Problem 1:
?- 1+1=2.
No
?- 1+1=:=2.
Yes
?- X+1=2*Y+Z.
  X = 2*._G161
  Y = _G161
  Z = 1 ;
No
?- X=1;X=2.
  X = 1 ;
  X = 2 ;
No
?- [X,Y|Xs]=[a,b,c,d].
  X = a
  Y = b
  Xs = [c, d] ;
No
?- [X,[Y,Z],3]=[[Z,Y],X,Z].
  X = [3, 3]
  Y = 3
Z = 3 ;
No
?- member(b,[a,,b,c]).
ERROR: Syntax error: Operator expected
ERROR: member(b,[a,,
ERROR: ** here **
ERROR: b,c]) .

?- member(d,[a,b,c]).
No
?- member(X,[a,b,c]).
X = a ;
X = b ;
X = c ;
No
?- length(Xs,2),member(a,Xs),member(b,Xs).
Xs = [a, b] ;
Xs = [b, a] ;
No

**Problem 2:**

(a)

?- member(X,[1,222,3,56,670]),X>100.
\( X = 222 ; \)
\( X = 670 ; \)
No

(b) \(-\) member\((X, [1, 222, 3, 56, 670]), X > 50, X < 100.\)
\( X = 56 ; \)
No

(c) \(-\) append\((_, [X|Xs], [a, a, b, c, c]), \text{member}(X, Xs).\)
\( X = a \text{ } Xs = [a, b, c, c] ; \)
\( X = c \text{ } Xs = [c] ; \)
No

Problem 3:
\begin{align*}
\text{alike}(a, aa). & \quad (1) \\
\text{alike}(aa, aaa). & \quad (2) \\
\text{alike}(X, X). & \quad (3) \\
\text{alike}(X, Y):=\text{alike}(Y, X). & \quad (4) \\
\text{alike}(X, Y):=\text{alike}(X, Z), \text{alike}(Z, Y). & \quad (5)
\end{align*}

(a)

?- alike(a, aaa).

ERROR: Out of local stack Exception: (29,573)

alike(a, aaa) ? no debug
The response doesn’t reflect “logical conclusions”. According to logic, (1)(2)(5) ⇒ alike(a,aaa), But actually, clause (4) in the predicate definition prevents prolog to ever try clause (5) in this case. When trying to resolve alike(a,aaa) using clause (4), it tries to resolve alike(aaa,a). When then trying to resolve alike(aaa,a) using clause (4), it tries to resolve alike(a,aaa), therefore enter a loop.

Please see the following Prolog search tree for resolving the goal. Since the Prolog system is using depth-first search, the resolution has infinite depth, thus cause the out of stack error. (Refer to textbook Section 12.4.5 Prolog’s Search Strategy.)

(b)
?- alike(aaa,aa).

Yes

According to logic, (2)(4) ⇒ alike(aaa,aa).

(c)
?- alike(a,X).

X = aa ;
X = a ;

4
\[ X = \text{a} ; \]
\[ X = \text{aa} ; \]
\[ X = \text{a} ; \]
\[ X = \text{a} ; \]
\[ X = \text{aa} ; \]
\[ X = \text{a} ; \]
\[ X = \text{a} ; \]
\[ X = \text{aa} \]

(continuing...)

Yes

It doesn’t reflect logical conditions. According to logic, \text{a}, \text{aa}, \text{aaa} are all possible values for \text{X}. However, Prolog system responses with only \text{a}, and \text{aa} repeatedly. The reason is similar to (a). The search tree has infinite depth in the branch along clause (4). Clause (5) will never be tried.

(d)

\(?- \text{alike(X,Y)}.\]
\[ X = \text{a} Y = \text{aa} ; \]
\[ X = \text{aa} Y = \text{aaa} ; \]
\[ X = \_G157 Y = \_G157 ; \]
\[ X = \_G157 Y = \_G157 ; \]
\[ X = \text{a} Y = \text{a} ; \]
\[ X = \text{aa} Y = \text{aa} ; \]
\[ X = \_G157 Y = \_G157 ; \]
\[ X = \text{a} Y = \text{aa} ; \]
\[ X = \text{aa} Y = \text{aaa} ; \]
Yes

It doesn’t reflect logical conditions. According to logic, X=a, Y=aa; X=aa, Y=a; X=a, Y=aaa; X=aaa, Y=a; X=aa, Y=aaa; X=aaa, Y=aa; and X=Y, are all possible values for alike(X,Y) to be true. However, the system’s responses missed the cases when X=aaa, Y=a, etc. This happens for the same reason as in (a) and (c).

Please see the following Prolog search tree for resolving the goal.