Exercise 1. Consider the following UML class diagram.

1. Express it in FOL.
2. Express it in DL-Lite\(_A\), highlighting the parts that are not expressible.
3. Given the ABox \(A = \{A(c)\}\), check formally if the diagram, as expressed in \(DL-Lite_A\), is consistent with \(A\).

Exercise 2. Model check the Mu-Calculus formula \(\nu X.\mu Y.((a \land \langle next \rangle X) \lor \langle next \rangle Y)\) and the CTL formula \(AG(EX(EXa \lor EXEXa))\) (showing its translation in Mu-Calculus) against the following transition system:

Exercise 3. Consider the following predicates: \(Person(x)\), saying that \(x\) is a person, \(Appetizer(x)\), saying that \(x\) is a gourmet appetizer, \(MainCourse(x)\), saying that \(x\) is a gourmet main course, and \(likes(x,y)\), saying that \(x\) likes \(y\). Express in FOL the following boolean queries, stating which ones are CQs (do not use abbreviations for cardinalities):

1. There exists a person who likes an appetizer and a main course.
2. There exists a person who likes two appetizers and a main course.
3. There exists a person who likes exactly one main course.
4. There exists a person who likes all appetizers.
5. Return the persons who like no main courses.
6. Return the pairs of persons such that the first likes all appetizers that the second likes, but not viceversa.

Exercise 4. Compute the certain answers to the CQ \(q(x) \leftarrow Person(x), Person(y), Likes(x,z), Likes(z,y)\) over the following incomplete database (naive tables), and discuss how you obtained the result:

<table>
<thead>
<tr>
<th>Person name</th>
<th>Likes ls</th>
<th>Likes ld</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>Smith</td>
<td>null(_1)</td>
</tr>
<tr>
<td>null(_1)</td>
<td>Brown</td>
<td>null(_1)</td>
</tr>
<tr>
<td>Brown</td>
<td>Brown</td>
<td>null(_2)</td>
</tr>
<tr>
<td>null(_2)</td>
<td>Green</td>
<td>null(_2)</td>
</tr>
<tr>
<td>Green</td>
<td>White</td>
<td>null(_2)</td>
</tr>
<tr>
<td></td>
<td>White</td>
<td>null(_2)</td>
</tr>
</tbody>
</table>

Exercise 5. Compute the weakest precondition for getting \(\{x = 100\}\) by executing the following program:

\[
\begin{align*}
x &:= 90 - y; \\
\text{if } (x = 0) \text{ then } \\
\quad&\begin{cases} \\
\quad\text{if } (y > 10) \text{ then } \\
\quad\quad x := y - x; \\
\quad\quad\text{else } x := 10 - x \\
\quad\end{cases} \\
\quad&x := x + y; \\
y &:= 10 + y
\end{align*}
\]