

Compte-rendu TP1

Programmation Logique

Réalisatrice:

Fatma Laribi GL3 groupe 2

Plan du Compte-rendu

- I. Objectif
- II. Introduction générale
- III.Exercice 1
- IV.Exercice 2

Objectif

Installer et se familiariser avec l'outil de
Programmation Logique SWI-Prolog

Introduction Générale

Prolog: C'est quoi?

Prolog

- Il a été créé par Alain Colmerauer et Philippe Roussel vers 1972 à Luminy, Marseille.
- Le but était de créer un langage de programmation où seraient définies les règles logiques attendues d'une solution et de laisser le compilateur la transformer en séquence d'instructions.
- Prolog est basé sur le calcul des prédicats du premier ordre
- On peut construire en Prolog une base de connaissances dans un ordre indéterminé, puisque seules comptent les relations en présence et non leur séquence d'écriture.

Atomes

Commencent par une **lettre minuscule**. Les textes constants constituent des atomes. Un atome est ordinairement constitué d'une chaîne de lettres, nombres et traits bas (_). Pour introduire un atome non alphanumérique, on l'entoure d'apostrophes : ainsi '+' est un atome, + un opérateur.

Nombres

Les implémentations courantes de Prolog ne distinguent pas les nombres entiers des flottants.

Variables

- Les variables sont indiquées en utilisant un ensemble de lettres, nombres et caractères de soulignement et commençant avec une lettre majuscule. Ainsi, X3 comme Prix_Unitaire sont des noms de variables admissibles.
- Prolog n'est pas un langage de programmation impératif ; une variable n'y est donc pas un contenant auquel on affecte une valeur, mais représente (comme en mathématiques dans $X > 0$) l'ensemble des valeurs admissibles pour elle dans le cadre des contraintes.

Termes composés

- aime(romeo, juliette)

Le foncteur est aime et l'arité 2, le terme composé s'écrit aime/2.

- f(g(X),h(Y))

Le foncteur est f et l'arité 2, le terme composé s'écrit f/2.

- initialisation

Le foncteur est initialisation et l'arité 0, le terme composé s'écrit initialisation/0. Un atome est donc un terme composé d'arité 0.

Listes

1. l'atome [] est une liste vide
2. si T est une liste et H est un élément, alors le terme '!(H, T) est une liste.

Le premier élément, appelé la tête, est H, suivi par les contenus du reste de la liste, indiqué comme T ou queue.

La liste [1, 2, 3] serait représentée en interne comme '!(1, '!(2, '!(3, [])) Un raccourci de syntaxe est [H | T], lequel est surtout utilisé pour construire des règles. La totalité d'une liste peut être traitée en agissant sur le premier élément, et ensuite sur le reste de la liste, par récursivité

Prédicats

La programmation en Prolog est très différente de la programmation dans un langage impératif. En Prolog, on alimente une base de connaissances de faits et de règles ; il est alors possible de faire des requêtes à la base de connaissances.

L'unité de base de Prolog est le prédicat, qui est défini comme étant vrai. Un prédicat consiste en une tête et un nombre d'arguments. Par exemple :

pere(pierre, marie).

signifie que Pierre est le père de Marie.

Réversibilité

age(capitaine, 45) est vrai ou faux ; age(capitaine, X) demande quel est l'âge X du capitaine, age(X, 45) demande quel X a 45 ans.

Règles

Le second type d'instructions en Prolog est la règle. Un exemple de règle est :

lumière(on) :- interrupteur(on).

Le « :- » signifie « si »; cette règle indique lumière(on) est vraie si interrupteur(on) est vrai. Les règles peuvent aussi utiliser des variables comme :

père(X,Y) :- parent(X,Y), mâle(X).

pour signifier qu'un X est père d'un Y si X est parent de Y et X est mâle, où " , " indique une conjonction.

On pourrait avoir de même :

parent(X, Y) :- père(X, Y) ; mère(X, Y).

pour signifier qu'un X est parent d'un Y si X est père de Y ou X est mère de Y, où " ; " indique une alternative.

Evaluation

Quand l'interpréteur reçoit une requête, il recherche les règles (faits inclus) dont la partie gauche peut être unifiée avec la requête, et effectue cette unification avec la première règle trouvée. Par exemple ayant ce code Prolog :

```
frère_ou_sœur(X,Y) :- parent(Z,X), parent(Z,Y), X \= Y.
```

```
parent(X,Y) :- père(X,Y).
```

```
parent(X,Y) :- mère(X,Y).
```

```
mère(trude, sally).
```

```
père(tom, sally).
```

```
père(tom, erica).
```

```
père(mike, tom).
```

Il en résulte que la demande suivante est évaluée comme vraie:

```
?- frère_ou_sœur(sally, erica).
```

oui.

Télécharger SWI-prolog + exécuter le
fichier téléchargé pour installer le
programme



Download binary

HOME DOWNLOAD DOCUMENTATION TUTORIALS COMMUNITY USERS

WIKI

⚠ Windows antivirus software works using *signatures* and *heuristics*. Using the huge amount of viruses and malware known today, arbitrary executables are often falsily classified as malicious. [Google Safe Browsing](#), used by most modern browsers, therefore often classifies our Windows binaries as malware. You can use e.g., [virustotal](#) to verify files with a large number of antivirus programs.

Our Windows binaries are cross-compiled on an isolated Linux container. The integrity of the binaries on the server is regularly verified by validating its SHA256 fingerprint.

Please select the checkbox below to enable the actual download link.

☒ I understand

[Download swipl-8.4.2-1.x64.exe](#) SHA256: 1438b04d0a6acb116ba836e7330b6002032b601d26334a86275993f316f95c69)

[VIRUSTOTAL Scan Result](#)

HOME

DOWNLOAD

DOCUMENTATION

TUTORIALS

COMMUNITY

USERS

WIKI



Linux versions are often available as a package for your distribution. We collect information about available packages and issues for building on specific distros [here](#). We provide a [PPA](#) for [Ubuntu](#) and [snap images](#)



Android binaries are available for [Termux](#) as the package `swi-prolog`. See also [Building SWI-Prolog on Android using LinuxOnAndroid](#)



Please check the [windows release notes](#) (also in the SWI-Prolog startup menu of your installed version) for details.



Examine the [ChangeLog](#).

Binaries



12,530,177
bytes

SWI-Prolog 8.4.2-1 for Microsoft Windows (64 bit)

Self-installing executable for Microsoft's Windows 64-bit editions. Requires at least Windows 7. See the [reference manual](#) for deciding on whether to use the 32- or 64-bits version. This binary is linked against GMP 6.1.1 which is covered by the LGPL license.

SHA256: 1438b04d0a6acb116ba836e7330b6002032b601d26334a86275993f316f95c69

SWI-Prolog 8.4.2-1 for Microsoft Windows (32 bit)

Self-installing executable for MS-Windows. Requires at least Windows 7. Installs `swipl-win.exe` and `swipl.exe`. This binary is linked against GMP 6.1.1 which is covered by the LGPL license.

SHA256: aa7c04dfefca4e1d1a44809ee1d1931e4530bc71b6ba7992948441ca1a3bc14a

SWI-Prolog 8.4.2-1 for MacOSX 10.14 (Mojave) and later on x86_64 and arm64

Installer with binaries created using [Macports](#). Installs `/opt/local/bin/swipl`. Needs [xquartz](#) (X11) and the Developer Tools (Xcode) installed for running the [development tools](#)

SHA256: c838ad6c230c78d5d0d731a538d09ea24c64479ed18aaa6323f19b8d740aa1a5

SWI-Prolog 8.4.1-1 for MacOSX bundle on intel

Installer with binaries created using [Macports](#). Installs `/opt/local/bin/swipl`. Needs [xquartz](#) (X11) and the Developer Tools (Xcode) installed for running the [development tools](#)

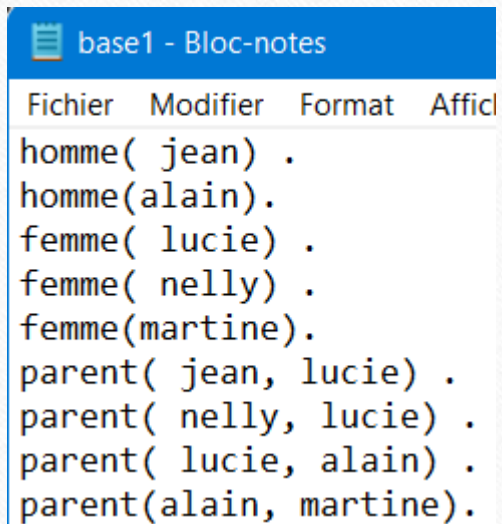
SHA256: 1b9c62caa781818a0dafd1d822ab563b8c10c7cd018ce10a3b71f900eb3a434f

Sources

Exercice 1

- i. Base de faits
- ii. Requêtes
- iii. Variables
- iv. Règles
- v. Variable anonyme

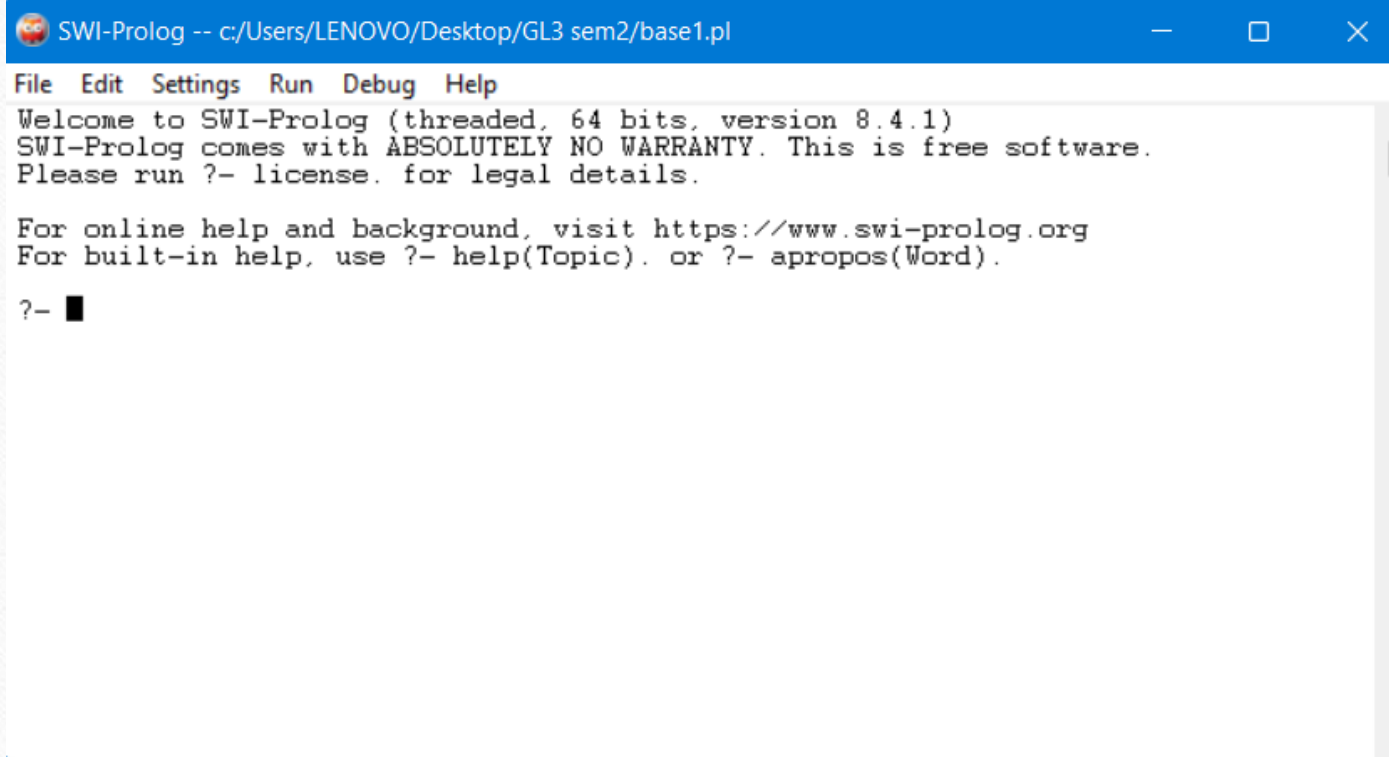
Base de faits



```
base1 - Bloc-notes
Fichier  Modifier  Format  Affic
homme( jean) .
homme(alain).
femme( lucie) .
femme( nelly) .
femme(martine).
parent( jean, lucie) .
parent( nelly, lucie) .
parent( lucie, alain) .
parent(alain, martine).
```

On commence par définir la base de faits et on la sauvegarde sous le nom de base1.pl

Requêtes



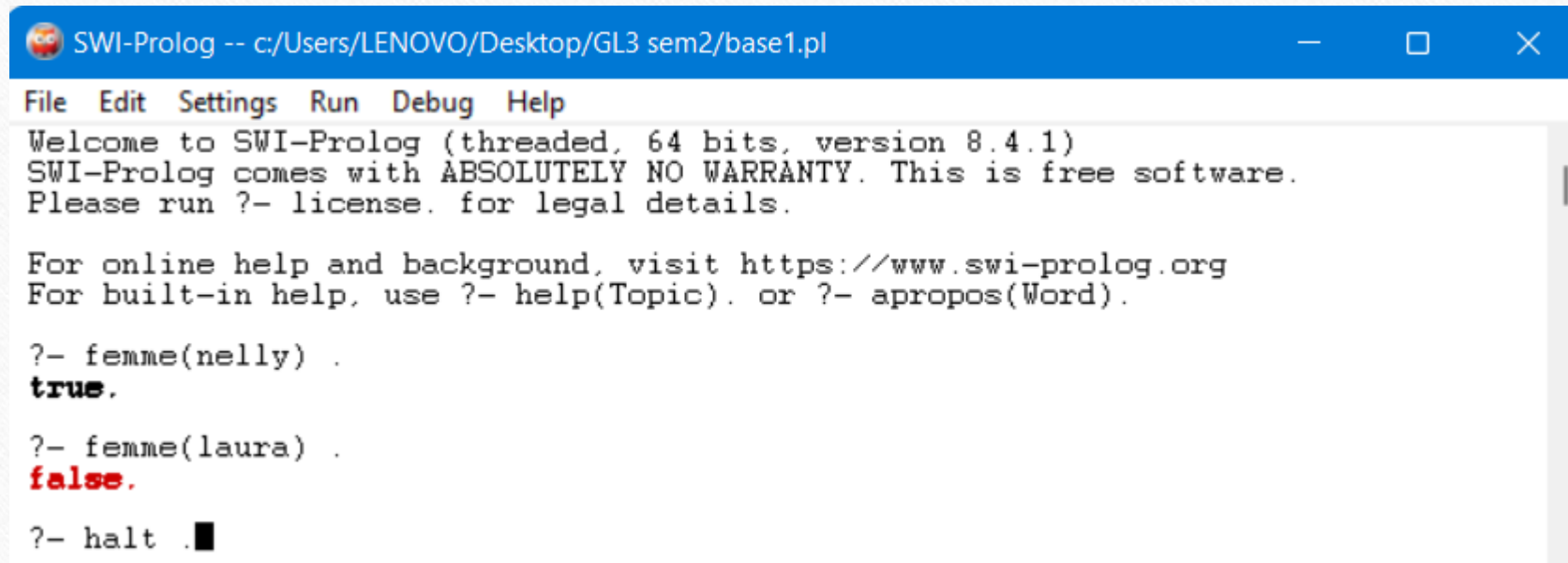
```
SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/base1.pl
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- █
```

Avec un double clic sur le fichier précédent on ouvre swi-prolog

On essaye les requêtes suivantes:



```
SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/base1.pl
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- femme(nelly) .
true.

?- femme(laura) .
false.

?- halt . █
```

Variables

SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/base1.pl

File Edit Settings Run Debug Help

Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run `?- license.` for legal details.

For online help and background, visit <https://www.swi-prolog.org>
For built-in help, use `?- help(Topic).` or `?- apropos(Word).`

`?- homme(X) .`

On cherche les X tels que X est un homme

`X = jean ;`

On tape espace pour voir s'il y a d'autres X qui satisfont la condition

`X = alain.`

`?- femme(X),parent(jean,X) .`

On peut chercher avec plusieurs buts séparés par une virgule

`X = lucie ;`


false.

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

false.

`?- ■`

Règles

 base1 - Bloc-notes

Fichier Modifier Format Affichage Aide

```
homme( jean) .  
homme(alain).  
femme( lucie) .  
femme( nelly) .  
femme(martine).  
parent( jean, lucie) .  
parent( nelly, lucie) .  
parent( lucie, alain) .  
parent(alain, martine).  
ancetre(X,Y) :- parent(X,Y) .  
ancetre(X,Y) :- parent(X,Z), ancetre(Z,Y) .
```

On ajoute les règles relatives à ancetre(X,Y) au fichier base1.pl

On teste:

Les ancêtres de martine

```
SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/base1.pl
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- ancetre(X,martine) .
X = alain ;
X = jean ;
X = nelly ;
X = lucie ;
false.
```

Les descendants d'alain

```
?- ancetre(alain,Y) .
Y = martine ;
false.
```

les decendants de jean qui sont des hommes

```
?- ancetre(jean,Y),homme(Y) .
Y = alain ;
false.
```


Variable anonyme

```
pere( X) : - pere( X, Y).
```

Ce qui veut dire quelque chose comme “X est père (tout court) si c’est le père de quelqu’un”...

Si on teste ce prédicat, Prolog affiche un Warning à la compilation qui n’est pas forcément une erreur, mais qui signifie : nous avons utilisé une variable Y mais dont la valeur n’apparaît pas dans le résultat.

Ce warning peut disparaître en utilisant une variable anonyme, qui est notée par un soulignement (_). Notre règle devient alors :

```
pere( X) : - pere( X, _).
```

On ajoute cette règle au fichier base1.pl

base1 - Bloc-notes

Fichier Modifier Format Affichage Aide

```
homme( jean) .  
homme(alain).  
femme( lucie) .  
femme( nelly) .  
femme(martine).  
parent( jean, lucie) .  
parent( nelly, lucie) .  
parent( lucie, alain) .  
parent(alain, martine).  
ancetre(X,Y) :- parent(X,Y) .  
ancetre(X,Y) :- parent(X,Z), ancetre(Z,Y) .  
pere(X,Y) :- parent(X,Y), homme(X) .  
pere(X) :- pere( X, _).
```

Et on la teste

```
?- pere(X) .  
X = jean ;  
X = alain.
```

On ajoute des règles aux fichiers tel que le fichier final devient:

On les teste:

```
base1 - Bloc-notes
Fichier  Modifier  Format  Affichage  Aide
homme( jean) .
homme(alain).
femme( lucie) .
femme( nelly) .
femme(martine).
parent( jean, lucie) .
parent( nelly, lucie) .
parent( lucie, alain) .
parent(alain, martine).
ancetre(X,Y) :- parent(X,Y) .
ancetre(X,Y) :- parent(X,Z), ancetre(Z,Y) .
pere(X,Y) :- parent(X,Y), homme(X) .
pere(X) :- pere( X, _ ) .
enfant(X,Y) :- parent(Y,X) .
fils(X,Y) :- enfant(X,Y),homme(X) .
fille(X,Y) :- enfant(X,Y),femme(X) .
mere(X,Y) :- parent(X,Y),femme(X) .
grand_parent(X,Y) :- parent(Z,Y),parent(X,Z) .
```

```
?- enfant(X,Y) .
X = lucie,
Y = jean ;
X = lucie,
Y = nelly ;
X = alain,
Y = lucie ;
X = martine,
Y = alain.
```

```
?- fils(X,Y) .
X = alain,
Y = lucie ;
false.
```

```
?- fille(X,Y) .
X = lucie,
Y = jean ;
X = lucie,
Y = nelly ;
X = martine,
Y = alain.
```

```
?- mere(X,Y) .
X = nelly,
Y = lucie ;
X = lucie,
Y = alain ;
false.
```

```
?- grand_parent(X,Y)
X = jean,
Y = alain ;
X = nelly,
Y = alain ;
X = lucie,
Y = martine.
```


Exercice 2

- i. Test initial du programme
- ii. Test du programme sans variables
- iii. Test du programme avec boucle
- iv. Test du programme avec factorielle

Test initial du programme

base2 - Bloc-notes

Fichier Modifier Format Affichage Aide

```
lire(X) :- write('donner un entier '), nl, read(X), nl, write('votre entier est '),write(X),nl, nl .  
calcul_carre(X,Y):- Y is X * X .  
ecrire_resultat(X,Y) :- write('le carré de '), write(X), write(' est '),write(Y), nl, nl .  
aller :- lire(X), calcul_carre(X,Y), écrire_resultat(X,Y) .
```


SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/prog log/base2.pl



File Edit Settings Run Debug Help

Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run `?- license.` for legal details.

For online help and background, visit <https://www.swi-prolog.org>
For built-in help, use `?- help(Topic).` or `?- apropos(Word).`

```
?- lire(X) .  
donner un entier  
|: 7.
```

votre entier est 7

X = 7.

```
?- calcul_carre(8,Y).  
Y = 64.
```

```
?- ecrire_resultat(8,64).  
le carré de 8 est 64
```

true.

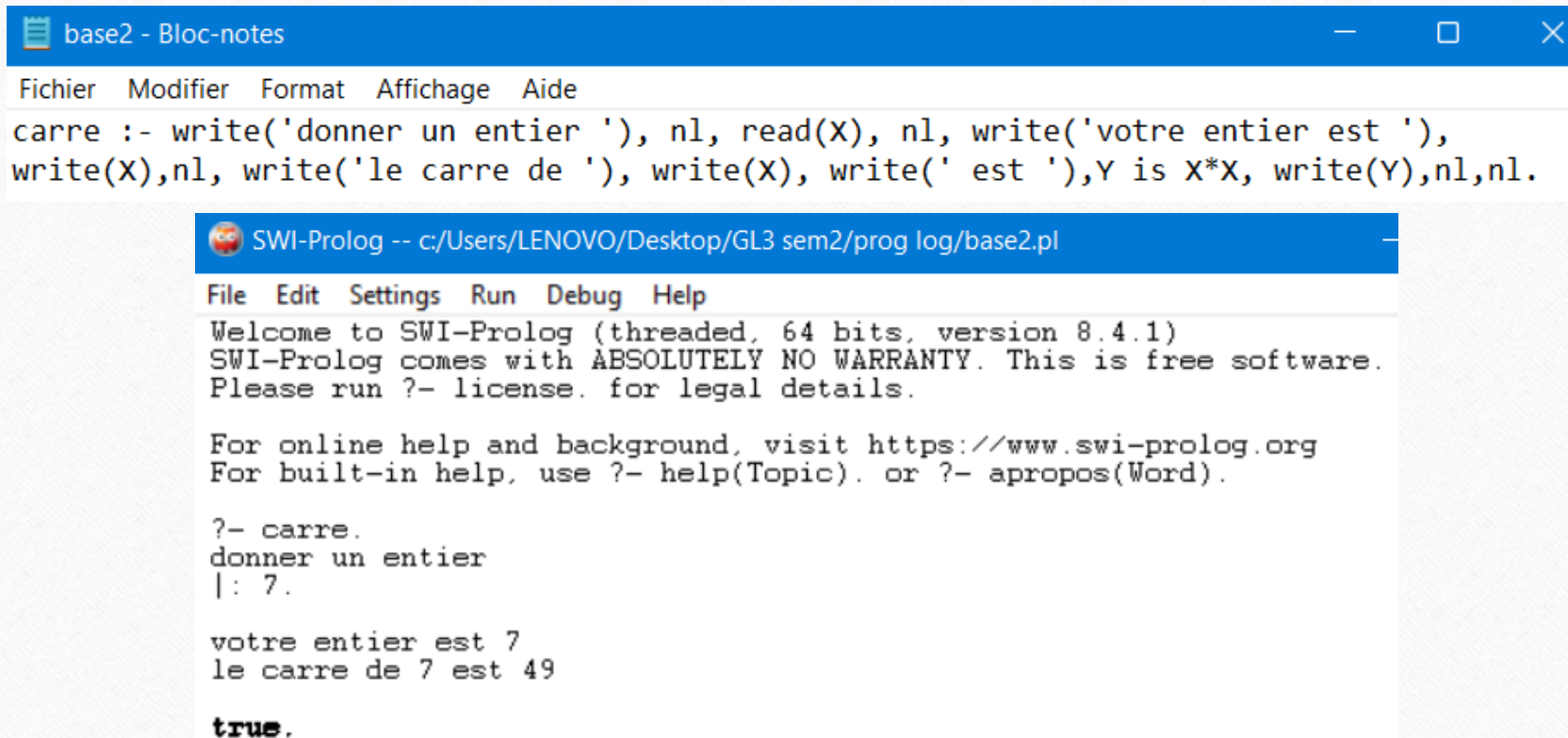
```
?- aller .  
donner un entier  
|: 5.
```

votre entier est 5

le carré de 5 est 25

true.

Test du programme sans variables



The image shows two overlapping windows. The top window, titled 'base2 - Bloc-notes', contains a Prolog program. The bottom window, titled 'SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/prog log/base2.pl', shows the execution of this program. The program prompts the user to enter an integer, calculates its square, and displays the result.

```
base2 - Bloc-notes
Fichier  Modifier  Format  Affichage  Aide
carre :- write('donner un entier '), nl, read(X), nl, write('votre entier est '),
write(X),nl, write('le carre de '), write(X), write(' est '),Y is X*X, write(Y),nl,nl.

SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/prog log/base2.pl
File  Edit  Settings  Run  Debug  Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.1)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.


For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- carre.
donner un entier
|: 7.

votre entier est 7
le carre de 7 est 49


true.
```


Test du programme avec boucle

 base2 - Bloc-notes

Fichier Modifier Format Affichage Aide

```
lire(X) :- write('donner un entier '), nl, read(X), nl, write('votre entier est '), write(X), nl, nl.  
calcul_carre(X,Y):- Y is X * X.  
ecrire_resultat(X,Y) :- write('le carre de '), write(X), write(' est '), write(Y), nl, nl.  
aller(X) :- lire(X), calcul_carre(X,Y), ecrire_resultat(X,Y).  
loop :- aller(X), X==0 -> write('end of loop'); loop .
```

 SWI-Prolog -- c:/Users/LENOVO/Desktop/GL3 sem2/prog log/base2.pl

File Edit Settings Run Debug Help

Please run `?- license.` for legal details.

For online help and background, visit <https://www.swi-prolog.org>

For built-in help, use `?- help(Topic).` or `?- apropos(Word).`

`?- loop.`

`donner un entier`

`|: 5.`

`votre entier est 5`

`le carre de 5 est 25`

`donner un entier`

`|: 2.`

`votre entier est 2`

`le carre de 2 est 4`

`donner un entier`

`|: 0.`

`votre entier est 0`

`le carre de 0 est 0`

`end of loop`

`true.`

Test du programme avec factorielle

base2 - Bloc-notes

Fichier Modifier Format Affichage Aide

```
lire(X) :- write('donner un entier '), nl, read(X), nl, write('votre entier est '),write(X),nl,nl.  
calcul_carre(X,Y):- Y is X * X.  
ecrire_resultat(X,Y) :- write('le carre de '), write(X), write(' est '),write(Y), nl,nl.  
aller(X) :- lire(X), calcul_carre(X,Y), écrire_resultat(X,Y).  
loop :- aller(X), X==0 -> write('end of loop'); loop .  
fact(0,1).  
fact(N,X) :- N>0, N1 is N-1, fact(N1,X1), X is N*X1.
```

Exécution

```
?- fact(5,X).  
X = 120 .
```




Merci pour votre
attention!

Réalisé par Fatma Laribi GL3/2