The students asked:

"Why are you riding on the donkey backwards? You must be very uncomfortable!"

He answered:

"If I sit facing forward, you would be behind me. If you were in front of me, I would be behind you. Either way, I would not be facing me. So, this is the most logical way!"

http://www.cs.biu.ac.il/~schiff/Net/

**END.**