Exercise 1 (SLD tree): (6 points)

Consider the following logic program $P$:

\[
\begin{align*}
\text{path}(X, X, Y) &. \\
\text{path}(X, Y, s(Z)) & :- \text{edge}(X, A), \text{path}(A, Y, Z). \\
\text{path}(X, Y, Z) & :- \text{eps}(X, A), \text{path}(A, Y, Z).
\end{align*}
\]

\[
\begin{align*}
\text{edge}(a, b). \\
\text{edge}(b, a). \\
\text{edge}(c, d). \\
\text{edge}(d, b). \\
\text{eps}(b, c).
\end{align*}
\]

The predicates $\text{edge}$ and $\text{eps}$ define the following graph $G$:

```
       a
       /\   \\
      /   \  \\
     b    c
      \   / \\
       v
      / \  \\
     d   
```

Furthermore, $\text{path}(X, Y, Z)$ is true if there is a path from $X$ to $Y$ in $G$ where at most $Z$ non-$\varepsilon$-edges are used along the path. As an example, $\text{?- path}(a, X, s(0))$ gives the solutions $X = a$, $X = b$, and $X = c$. Here, natural numbers are represented by the function symbols $0$ and $s$ (i.e., $s(0)$ stands for $1$, $s(s(0))$ stands for $2$, etc.)

Please give the SLD tree for the query $\text{?- path}(b, b, s(s(0)))$. Subtrees that Prolog explores after having found the second solution should be abbreviated with dots ($\ldots$).

Solution: ...
Exercise 2 (SLD tree): \((2+6 = 8\) points)

Consider the following logic program \(\mathcal{P}\):

\[
\begin{align*}
p(s(X), Y) & \leftarrow q(X, Y). \\
p(s(X), s(X)) & \leftarrow q(Y, s(0)), p(s(Y), s(0)), q(s(0), X). \\
p(s(s(X)), s(Y)) & \leftarrow p(s(X), Y). \\
p(s(0), 0). \\
q(s(0), \_). \\
q(X, s(X)) & \leftarrow p(X, X).
\end{align*}
\]

Below the corresponding SLD tree for the query \(p(s(0), X)\) is shown. Here, \(\square\) marks nodes that cannot be evaluated further. In this whole exercise, you should assume that Prolog uses “proper” unification (with occur check).
a) Give the answer substitutions in the order Prolog finds them. Explain your answer.

b) Change the order of exactly two literals in the clauses of \( P \) such that for the query from the first exercise part the resulting SLD tree is infinite. Give the resulting SLD tree, where infinite paths are marked with \( \infty \) and subtrees that Prolog explores after having found the second solution are abbreviated with dots (\( \ldots \)). Which solutions does Prolog find now?

Solution:

a) Prolog finds the first solutions from left to right (e.g. \( X/s(0) \), \( X/s(0) \), \( X/0 \)), because it traverses the tree with a depth-first search (from left to right).

b) \[
\begin{align*}
p(s(X),Y) &:- q(X,Y). \\
p(s(X),s(X)) &:- p(s(Y),Y), q(Y,s(0)), q(s(0),X). \\
p(s(s(X)),s(Y)) &:- p(s(X),Y). \\
p(s(0),0). \\
q(s(0),_). \\
q(X,s(X)) &:- p(X,X).
\end{align*}
\]

We initially assumed the following solution to be correct. However, the program DOES NOT find the first solution again. In fact it will be nonterminating after the first solution (The branch that originally was labeled \( \infty \) is indeed infinite and does not contain a solution).
Now Prolog will only find the first solution infinitely many times, because the path below the application of the second rule for $p$ will result in an endless recursion.

In a very strict sense the task is still solvable (by not abbreviating any subtree with ... as done above). We will accept any solution that contained the SLD tree above or some other sane SLD tree. Therefore, the real solution is (Note, that students were not supposed to draw a tree this big, it is only given for sake of correctness):
Prolog will now only find the first solution.