الغاية من الجلسة: التعرف على اللوائح Lists في البرولوغ.

مقدمة:

اللوائح في البرولوغ هي هيكل بيانات أساسي يسمح بتنظيم وتخزين المعلومات. إليك أهمية اللوائح في البرولوغ بإيجاز:

1. تنظيم البيانات: اللوائح تتيح تنظيم المعلومات بشكل منظم وهرمي.
2. تمثيل المعلومات: تُستخدم لتخزين البيانات كمجموعة من العناصر.
3. إمكانيات البحث والاستعلام: تمكين عمليات البحث والاستعلام بفعالية.
4. تحقيق المرونة في التعبير: تُستخدم لتمثيل بيانات متغيرة ومعقدة.

بشكل عام، تعد اللوائح أداة قوية في برمجة البرولوغ تسهم في تسهيل التعامل مع البيانات وتحقيق مزيد من المرونة في البرمجة.

أمثلة على اللوائح:

[mia, vincent, jules, yolanda]

[mia, robber(honey_bunny), X, 2, mia]

[]

[mia, [vincent, jules], [butch, girlfriend(butch)]]

[[], dead(zed), [2, [b, chopper]], [], Z, [2, [b, chopper]]]

كما ترى فإن اللوائح يمكنها أن تحوي أي نوع من العناصر سواء أكان ثابتاً (atom أو عدد) أم متحولاً أم complex term وحتى اللوائح نفسها.
إن اللائحة [] هي نمط خاص من اللوائح وتدعى اللائحة الفارغة.
إن طول اللائحة هو عدد العناصر التي توجد ضمن اللائحة، فمثلاً طول اللائحة الفارغة هو 0 أما طول اللائحة الأولى في الأمثلة السابقة هو 4 أما طول اللائحة الأخيرة فهو 6.

أقسام اللائحة:

تقسم اللائحة إلى رأس head وذيل tail حيث أن الرأس هو العنصر في بداية اللائحة من اليسار، أما الذيل فهو اللائحة المكونة من باقي العناصر في اللائحة (أي أن الذيل هو اللائحة الأساسية بدون الرأس).

أمثلة:

List = [mia, vincent, jules, yolanda]

Head = mia, **Tail** = [vincent, jules, yolanda].

List = [[], dead(zed), [2, [b, chopper]], [], Z, [2, [b, chopper]]]

Head = [], **Tail** = [dead(zed), [2,[b,chopper]],[],Z,[2,[b, chopper]]].

List = [dead(zed)], **head** = dead(zed), **Tail** = [].

ويمكن تقسيم اللائحة في البرولوغ باستعمال معامل التقسيم | الذي يقسم اللائحة إلى رأس وذيل، وهنا بعض الأمثلة:

```
?- [Head| Tail] = [mia, vincent, jules, yolanda].
Head = mia,
Tail = [vincent, jules, yolanda].
```

```
?- [X|Y] = [].
false.

?- [X|Y] = [[], dead(zed), [2, [b, chopper]], [], Z].
X = [],
Y = [dead(zed), [2, [b, chopper]], [], Z].

?- [X,Y | W] = [[], dead(zed), [2, [b, chopper]], [], Z].
X = [],
Y = dead(zed),
W = [[2, [b, chopper]], [], Z].


?- [X,Y,C,O,P| W] = [[], dead(zed), [2, [b, chopper]], [], Z].
X = O, O = W, W = [],
Y = dead(zed),
C = [2, [b, chopper]],
P = Z.


?- [X,Y,Z|T] = [1,2].
false.


?- [_,X,_,Y|_] = [[], dead(zed), [2, [b, chopper]], [], Z].
X = dead(zed),
Y = [].


?- [_,_,[_|X]|_] =
|    [[], dead(zed), [2, [b, chopper]], [], Z, [2, [b, chopper]]].
X = [[b, chopper]].
```

لا تنس، الـ (_) underscore تعني أي شيء، ففي المثال الأخير نحن نستعلم عن ذيل العنصر الثالث (الذي هو لائحة).

مثال عن استخدام اللائحة:

تابع العضوية member والذي يختبر هل الوسيط الأول ينتمي إلى الوسيط الثاني الذي هو لائحة؟ إليك المثال الآتي:

?- member(vincent,[yolanda,trudy,vincent,jules]).

جواب الاستعلام السابق هو true أي أن vincent ينتمي للائحة.

?- member(vincent,[vincent]).

جواب الاستعلام السابق هو أيضاً true.

والآن حان دورك لتجيب عن الاستعلامات الآتية:

?- member(vincent,[]).

?- member(yolanda,[_,trudy,vincent,jules]).

?- member(X,[yolanda,trudy,vincent,jules]).

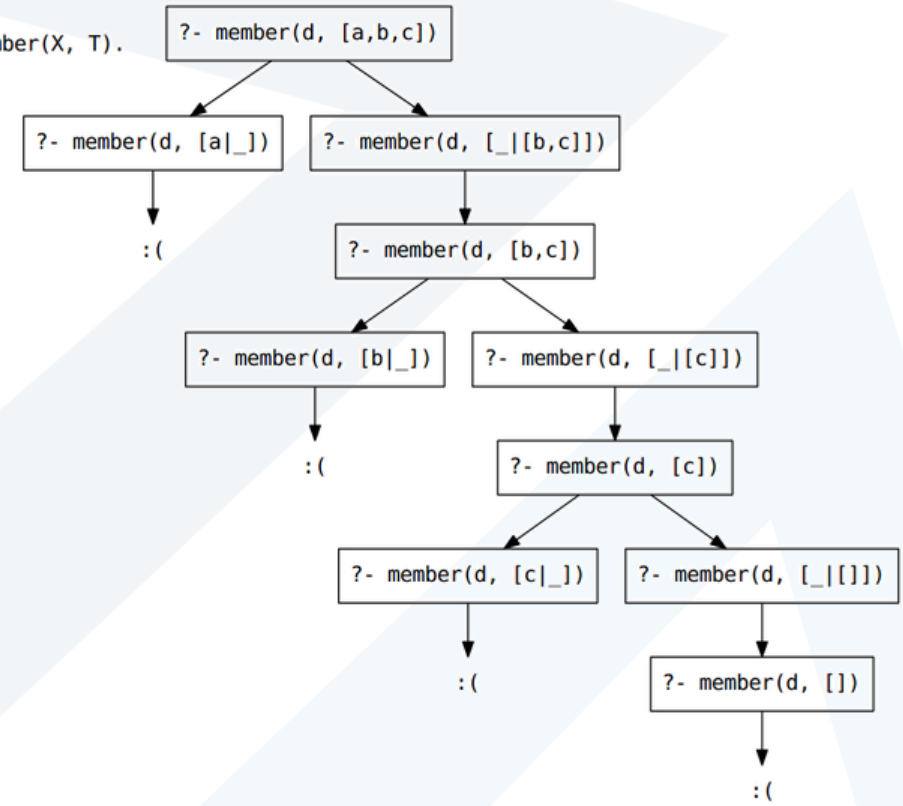
والآن ما هو شكل قاعدة المعرفة للمعلن member؟

member(X,[X|T]).

member(X,[H|T]) :- member(X,T).

الـ proof search الخاصة بالـ member:

```
member(X, [X|_]).
member(X, [_|T]) :- member(X, T).
```



تدريب:

لدينا معلن (predicate) تسمى a2b/2 تأخذ قائمتين كوسيطين وتنجح إذا كانت القائمة الأولى هي قائمة من كذا a، والقائمة الثانية هي قائمة من كذا b بنفس الطول تمامًا. على سبيل المثال، إذا قدمنا الاستعلام التالي:

?- a2b([a,a,a,a],[b,b,b,b]).

فالجواب هو true.

أو الاستعلام:

?- a2b([a,a,a,a],[b,b,b]).

والجواب هو false.

أو الاستعلام:

?- a2b([a,c,a,a],[b,b,5,4]).

والجواب هو false.

الحل:

```
a2b([],[]).
a2b([a|Ta],[b|Tb]) :- a2b(Ta,Tb).
```

تدريب:

المعلن الآتي اسمه la يعيد آخر عنصر في لائحة:

قاعدة للحالة عندما يكون القائمة تحتوي على عنصر واحد

la([X], X) :-!.

قاعدة للتحقق من أن القائمة فارغة وبالتالي لا يوجد آخر عنصر

la([], 'empty') .

قاعدة للحالة العامة

la([_|Tail], Last) :- la(Tail, Last).

تمرين للحل:

كيف سيجيب البرولوغ على الاستعلامات الآتية:

```
?-[a,b,c,d] = [a,[b,c,d]].
?-[a,b,c,d] = [a|[b,c,d]].
?-[a,b,c,d] = [a,b,[c,d]].
?-[a,b,c,d] = [a,b|[c,d]].
?-[a,b,c,d] = [a,b,c,[d]].
?-[a,b,c,d] = [a,b,c|[d]].
?-[a,b,c,d] = [a,b,c,d,[]].
?-[a,b,c,d] = [a,b,c,d|[]].
?-[] = _.
?-[] = [_].
?-[] = [_|[]].
```