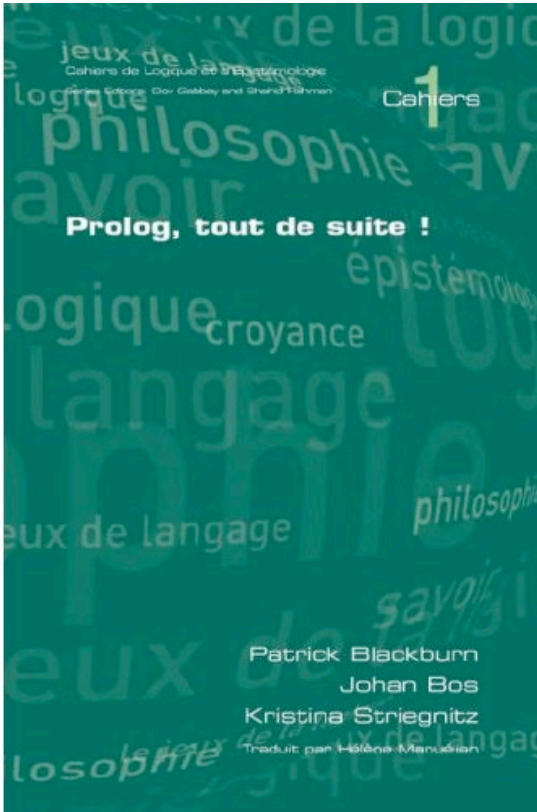


ELFE

- Programme:
 - programmation logique en Prolog
 - (6 semaines)
 - programmation fonctionnelle en OCaml
 - (6 semaines)
- Dates:
 - mercredi 11/11 férié. Les CTD des groupes 3 et 4 auront lieu vendredi 6/11
 - DS Prolog: vendredi 13 novembre
 - DS OCaml: vendredi 18 novembre

Prolog, tout de suite!



- original anglais gratuit en ligne
www.learnprolognow.org
- traduction française: 12euros,
amazon.fr
- commandé par la bibliothèque
universitaire

SWI Prolog

- Interpréteur de Prolog libre
 - Linux,
 - Windows, et
 - Mac OS
 - www.swi-prolog.org
- Il en existe d'autres

Cours 1

- Sujets:
 - Introduction à Prolog
 - Fait, règles et requêtes
 - La syntaxe de Prolog
- ... illustré par des exemples

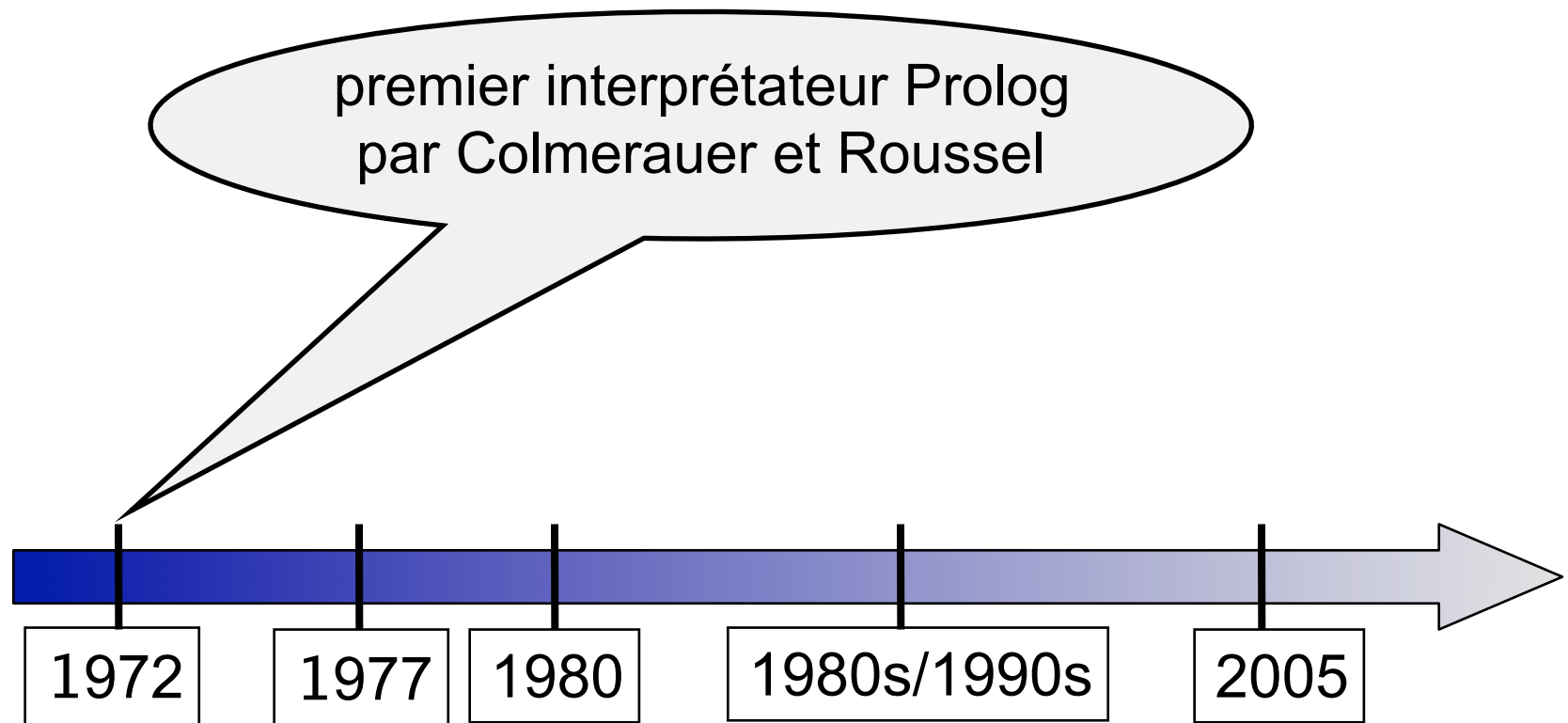
But de cette séance

- Montrer quelques programmes simples
- Discuter les 3 constructions essentielles de Prolog:
 - les faits
 - les règles
 - les buts à prouver, les requêtes
- Introduire d'autres aspects, tels que
 - le rôle de la logique
 - l'unification à l'aide de variables
- Démarrons par une inspection systématique de Prolog en définissant les éléments de sa syntaxe: atomes, termes, et variables

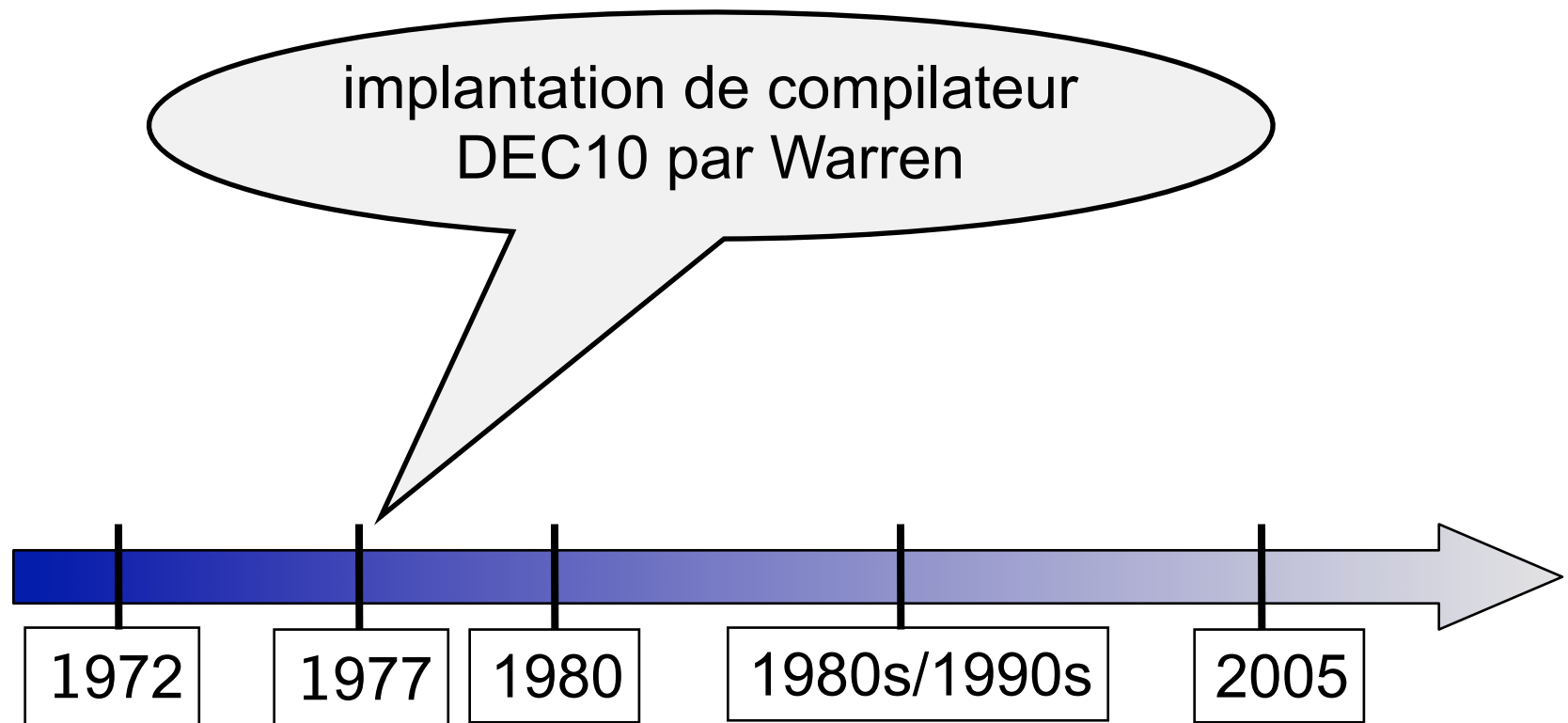
Prolog

- "Programmation avec la logique"
 - **déduction de nouveaux faits** à partir de faits et règles
- Un style déclaratif
 - ne pas dire *comment*, mais *quoi*
- Très différent des langages de programmation procéduraux/impératifs
- **L'exécution d'un programme est une preuve**: peut-on déduire une assertion d'une base de connaissances?
- Utile pour des tâches riches en connaissances

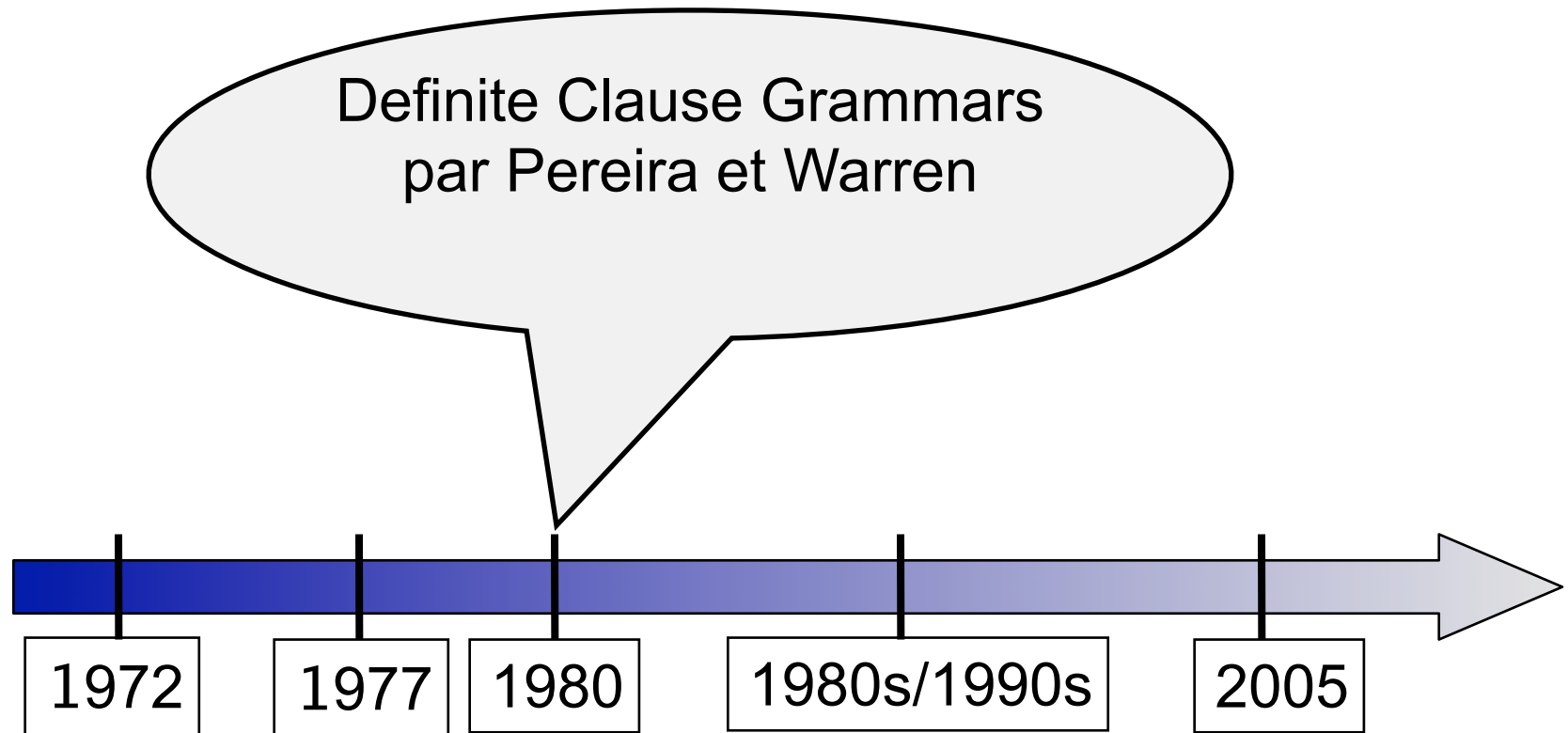
Histoire de Prolog



Histoire de Prolog

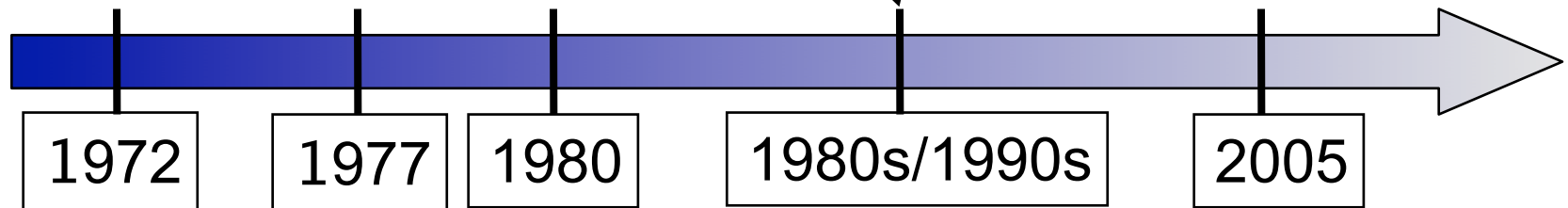


Histoire de Prolog



Histoire de Prolog

La popularité de Prolog augment
notamment en Europe et au Japon



Histoire de Prolog



1972

1977

1980

1980s/1990s

2005

Idée centrale de Prolog

- Décrire une situation d'intérêt
- Poser des requêtes
- Prolog **déduit des nouveaux faits**

A $A \Rightarrow B$

modus ponens

B

**B est une conséquence logique
d'A**

- Prolog nous donne ses conclusions
comme réponses

Conséquences

- Penser déclarativement, pas procéduralement
 - Challenge!
 - Demande une autre manière de voir les choses
- Langage à haut niveau
 - Moins efficace que, p.ex. le langage C
 - Bien pour le prototypage rapide
 - Utile pour nombreuses applications en intelligence artificielle

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?-
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- femme(mia).
```


Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- femme(mia).  
yes  
?-
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- femme(mia).  
yes  
?- joueAirGuitar(jody).
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- femme(mia).  
yes  
?- joueAirGuitar(jody).  
yes  
?-
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- femme(mia).  
yes  
?- joueAirGuitar(jody).  
yes  
?- joueAirGuitar(mia).  
no
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- tatoue(jody).
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- tatoue(jody).  
no  
?-
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- tatoue(jody).  
ERROR: Prédicat tatoue/1 not defined.  
?-
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- party.
```


Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- party.  
yes  
?-
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- rockConcert.
```

Base de connaissances 1

```
femme(mia).  
femme(jody).  
femme(yolanda).  
joueAirGuitar(jody).  
party.
```

```
?- rockConcert.  
no  
?-
```

Base de connaissances 2

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

Base de connaissances 2

fait

cool(yolanda).

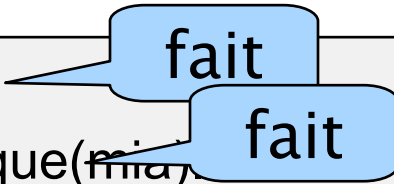
ecoute2laMusique(mia).

ecoute2laMusique(yolanda):- cool(yolanda).

joueAirGuitar(mia):- ecoute2laMusique(mia).

joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).

Base de connaissances 2



```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

Base de connaissances 2

cool(yolanda).

fait

ecoute2laMusique(mia).

fait

ecoute2laMusique(yolanda):- cool(yolanda).

règle

joueAirGuitar(mia):- ecoute2laMusique(mia).

joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).

Base de connaissances 2

cool(yolanda).

fait

ecoute2laMusique(mia).

fait

ecoute2laMusique(yolanda):- cool(yolanda).

règle

joueAirGuitar(mia):- ecoute2laMusique(mia).

règle

joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).

Base de connaissances 2

cool(yolanda).

fait

ecoute2laMusique(mia).

fait

ecoute2laMusique(yolanda):- cool(yolanda).

règle

joueAirGuitar(mia):- ecoute2laMusique(mia).

règle

joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).

règle

Base de connaissances 2

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

tête

corps

Base de connaissances 2

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

?-

Base de connaissances 2

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

```
?- joueAirGuitar(mia).  
yes  
?-
```

Base de connaissances 2

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

```
?- joueAirGuitar(mia).  
yes  
?- joueAirGuitar(yolanda).  
yes
```

Clauses

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

Cette base de connaissances contient 5 clauses:

deux faits, trois règles.

La fin d'une clause est indiquée par un point.

Prédicats

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

Notre base de connaissances contient trois prédicats:

cool, ecoute2laMusique, et joueAirGuitar

Base de connaissances 3

```
cool(vincent).  
ecoute2laMusique(butch).  
joueAirGuitar(vincent):-  
    ecoute2laMusique(vincent), cool(vincent).  
joueAirGuitar(butch):- cool(butch).  
joueAirGuitar(butch):- ecoute2laMusique(butch).
```


Exprimer la conjonction (“et” logique)

```
cool(vincent).  
ecoute2laMusique(butch).  
joueAirGuitar(vincent):- ecoute2laMusique(vincent), cool(vincent).  
joueAirGuitar(butch):- cool(butch).  
joueAirGuitar(butch):- ecoute2laMusique(butch).
```

La virgule “,” exprime la conjonction en Prolog

Base de connaissances 3

```
cool(vincent).
ecoute2laMusique(butch).
joueAirGuitar(vincent):- ecoute2laMusique(vincent), cool(vincent).
joueAirGuitar(butch):- cool(butch).
joueAirGuitar(butch):- ecoute2laMusique(butch).
```

```
?- joueAirGuitar(vincent).
no
?-
```

Base de connaissances 3

```
cool(vincent).  
ecoute2laMusique(butch).  
joueAirGuitar(vincent):- ecoute2laMusique(vincent), cool(vincent).  
joueAirGuitar(butch):- cool(butch).  
joueAirGuitar(butch):- ecoute2laMusique(butch).
```

```
?- joueAirGuitar(butch).  
yes  
?-
```

Exprimer la disjonction ("ou" logique)

```
cool(vincent).  
ecoute2laMusique(butch).  
joueAirGuitar(vincent):- ecoute2laMusique(vincent), cool(vincent).  
joueAirGuitar(butch):- cool(butch).  
joueAirGuitar(butch):- ecoute2laMusique(butch).
```

```
cool(vincent).  
ecoute2laMusique(butch).  
joueAirGuitar(vincent):- ecoute2laMusique(vincent), cool(vincent).  
joueAirGuitar(butch):- cool(butch); ecoute2laMusique(butch).
```

point-virgule: syntaxe possible, mais à éviter!

Prolog et la logique

- Prolog a un rapport clair avec la logique
- Opérateurs
 - Implication :-
 - Conjonction ,
 - Disjonction ; *(syntaxe possible mais à éviter)*
- Application du modus ponens
- Négation

Base de connaissances 4

```
femme(mia).  
femme(jody).  
femme(yolanda).  
  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

Les variables Prolog

```
femme(mia).  
femme(jody).  
femme(yolanda).
```

```
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- femme(X).
```

Instantiation de variable

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- femme(X).  
X=mia
```


Demander une alternative

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- femme(X).  
X=mia ;
```

Demander une alternative

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- femme(X).  
X=mia;  
X=jody ;
```

Demander une alternative

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- femme(X).  
X=mia;  
X=jody;  
X=yolanda
```

Demander une alternative

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- femme(X).  
X=mia;  
X=jody;  
X=yolanda ;  
no
```

Base de connaissances 4

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- aime(marsellus,X), femme(X).
```

Base de connaissances 4

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- aime(marsellus,X), femme(X).  
X=mia  
yes  
?-
```

Base de connaissances 4

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- aime(mon_chou,X), femme(X).
```

Base de connaissances 4

```
femme(mia).  
femme(jody).  
femme(yolanda).  
aime(vincent, mia).  
aime(marsellus, mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).
```

```
?- aime(mon_chou,X), femme(X).  
no  
?-
```


Base de connaissances 5

```
aime(vincent,mia).  
aime(marsellus,mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).  
jaloux(X,Y):- aime(X,Z), aime(Y,Z).
```

Base de connaissances 5

```
aime(vincent,mia).  
aime(marsellus,mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).  
jaloux(X,Y):- aime(X,Z), aime(Y,Z).
```

```
?- jaloux(marsellus,W).
```

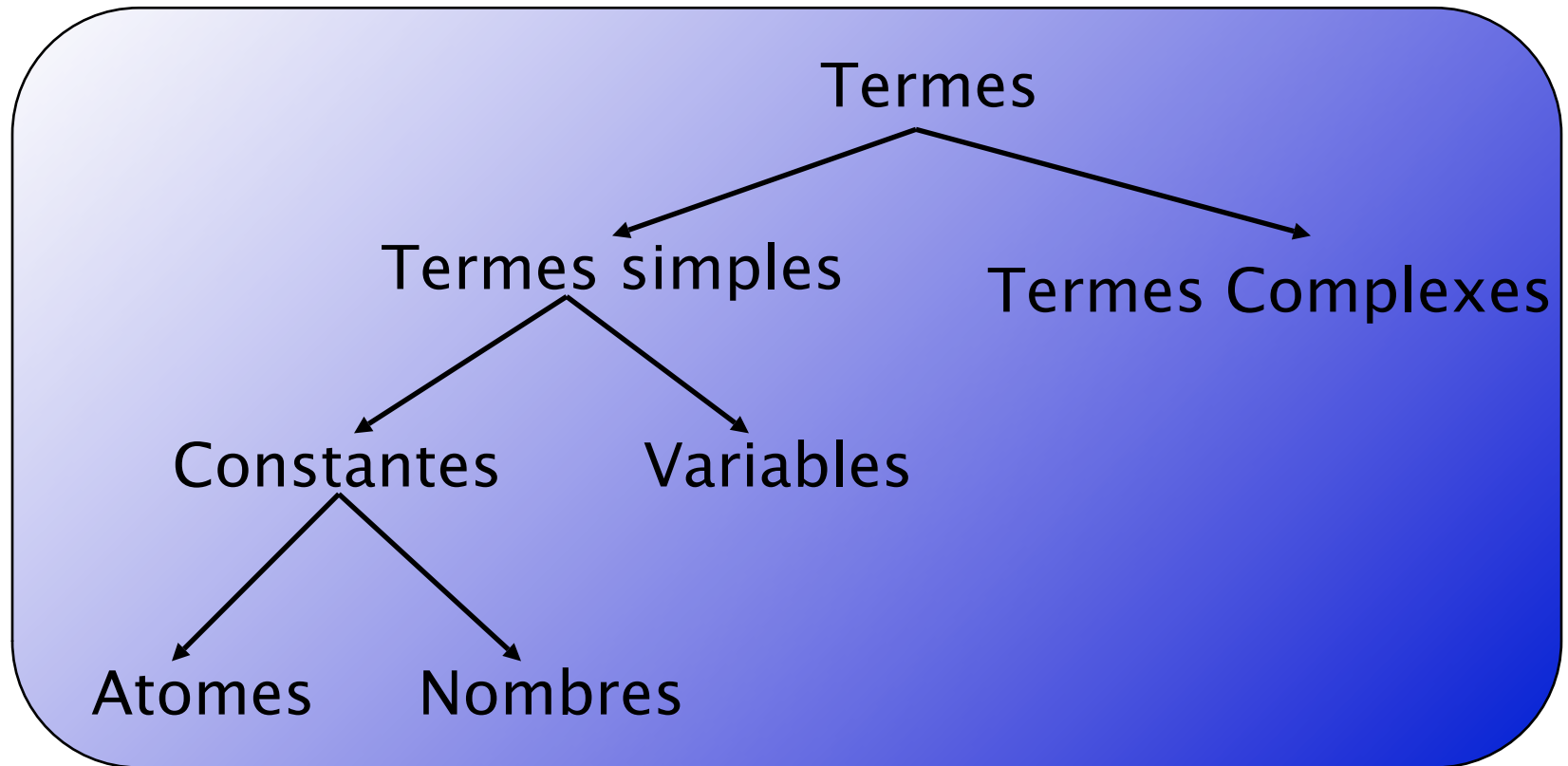
Base de connaissances 5

```
aime(vincent,mia).  
aime(marsellus,mia).  
aime(lapin, mon_chou).  
aime(lapin, mon_chou).  
jaloux(X,Y):- aime(X,Z), aime(Y,Z).
```

```
?- jaloux(marsellus,W).  
W=vincent  
?-
```

Syntaxe Prolog

- Comment précisément construire les faits, règles et requêtes?



Atomes

- Une chaîne de caractères de lettres majuscules, minuscules, chiffre, et tiret_bas, commençant avec une lettre **minuscule**
 - *Exemples:* **butch**, **big_kahuna_burger**, **jouerALaGuitarre**
- Une chaîne quelconque de caractères encadré par deux single quotes
 - *Exemples:* **'Vincent'**, **'Five dollar shake'**, **'@\$%'**
- Une chaîne de caractères spéciaux
 - *Exemples:* **:**, **,**, **;**, **.**, **:-**

Nombres

- Entiers: 12, -34, 22342
- Nombres réels: 34573.3234

Variables

- Une chaîne de lettres majuscules, minuscules, chiffres ou tiret_bas commençant soit par une lettre **majuscule** soit par un **tiret_bas**
- Exemples:

X, Y, Variable, Vincent, _tag

Termes complexes

- Les atomes, nombres et variables sont les composantes des termes complexes
- Un terme complexe est construit par un **foncteur** suivi d'une **séquence d'arguments**
- Les arguments sont placés en parenthèses (), et séparés par des virgules
- Le foncteur doit être un **atome**

Définition des termes

Forme Backus Naur
(BNF)

foncteur

arguments

$\text{terme} ::= \text{atome}(\text{terme}, \dots, \text{terme}) \mid \text{variable} \mid \text{constante}$

$\text{constante} ::= \text{atome} \mid \text{nombre}$

Exemples de termes

- Vu précédemment:
 - joueAirGuitar(jody)
 - aime(vincent, mia)
 - jaloux(marsellus, W)
- Termes complexes dans termes complexes:
 - hide(X,father(father(father(butch))))

Arité

- Le **nombre d'arguments** d'un terme complexe est son arité.
- Exemples:

| | |
|------------------------------|-----------------|
| femme(mia) | terme d'arité 1 |
| aime(vincent,mia) | arité 2 |
| father(father(butch)) | arité 1 |

L'arité importe

- En Prolog on peut définir deux prédicats avec le même foncteur, mais d'arités différentes
- Prolog les considère comme deux prédicats distincts
- Dans la documentation Prolog l'arité d'un prédicat est indiquée par le suffixe "/" suivi d'un nombre (son arité).

Exemple d'arité

```
cool(yolanda).  
ecoute2laMusique(mia).  
ecoute2laMusique(yolanda):- cool(yolanda).  
joueAirGuitar(mia):- ecoute2laMusique(mia).  
joueAirGuitar(yolanda):- ecoute2laMusique(yolanda).
```

- Cette base de connaissances défini
 - cool/1
 - ecoute2laMusique/1
 - joueAirGuitar/1

Exercices

Section 1.3 du livre!

Résumé

- Exemples de programmes Prolog simples
- Les trois constructions de base de Prolog:
 - faits et règles (Base de connaissances)
 - requêtes
- Nous avons vu d'autres concepts, tels que
 - le rôle de la logique, la déduction par le modus ponens
 - l'unification avec des variables
- Éléments de la syntaxe Prolog:
termes, atomes, et variables

Prochaine séance

- Plus sur l'**unification** en Prolog
- La stratégie de recherche de Prolog