



**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**  
**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO**  
**LOGIC BẬC NHẤT**

|                         |                 |
|-------------------------|-----------------|
| <b>Họ và tên</b>        | <b>MSSV</b>     |
| <b>Trần Hữu Chí Bảo</b> | <b>18120288</b> |
| <b>Trần Thanh Tùng</b>  | <b>18120258</b> |
| <b>Vòng Cảnh Chi</b>    | <b>18120293</b> |

**Môn : Cơ sở trí tuệ nhân tạo**

**Thành phố Hồ Chí Minh - 2020**

# Mục lục

## Contents

|  |    |
|--|----|
| Mục lục .....  | 2  |
| Phần 1. Làm quen với công cụ Prolog .....  | 3  |
| I.    Tìm hiểu ngôn ngữ Prolog. Viết báo cáo về các đặc điểm chính của ngôn ngữ. Cần đưa ra nhiều ví dụ minh họa, liên hệ với kiến thức về logic bậc nhất đã học. ....   | 3  |
| 1.1    Sơ lược vài nét về ngôn ngữ Prolog: .....   | 3  |
| 1.2    Cú pháp:.....   | 3  |
| 1.3    Ngữ nghĩa: .....  | 4  |
| 1.4    Thực thi: .....   | 5  |
| 1.5    Vòng lặp và đệ quy:.....  | 6  |
| 1.6    Liên hệ giữa Prolog và logic bậc nhất: .....  | 6  |
| II.    Tìm hiểu một môi trường lập trình Prolog. Viết báo cáo về cách thức triển khai ngôn ngữ Prolog trên công cụ đã tìm hiểu. Trình bày ít nhất 5 ví dụ minh họa. .... | 8  |
| 2.1    Vài nét về môi trường SWI-Prolog: .....   | 8  |
| 2.2    Hướng dẫn cài đặt:.....   | 8  |
| 2.3    Tiến hành triển khai Prolog trên SWI-Prolog:.....   | 8  |
| 2.4    Một số vấn đề khác khi làm việc với SWI – Prolog:.....  | 14 |
| III.    Xây dựng cây phả hệ cho gia đình hoàng gia Anh: .....  | 15 |
| Phần 2: Xây dựng cơ sở tri thức với công cụ Prolog .....   | 19 |
| I.    Cơ sở tri thức hệ thống hành chính Việt Nam .....  | 19 |
| Phần 3. Cài đặt hệ thống suy diễn logic bằng ngôn ngữ lập trình .....  | 26 |
| I.    Cài đặt hệ suy diễn logic cho cơ sở tri thức phả hệ hoàng gia Anh:.....  | 26 |
| II.    Cài đặt hệ suy diễn logic cho cơ sở chi thức phân cấp hành chính Việt Nam: .....  | 27 |

---

## Phần 1. Làm quen với công cụ Prolog

---

### I. Tìm hiểu ngôn ngữ Prolog. Viết báo cáo về các đặc điểm chính của ngôn ngữ. Cần đưa ra nhiều ví dụ minh họa, liên hệ với kiến thức về logic bậc nhất đã học.

#### 1.1 Sơ lược vài nét về ngôn ngữ Prolog:

Prolog là một ngôn ngữ lập trình logic được ứng dụng nhiều trong trí thông minh nhân tạo và ngôn ngữ học tính toán.

Tên gọi Prolog được xuất phát từ cụm từ tiếng Pháp *Programmation en logique*, nghĩa là lập trình theo logic.

Xuất hiện từ năm 1972 (do Alain Colmerauer và Philippe Roussel thiết kế dựa trên thủ tục diễn dịch mệnh đề Horn của Robert Kowalski), mục tiêu của Prolog là giúp người dùng mô tả lại bài toán trên ngôn ngữ của logic, dựa trên đó, máy tính sẽ tiến hành suy diễn tự động dựa vào những cơ chế suy diễn có sẵn (hợp nhất, quay lui và tìm kiếm theo chiều sâu) để tìm câu trả lời cho người dùng.



Hình 1. Chân dung Alain Colmerauer (bên trái) và Robert Kowalski (bên phải)

#### 1.2 Cú pháp:

Chương trình Prolog mô tả các quan hệ, được xác định bằng các mệnh đề. Prolog thuần được biểu diễn dưới dạng mệnh đề Horn. Có 2 loại mệnh đề: **dữ kiện** và **luật**.

- Luật

Một **luật** có dạng:

**Head:-Body.**

Trong đó **Head** là một vị từ logic, còn **Body** có thể là rỗng hoặc một tập vị từ logic.

Ví dụ:

**Lẻ(X) :- X chia\_dư 2 = 1**

**Chẵn(X) :- X chia\_dư 2 = 0**

**Luật** thường được thể hiện những phát biểu logic của bài toán, ví dụ: Nếu người hít thở thì còn sống:

**Còn\_sống(X):-hít\_thở(x).**

Toán tử ":-" được dịch thô là "nếu", trong logic thì nó đại diện cho toán tử suy ra "<-". Phát biểu trên được phát biểu dưới dạng văn xuôi là "Nếu người X hít thở thì người X còn sống". Tất nhiên, bạn có thể chưa thỏa mãn với phát biểu này và bổ sung vào nó để có một phát biểu chặt chẽ hơn như là:

**Còn\_sống(X):-hít\_thở(X), còn\_ấm(X).**

Dấu phẩy "," trong mệnh đề trên được dịch là toán tử "và"; biến trong Prolog được quy ước bắt đầu là một chữ cái hoa.

### • Dữ kiện

Một mệnh đề mà có **Body** rỗng gọi là 1 **Dữ kiện**. Ví dụ:

**Chuột(jerry).**

Dữ kiện trên tương đương với luật:

**Chuột(jerry):-true.**

Vị từ tích hợp luôn đúng

Phần lớn các trình biên dịch của các chương trình Prolog đều yêu cầu vị từ logic ở phần đầu của một mệnh đề Horn là một vị từ dương (không có dấu phủ định đi kèm), còn các vị từ trong phần Body có thể có dấu phủ định đi kèm. Chương trình logic mà không có sự xuất hiện của dấu phủ định đi kèm gọi là chương trình logic xác định, còn không thì được gọi là chương trình logic thường.

## 1.3 Ngữ nghĩa:

Một chương trình logic có ngữ nghĩa của riêng nó. Ngữ nghĩa quyết định những kết luận "đúng" nào có thể rút ra được từ một chương trình Prolog. Ví dụ một chương trình Prolog gồm một dữ kiện:

**Vịt(donald).**

Khi đó, ta có thể rút ra duy nhất một dữ kiện đúng là "donald là một con vịt".

Trong một ứng dụng Prolog, có thể đưa ra 2 câu trả lời sau từ dữ kiện đầu vào như trên:

?-Vịt(donald).

Yes.

Và

?-Vịt(X).

X = donald;

No.

Trong ví dụ trên, "no" có nghĩa là không còn câu trả lời nào nữa. Mọi câu hỏi khác đều cho trả lời là sai. Điều này có nghĩa là trong một chương trình Prolog, người ta sử dụng giả thiết thế giới đóng, mọi thứ bạn khai báo là đúng, nếu không thì nó là sai. Vì vậy trong ví dụ trên, khi bạn hỏi "tom có phải là một con vịt hay không", bạn sẽ nhận được câu trả lời "no".

?-Vịt(tom).

no.

Với một chương trình Prolog xác định, ngữ nghĩa của nó được định nghĩa là một mô hình tối thiểu của nó.

Với một chương trình Prolog bình thường, có nhiều loại ngữ nghĩa được sử dụng như ngữ nghĩa đầy đủ, ngữ nghĩa tối thiểu, ngữ nghĩa hoàn chỉnh.

#### 1.4 Thực thi:

Việc thực thi chương trình Prolog được bắt đầu bằng việc người dùng nhập một mục tiêu duy nhất, được gọi là truy vấn. Về mặt logic, công cụ Prolog cố gắng tìm ra một giải pháp bằng cách bác bỏ truy vấn bị phủ định (tương tự như chứng minh bằng phản chứng). Phương pháp được Prolog sử dụng được gọi là **SLD resolution**.

Nếu truy vấn bị phủ định có thể bị bác bỏ, thì truy vấn đó, với các ràng buộc biến thích hợp, là một hệ quả logic của chương trình. Thì tất cả các liên kết biến được tạo đều được báo cáo cho người dùng và truy vấn được cho là đã thành công.

Về mặt hoạt động, chiến lược thực thi của Prolog có thể được coi là sự tổng hợp của các lệnh gọi hàm trong các ngôn ngữ khác, điểm khác biệt là nhiều đầu mệnh đề có thể khớp với một lệnh gọi nhất định. Trong trường hợp đó, hệ thống tạo ra một "điểm lựa chọn", thống nhất mục tiêu với các đầu mệnh đề của phương án đầu tiên và tiếp tục với các mục tiêu của phương án đầu tiên đó. Nếu bất kỳ mục tiêu nào không thành công trong quá trình thực thi chương trình, tất cả các ràng buộc biến được thực hiện kể từ khi điểm lựa chọn gần đây nhất được tạo sẽ được hoàn tác và việc thực thi tiếp tục với sự thay thế tiếp theo của điểm lựa chọn đó. Chiến lược thực thi này được gọi là *quay lui* theo trình tự thời gian.

Ví dụ:

```

mother_child(trude, sally).
father_child(tom, sally).
father_child(tom, erica).
father_child(mike, tom).

sibling(X, Y) :- parent_child(Z, X), parent_child(Z, Y).
parent_child(X, Y) :- father_child(X, Y).
parent_child(X, Y) :- mother_child(X, Y).

```

Như vậy kết quả của câu truy vấn dưới đây sẽ trả về true:

```

?- sibling(sally, erica).
Yes

```

Kết quả của câu truy vấn được thu về nhờ quá trình như sau:

Ban đầu, đầu mệnh đề phù hợp duy nhất cho anh chị em truy vấn **sibling(sally, erica)** là mệnh đề đầu tiên, vì vậy việc chứng minh truy vấn tương đương với việc chứng minh phần thân của mệnh đề đó với các ràng buộc biến thích hợp tại chỗ, kết hợp với **(parent\_child(Z,sally), parent\_child(Z,erica))**. Bước tiếp theo được chứng minh là **parent\_child(Z, sally)**. 2 đầu mệnh đề phù hợp với mục tiêu này. Hệ thống tạo ra một điểm lựa chọn và thử phương án thay thế đầu tiên, có phần thân là **father\_child(Z, sally)**. Điều này có thể được chứng minh bằng mệnh đề **father\_child(tom, sally)**, nên ràng buộc **Z = tom** được tạo ra và mục tiêu tiếp theo là chứng minh phần thứ 2 của kết hợp: **parent\_child(tom,erica)**. Một lần nữa, điều này có thể được chứng minh bằng các mệnh đề. Vì tất cả các mục tiêu có thể được chứng minh, truy vấn thành công. Vì truy vấn không chứa biến nên không có ràng buộc nào được báo cáo cho người dùng.

Một truy vấn với biến, ví dụ:

```

?- father_child(Father, Child).

```

Sẽ liệt kê tất cả các câu trả lời hợp lệ thông qua phương pháp quay lui.

### 1.5 Vòng lặp và đệ quy:

Các thuật toán lặp lại có thể được thực hiện bằng các vị từ đệ quy.

### 1.6 Liên hệ giữa Prolog và logic bậc nhất:

Về cơ bản, logic bậc nhất là logic biểu diễn kết hợp thêm 2 vị từ  $\forall$  và  $\exists$ .

Mệnh đề logic bậc nhất :

$$u \leftarrow (p \wedge q \wedge \dots \wedge t)$$

Được viết lại trong Prolog như sau:

**u :- p, q, ..., t.**

Mệnh đề này có nghĩa là “u đúng nếu p,q,...,t đều đúng”.

Phép  $\wedge$  có thể được biểu diễn bằng dấu “,” trong Prolog.

Trong logic bậc nhất, mệnh đề **love(john, mary)** có thể được biểu diễn lại trong Prolog gần như tương tự:

**love(john,mary)**

Theo logic bậc nhất, mệnh đề **man(x)** có thể được biểu diễn trong Prolog **man(x)**.

Bây giờ, mệnh đề  $\forall x \text{ man}(x)$  có thể được biểu diễn trong Prolog:

**forall(x,man(x))**

Và mệnh đề  $\exists x \text{ man}(x)$  có thể được biểu diễn trong Prolog:

**exist(x,man(x))**

Dấu phủ định  $\neg$  trong prolog được biểu diễn dưới dạng tiền tố **!** ví dụ:

**legal(X) :- ! illegal(X).**

Phép toán  $\vee$  trong mệnh đề có thể được biểu diễn như sau:

Ví dụ “Nếu giàu hay nổi tiếng sẽ vui”

Logic bậc nhất:  $\text{vui}(x) \leftarrow \text{giàu}(x) \vee \text{nổi\_tiếng}(x)$

Prolog:

**vui(x):- giàu(x)**

**vui(x):-nổi\_tiếng(x)**

Hay

**vui(x):- giàu(x); nổi\_tiếng(x)**

Đối với mệnh đề có  $\leftrightarrow$  có thể chuyển thành  $\leftarrow$

Ví dụ:

$$a \leftrightarrow b \equiv (a \rightarrow b) \wedge (b \rightarrow a)$$

## II. Tìm hiểu một môi trường lập trình Prolog. Viết báo cáo về cách thức triển khai ngôn ngữ Prolog trên công cụ đã tìm hiểu. Trình bày ít nhất 5 ví dụ minh họa.

Có nhiều môi trường lập trình Prolog khác nhau như: , DOS-PROLOG, Open Prolog, Ciao Prolog, GNU Prolog, SWI-Prolog. Ở đây nhóm chọn môi trường **SWI-Prolog**.

### 2.1 Vài nét về môi trường SWI-Prolog:

SWI-Prolog là một môi trường lập trình miễn phí của ngôn ngữ lập trình Prolog, thường được sử dụng để giảng dạy và các ứng dụng semantic web. Nó có một bộ tính năng phong phú, thư viện để lập trình logic ràng buộc, đa luồng, kiểm tra đơn vị, GUI, giao diện với Java, ODBC và những thứ khác, lập trình ngữ nghĩa, web server, SGML, RDF, RDFS, các công cụ dành cho nhà phát triển (bao gồm một IDE với một GUI debugger và GUI profiler), và các tài liệu mở rộng.

SWI-Prolog chạy trên các nền tảng Unix, Windows, Macintosh và Linux.

SWI-Prolog đã được phát triển liên tục kể từ năm 1987. Tác giả chính của nó là Jan Wielemaker.

Tên gọi SWI có nguồn gốc từ Sociaal-Wetenschappelijke Informatica ("Social Science Informatics"), là tên trước của đại học Amsterdam, nơi Wielemaker là nhân viên.

### 2.2 Hướng dẫn cài đặt:

Truy cập trang chủ của SWI- Prolog : <https://www.swi-prolog.org/>

Khuyến cáo : nên tải phiên bản stable mới nhất phù hợp với hệ điều hành để tránh phát sinh lỗi trong quá trình sử dụng.

Tiến hành cài đặt như bình thường.

### 2.3 Tiến hành triển khai Prolog trên SWI-Prolog:

Có 2 cách chính để triển khai Prolog là:

- Load file chứa các lệnh Prolog vào SWI-Prolog
- Nạp dữ liệu thông qua Console của SWI-Prolog

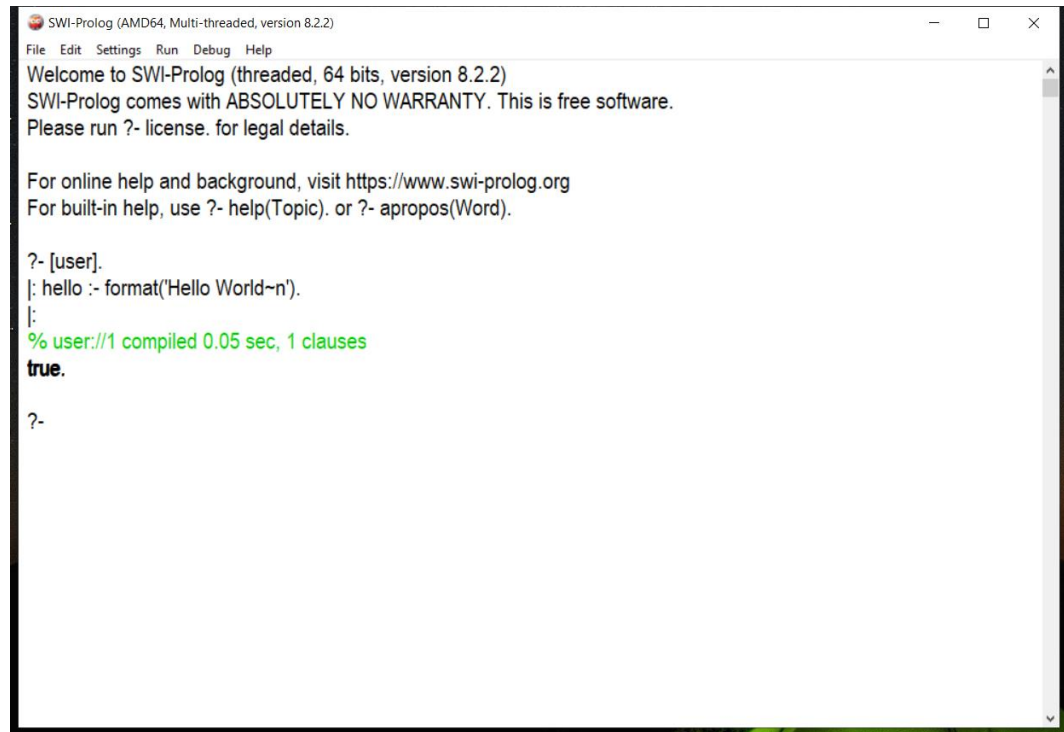
Mặc dù chính những người tạo ra SWI-Prolog cũng khuyến cáo với chúng ta rằng nên cho chương trình của bạn vào file, để tiện chỉnh sửa và reload nó vào SWI-Prolog. Nhưng ở đây vẫn trình bày cách nạp các luật và dữ kiện thông qua cửa sổ Console của SWI-Prolog.

#### a) Nạp dữ liệu thông qua Console của SWI-Prolog:

Cách thuận tiện nhất để nạp các mệnh đề vào cửa sổ Console của SWI là thông qua tệp giả **user**. Input nhập vào kết thúc bằng các sử dụng ký tự hệ thống end-of-file.

Chẳng hạn, nhập mệnh đề hello vào console:





```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
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).

?- [user].
|: hello :- format('Hello World~n').
|:
% user://1 compiled 0.05 sec, 1 clauses
true.

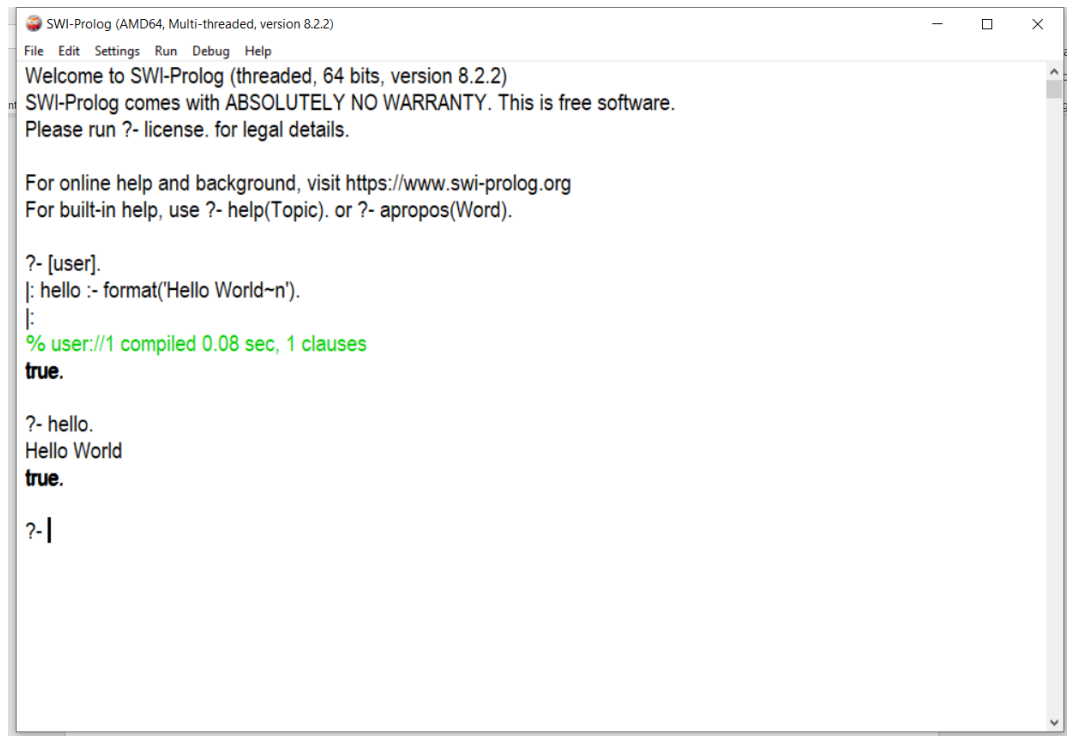
?-
```

Hình 2. Nhập mệnh đề hello vào cửa sổ console thông qua từ khóa user

Lưu ý : khi kết thúc nhập cần bấm tổ hợp phím ctrl + D thay cho ký tự end-of-file đối với hệ điều hành Windows.

Kết thúc một mệnh đề trong SWI – Prolog là dấu “.”.

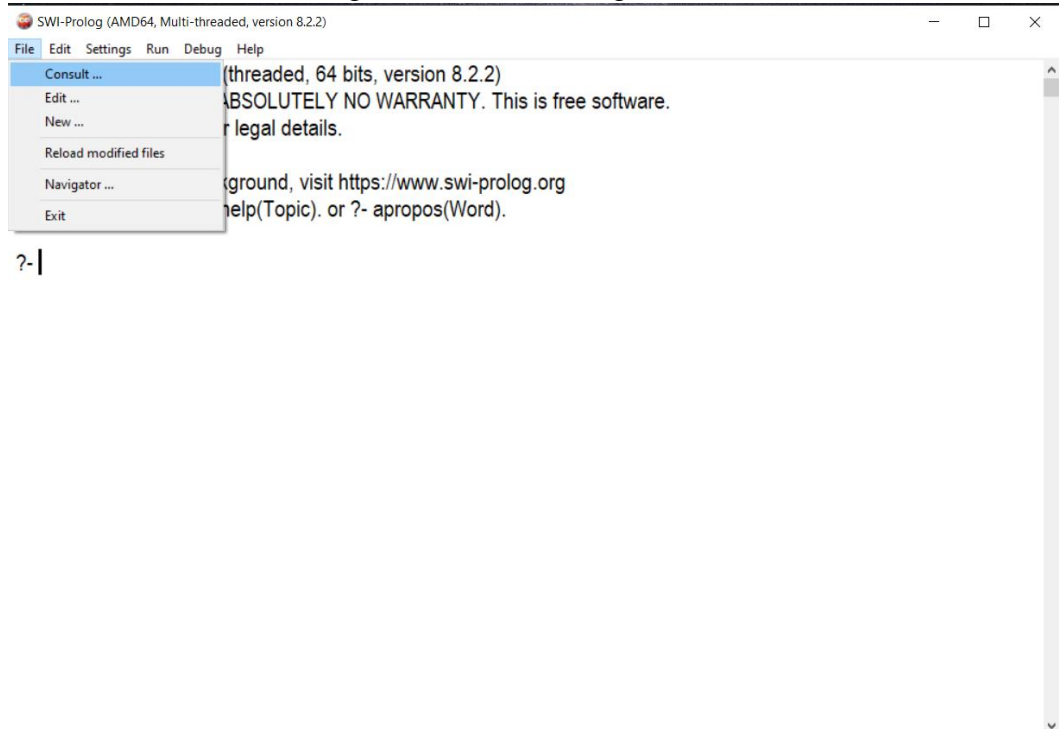
Đối với ví dụ trên, khi truy vấn **hello**. Sẽ trả về kết quả “hello world” kèm theo true.



Hình 3. Kết quả của câu truy vấn hello.

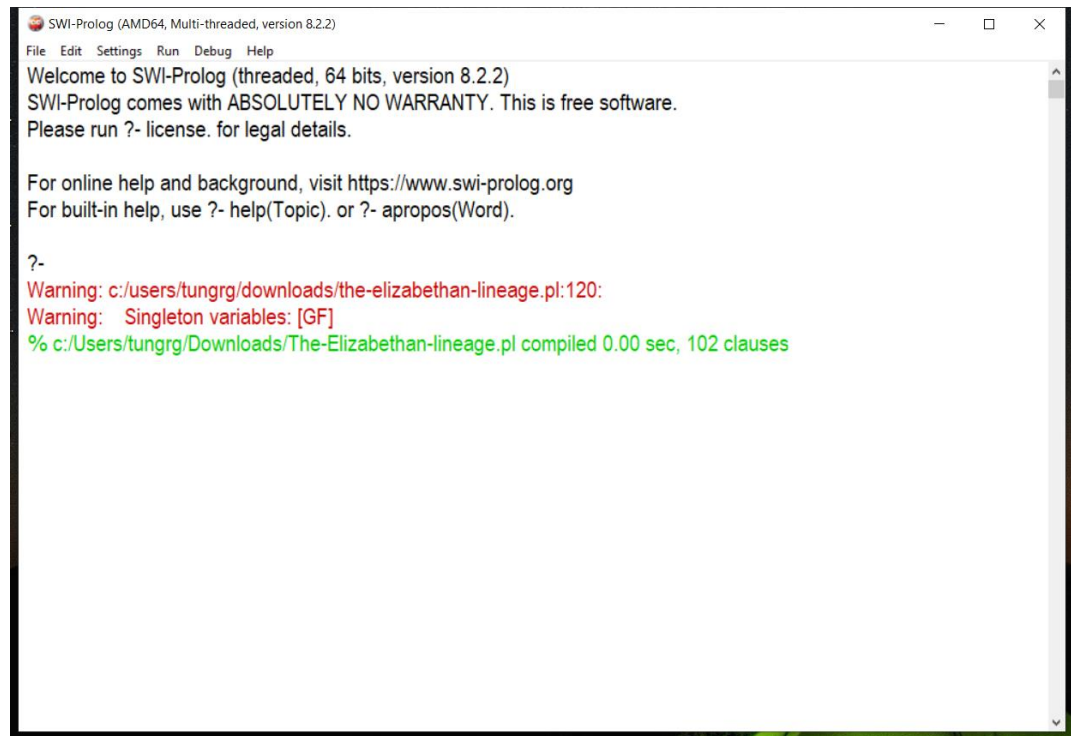
b) Load file chương trình vào SWI-Prolog:

Bước 1: Chọn consult trong File trên thanh công cụ



Hình 4. Chọn Consult trong File trên thanh công cụ.

Bước 2: Chọn file .pl bạn cần load, ở đây là file “the-Elizabethan-lineage.pl”



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

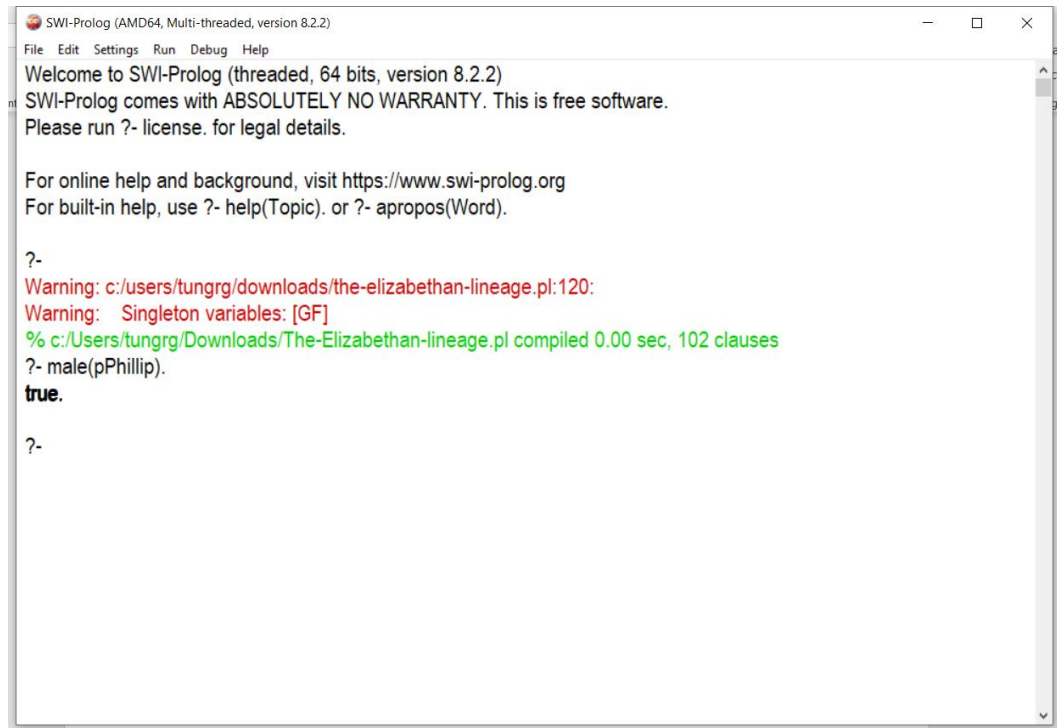
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
Warning: c:/users/tungrg/downloads/the-elizabethan-lineage.pl:120:
Warning: Singleton variables: [GF]
% c:/Users/tungrg/Downloads/The-Elizabethan-lineage.pl compiled 0.00 sec, 102 clauses
```

Hình 5. Kết quả hiện trên console sau khi consult file the-Elizabethan-lineage.pl

Tiến hành truy vấn hay xử lý, ở đây thử gọi câu truy vấn **male(pPhillip).**

Câu truy vấn này có nghĩa là : “có tồn tại một nam tên định danh pPhillip hay không”



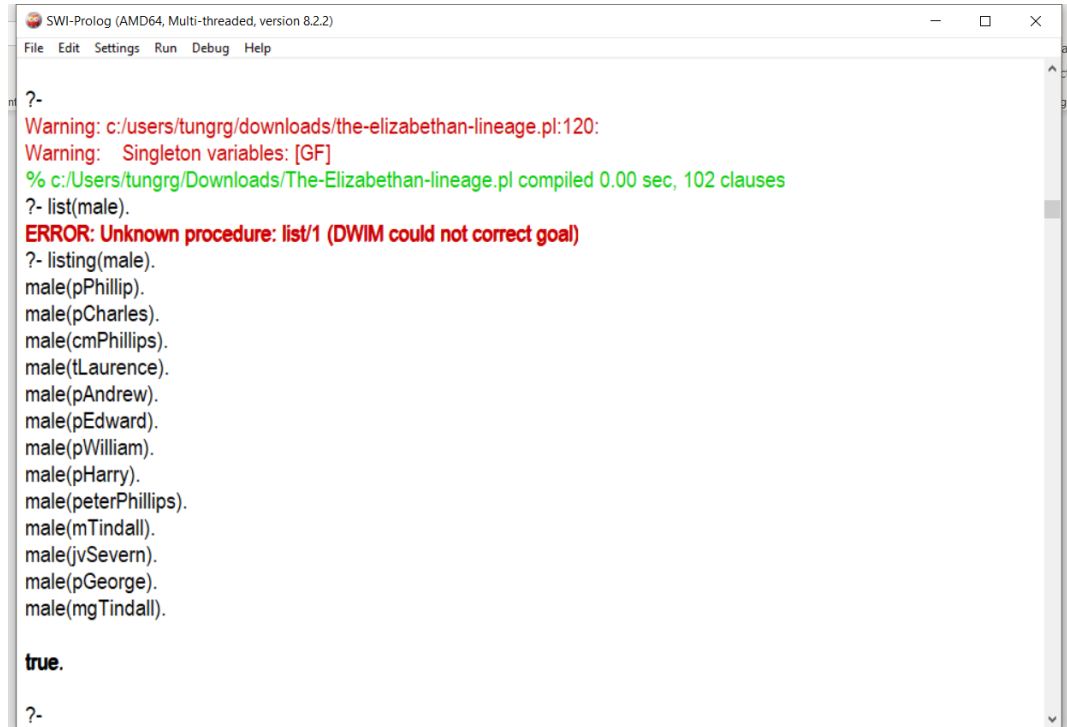
```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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

?-
Warning: c:/users/tungrg/downloads/the-elizabethan-lineage.pl:120:
Warning: Singleton variables: [GF]
% c:/Users/tungrg/Downloads/The-Elizabethan-lineage.pl compiled 0.00 sec, 102 clauses
?- male(pPhillip).
true.
?-
```

Hình 6. Kết quả câu truy vấn `male(pPhillip)` trả về `true`.

Sử dụng câu truy vấn `listing(male)` để liệt kê tất cả `male` :

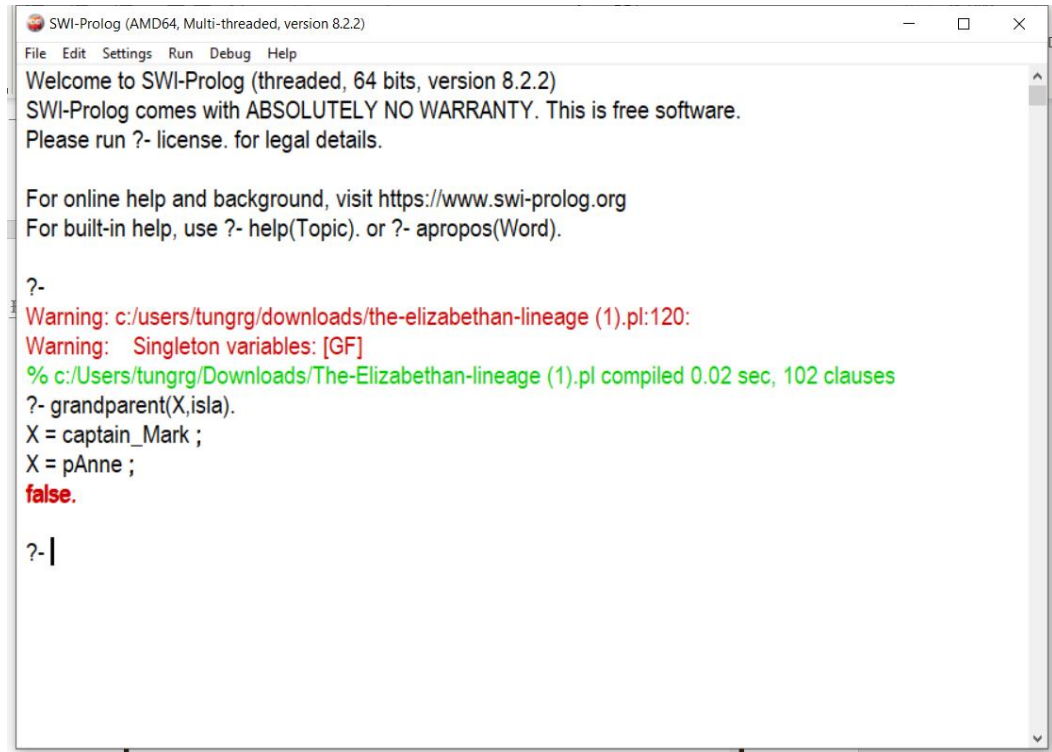


```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Warning: c:/users/tungrg/downloads/the-elizabethan-lineage.pl:120:
Warning: Singleton variables: [GF]
% c:/Users/tungrg/Downloads/The-Elizabethan-lineage.pl compiled 0.00 sec, 102 clauses
?- list(male).
ERROR: Unknown procedure: list/1 (DWIM could not correct goal)
?- listing(male).
male(pPhillip).
male(pCharles).
male(cmPhillips).
male(tLaurence).
male(pAndrew).
male(pEdward).
male(pWilliam).
male(pHarry).
male(peterPhillips).
male(mTindall).
male(jvSevern).
male(pGeorge).
male(mgTindall).
true.
?-
```

Hình 7. Câu truy vấn `Listing(male)` liệt kê tất cả `male` trong gia phả.

Sử dụng câu truy vấn **grandparent(X, isla)** để tìm ông bà của isla.

Lưu ý, sau khi bấm **'enter'** để thực thi câu truy vấn Prolog trả ta về kết quả đầu tiên, nếu đó không phải là kết quả duy nhất (isla có nhiều ông hoặc bà) thì có thể bấm phím **','** để Prolog trả về các kết quả tiếp đó. Sau kết quả cuối cùng thì Prolog sẽ trả về **'false.'**



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

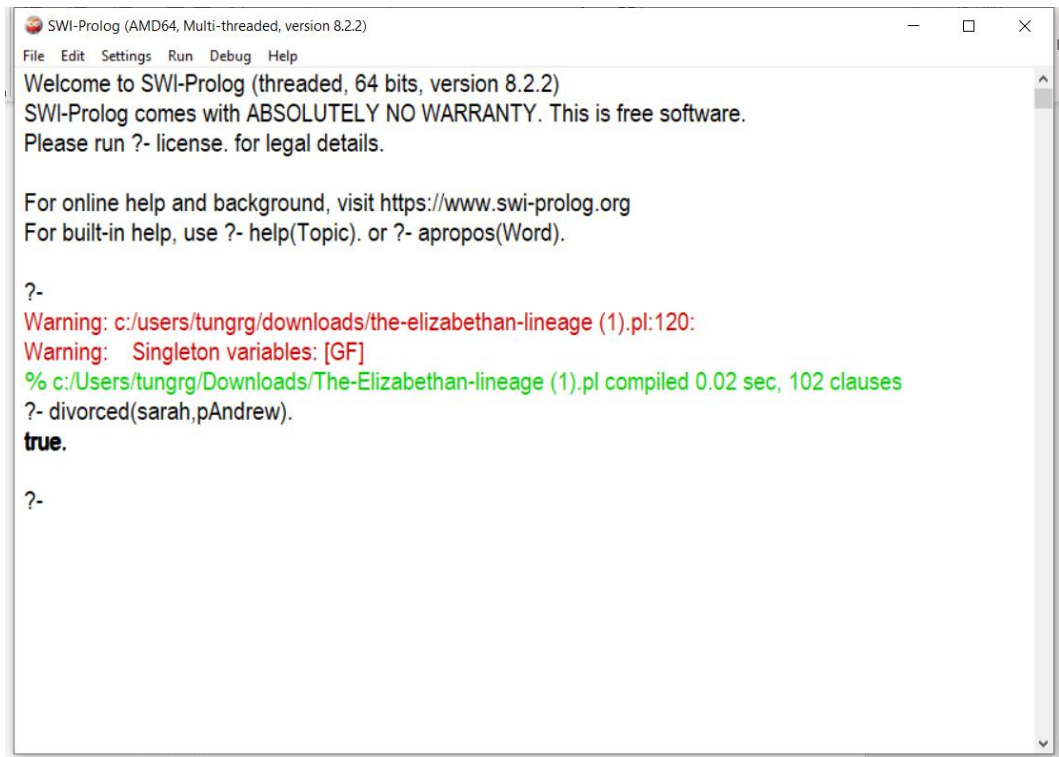
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
Warning: c:/users/tungrg/downloads/the-elizabethan-lineage (1).pl:120:
Warning: Singleton variables: [GF]
% c:/Users/tungrg/Downloads/The-Elizabethan-lineage (1).pl compiled 0.02 sec, 102 clauses
?- grandparent(X, isla).
X = captain_Mark ;
X = pAnne ;
false.

?- |
```

Hình 9. Sử dụng câu truy vấn **grandparent(X, isla)** để tìm ông bà của isla

Sử dụng câu truy vấn **divorced(sarah, pAndrew)** để kiểm tra sarah và pAndrew đã li dị chưa:



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
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).

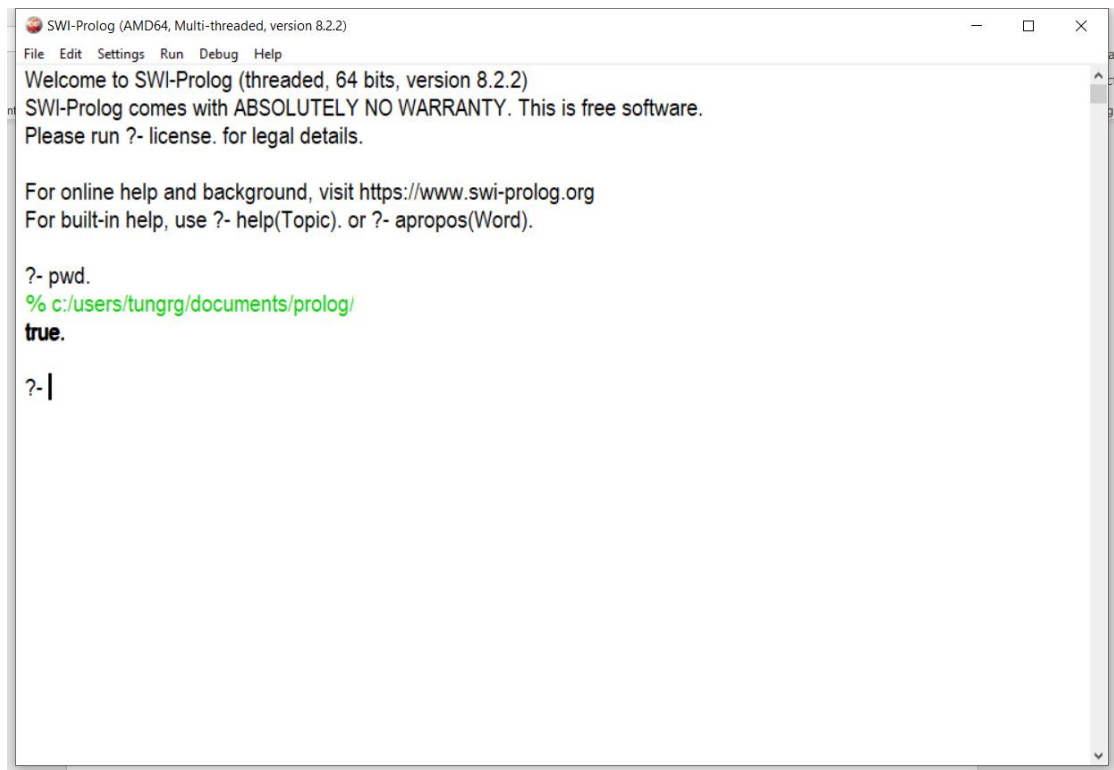
?-
Warning: c:/users/tungrg/downloads/the-elizabethan-lineage (1).pl:120:
Warning: Singleton variables: [GF]
% c:/Users/tungrg/Downloads/The-Elizabethan-lineage (1).pl compiled 0.02 sec, 102 clauses
?- divorced(sarah,pAndrew).
true.
?-
```

Hình 10. Kết quả true trả về cho câu truy vấn `divorced(sarah,pAndrew)`

## 2.4 Một số vấn đề khác khi làm việc với SWI – Prolog:

### a) Xem và thay đổi thư mục làm việc:

Thao tác xem và thay đổi thư mục làm việc có thể được thực hiện thông qua 2 lệnh **pwd** và **cd** tại cửa sổ console.



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
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).

?- pwd.
% c:/users/tungrg/documents/prolog/
true.

?- |
```

Hình 11. Thông qua lệnh `pwd` để biết thư mục làm việc hiện tại.

b) Dừng Prolog:

Có thể dừng Prolog thông qua 2 cách: dùng tổ hợp phím “Ctrl + D” hay thực thi câu lệnh **Halt**.

### III. Xây dựng cây phả hệ cho gia đình hoàng gia Anh:

a) Bộ 20 câu hỏi hệ tri thức cùng với câu truy vấn tương ứng trong Prolog:

1. Ai là ông bà của Isla Phillips?

?- grandparent(X,iPhillips).

2. Ai là cháu của Queen Elizabeth II?

?- child(X,qElizabethII).

3. Prince William và Prince Harry có phải là anh em?

?- sibling(pWilliam,pHarry).

4. Prince William và Princess Beatrice có phải là anh em?

?- sibling(pWilliam,pBeatrice).

5. Peter Phillips có con trai không?

?- son(X,peter).

6. Kate Middleton có con gái không?

?- daughter(X,peter).

7. Ai là cậu của Isla Phillips?

?- uncle(X,isla).

8. Ai là dì của Prince William?

?- aunt(X,pWilliam).

9. Ai là mẹ của Princess Beatrice?

?- mother(X,pBeatrice).

10. Vợ của Peter Phillips là ai?

?- wife(X,peter).

11. Timothy Laurence có phải là chồng của Princess Diana?

?- husband(timothy,pDiana).

12. Prince Phillip có phải là ông của Zara Phillips?

?- grandfather(pPhillip,zara).

13. Ai là bà của James Viscount Severn?

?- grandmother(X,james).

14. Cháu trai của Camilla Parker Bowles là ai?

?- nephew(X,pCharles).

15. Mia Grace Tindall có phải cháu gái của Queen Elizabeth II?

?- niece(mia\_Grace,qElizabethII).

16. Anh (em) trai của Prince Edward là ai?

?- brother(X,pEdward).

17. Princess Charlotte có phải cháu gái của Savannah Phillips?

?- niece(pCharlotte,savannah).

18. Lady Louise và James có phải chị em không?

?- sister(lady\_Louise,james).

19. Princess Diana và Prince Charles có phải là vợ chồng?

?- married(pDiana,pCharles).

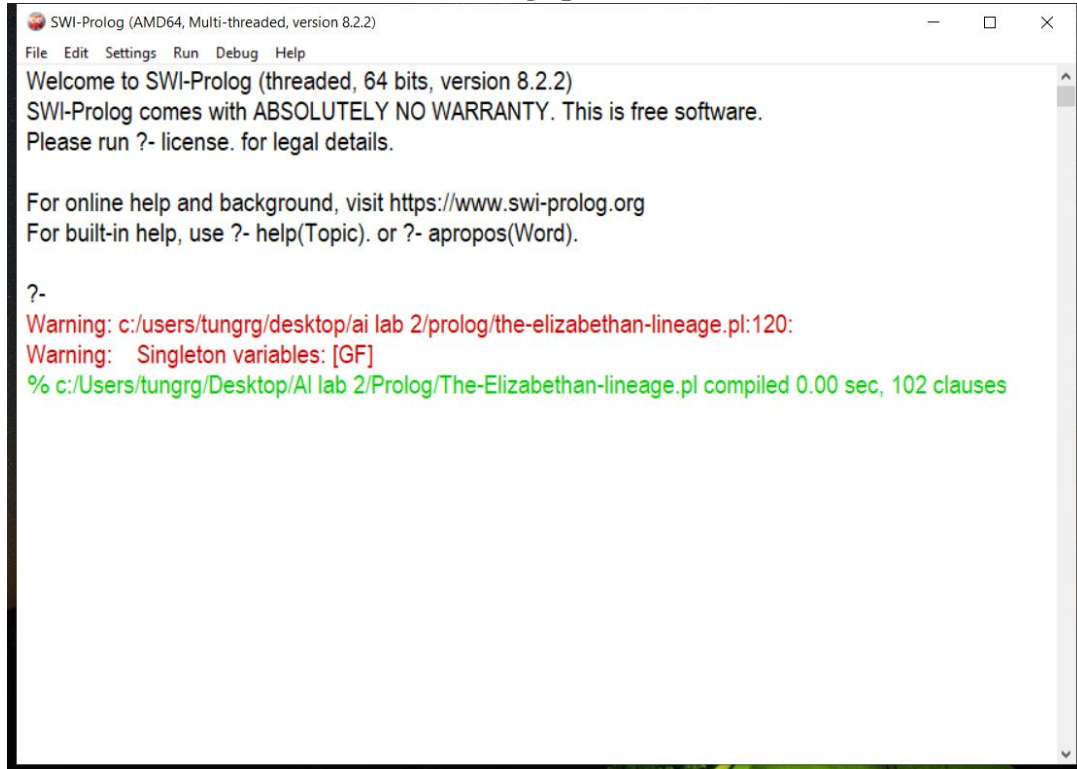


20. Có phải Sarah Ferguson và Prince Andrew đã li hôn?

?- divorced(sarah,pAndrew).

b) Cách chạy thử nghiệm trong Prolog:

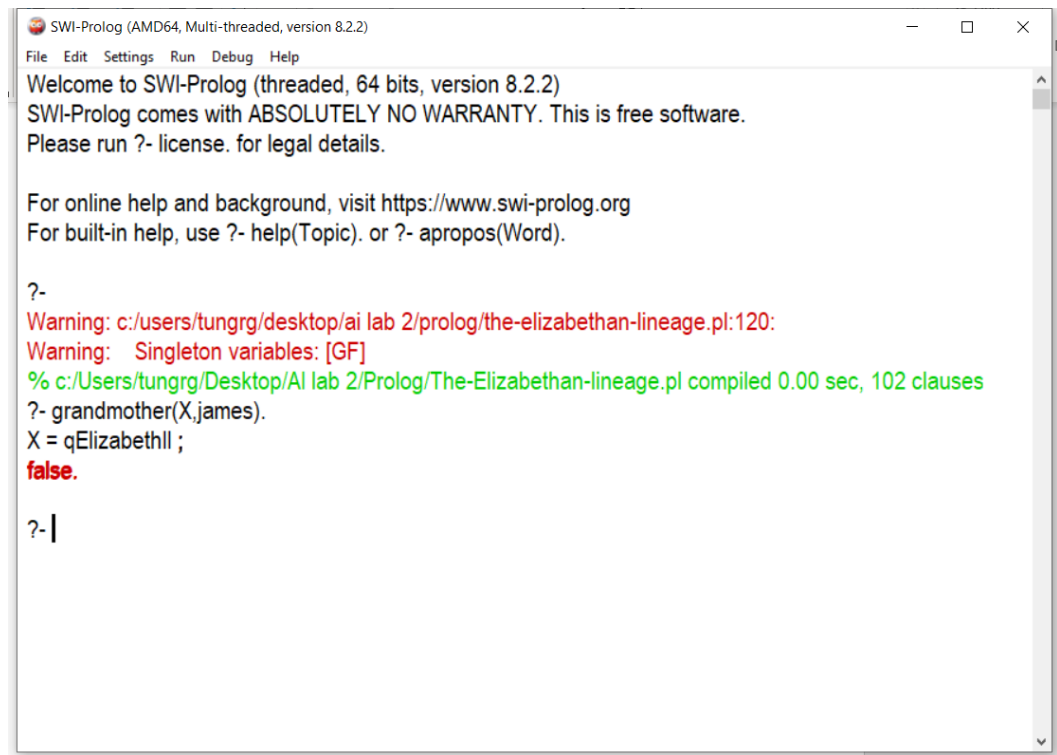
B1: Consult file **The-Elizabethan-lineage.pl**.



Hình 12. Consult file The-Elizabethan-lineage.pl

B2: Bắt đầu thực hiện các câu truy vấn trong bộ câu hỏi.

Vd: Thực hiện câu truy vấn ?- grandmother(X,james).



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.2)
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).

?-
Warning: c:/users/tungrg/desktop/ai lab 2/prolog/the-elizabethan-lineage.pl:120:
Warning: Singleton variables: [GF]
% c:/Users/tungrg/Desktop/AI lab 2/Prolog/The-Elizabethan-lineage.pl compiled 0.00 sec, 102 clauses
?- grandmother(X,james).
X = qElizabethII ;
false.
?- |
```

Hình 13. Kết quả trả về của câu truy vấn `?- grandmother(X,james).`

---

## Phần 2: Xây dựng cơ sở tri thức với công cụ Prolog

---

### I. Cơ sở tri thức hệ thống hành chính Việt Nam

Cơ sở tri thức hệ thống hành chính được xây dựng trên các vị từ cơ bản:

country(<tên quốc gia>)

region(<tên khu vực>)

province(<tên tỉnh>)

city(<tên thành phố >)

district(<tên huyện>)

ward(<tên phường>)

commune(<tên xã>)

Định nghĩa các vị từ sau từ các vị từ đã có ở trên:

include(country,region)

include(region,province)

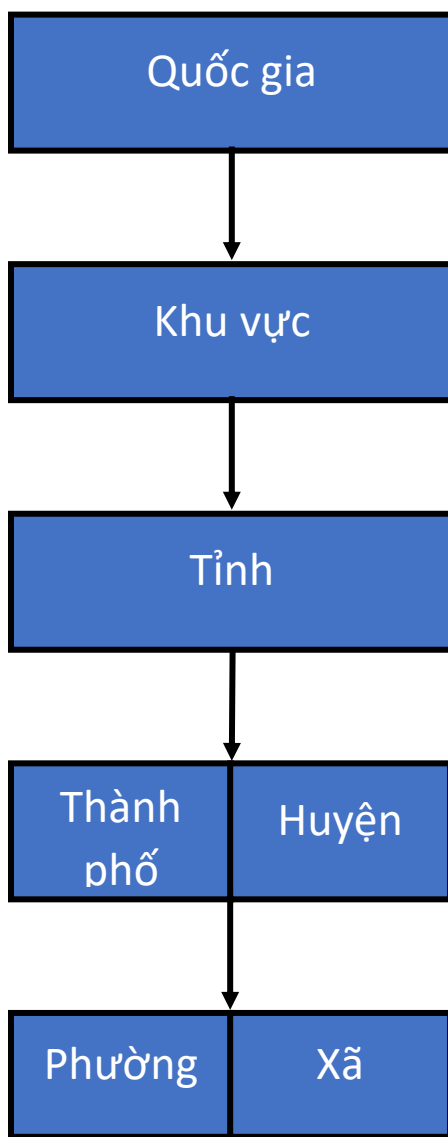
include(province,city)

include(city,district)

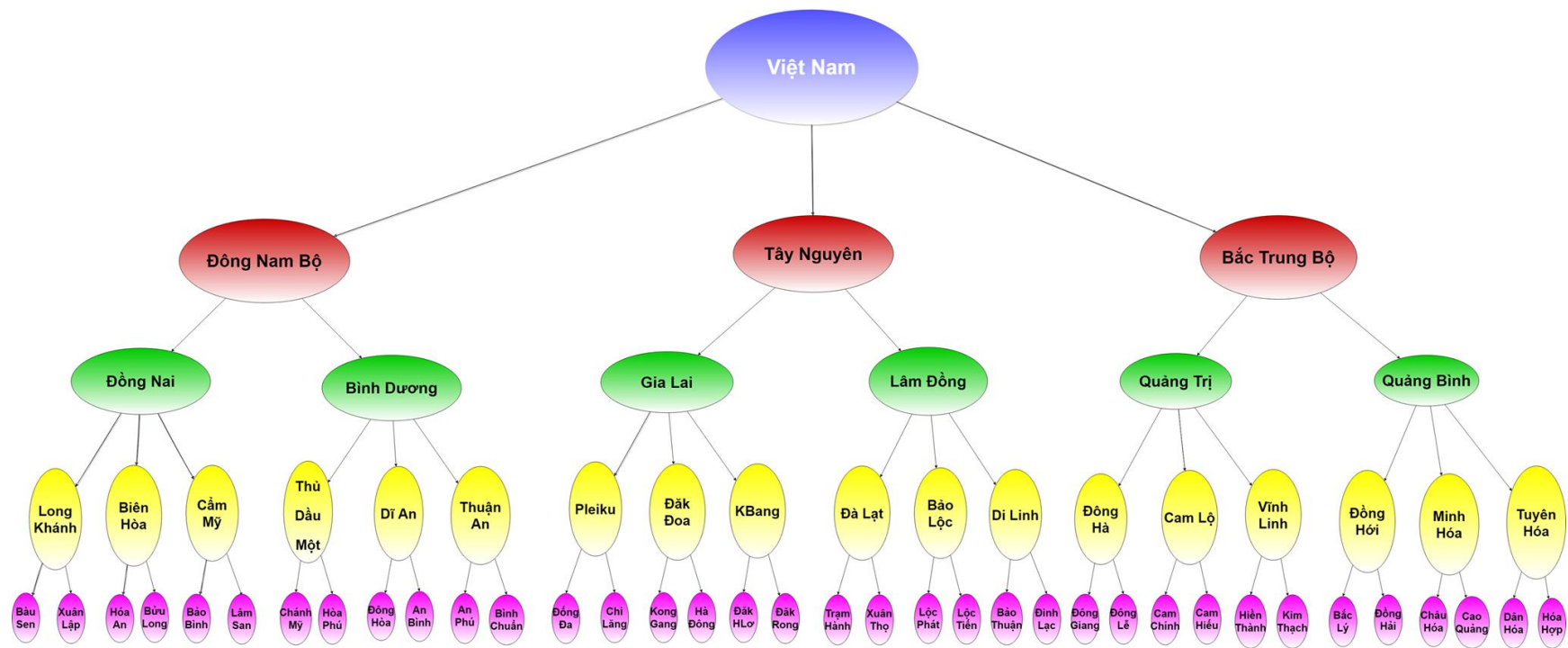
include(district,ward)

include(district,commune)

Mô hình phân cấp của khu vực hành chính:



Mô hình hành chính được trực quan hóa cụ thể của hệ cơ sở tri thức:



Hình 14.Trực quan hóa cụ thể của mô hình hành chính

a) Bộ 20 câu hỏi cho hệ cơ sở tri thức và câu truy vấn trong Prolog tương ứng:

1. Tỉnh nào bao gồm thành phố Long Khánh?

?- include(X,longKhanh).

2. Huyện Kbang trực thuộc tỉnh nào?

?- belongsto(kbang,X).

3. Tỉnh Đồng Nai có bao gồm thành phố Long Khánh không?

?- includes(dongNai,longKhanh).

4. Tỉnh Đồng Nai có bao gồm phường Bàu Sen không?

?- includes(dongNai,bauSen).

5. Bàu sen và Xuân Lập thuộc cùng một thành phố không?

?- sibling(bauSen,xuanLap).

6. Tìm thành phố cùng tỉnh với thành phố Long Khánh.

?- sibling(longKhanh,X).

7. Tỉnh Đồng Nai và tỉnh Gia Lai có thuộc cùng một khu vực không?

?- sibling(dongNai,giaLai).

8. Thành phố Dĩ An trực thuộc tỉnh nào?

?- belongsto(diAn,X).

9. Tỉnh Bình Dương có bao gồm phường An Phú không?

?- includes(binhDuong,anPhu).

10. Tỉnh Bình Dương có bao gồm thành phố Pleiku không?

?- include(binhDuong,pleiku).

11. Tìm khu vực thành phố Đồng Nai trực thuộc.

?- belongto(dongNai,X).

12. Việt Nam có tỉnh Lâm Đồng không?

?- includes(vietNam,lamDong).

13. Thành phố Đà Lạt và thành phố Đông Hà có thuộc cùng một tỉnh không?

?- sibling(daLat,dongHa).

14. Tìm các phường cùng thành phố với phường Trạm Hành.

?- sibling(tramHanh,X).

15.Khu vực Tây Nguyên và Bắc Trung Bộ có thuộc cùng một quốc gia không?

?- sibling(tayNguyen,bacTrungBo).

16.Tìm xã cùng huyện với xã Cam Hiếu.

?- sibling(camHieu,X).

17.Xã Cam hiếu có trực thuộc thành phố Hội An hay không?

?- belongsto(camHieu,diAn).

18.Quảng Trị bao gồm các đơn vị hành chính nào?

?- includes(quangTri,X).

19.Huyện Cam Lộ trực thuộc các khu vực hành chính nào?

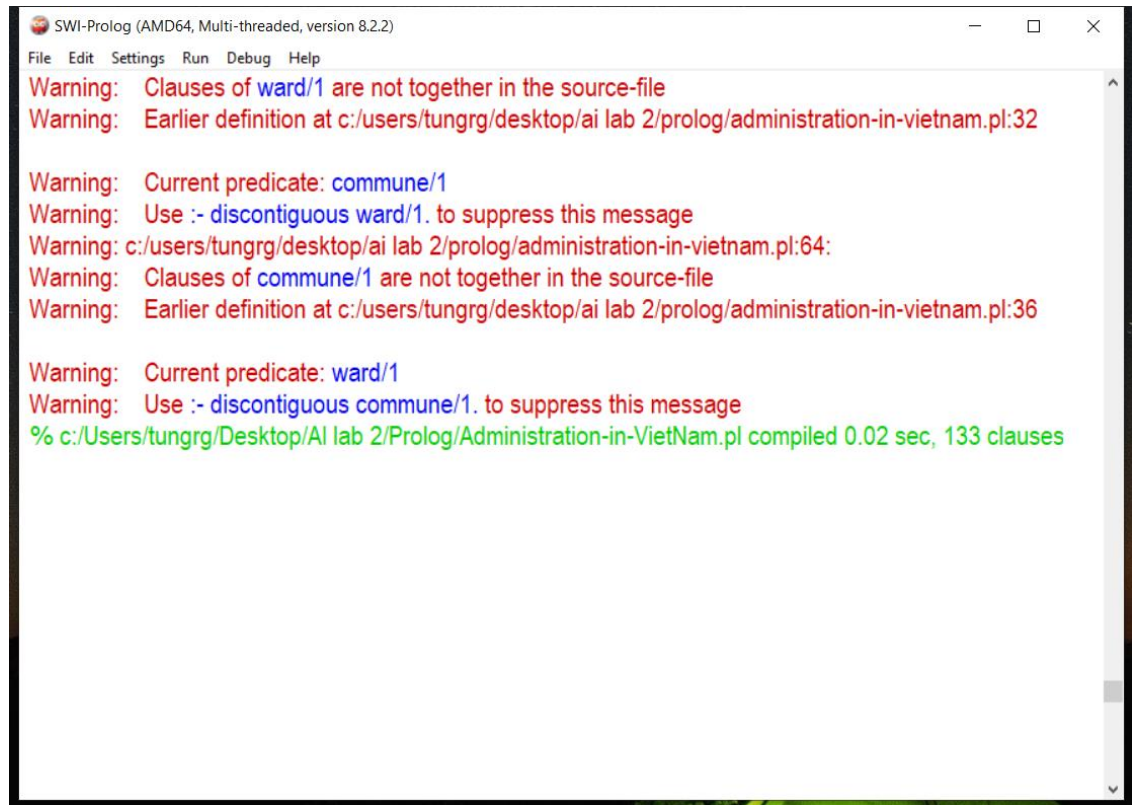
?- belongsto(camLo,X).

20. Xã Kim Thạch và xã Hiền Thanh có cùng một huyện hay không?

?- sibling(kimThach,hienThanh).

b) Cách chạy thử nghiệm trong Prolog:

B1: Consult file **Administration-in-VietNam.pl**



```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Warning: Clauses of ward/1 are not together in the source-file
Warning: Earlier definition at c:/users/tungrg/desktop/ai lab 2/prolog/administration-in-vietnam.pl:32

Warning: Current predicate: commune/1
Warning: Use :- disjointiguous ward/1. to suppress this message
Warning: c:/users/tungrg/desktop/ai lab 2/prolog/administration-in-vietnam.pl:64:
Warning: Clauses of commune/1 are not together in the source-file
Warning: Earlier definition at c:/users/tungrg/desktop/ai lab 2/prolog/administration-in-vietnam.pl:36

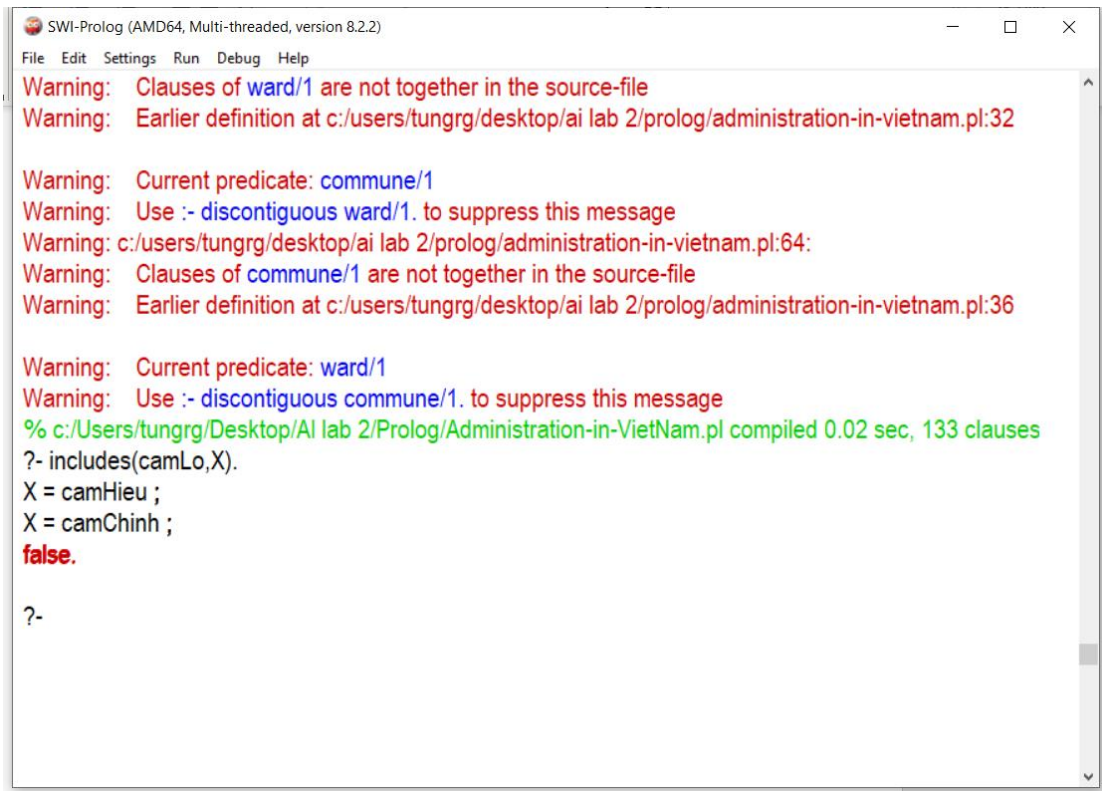
Warning: Current predicate: ward/1
Warning: Use :- disjointiguous commune/1. to suppress this message
% c:/Users/tungrg/Desktop/AI lab 2/Prolog/Administration-in-VietNam.pl compiled 0.02 sec, 133 clauses
```

Hình 15. Consult file Administration-in-Vietnam.pl

B2: Thực hiện các câu truy vấn trong bộ câu hỏi:

Vd: Thực hiện câu truy vấn : ?- includes(camLo,X).





```
SWI-Prolog (AMD64, Multi-threaded, version 8.2.2)
File Edit Settings Run Debug Help
Warning: Clauses of ward/1 are not together in the source-file
Warning: Earlier definition at c:/users/tungrg/desktop/ai lab 2/prolog/administration-in-vietnam.pl:32

Warning: Current predicate: commune/1
Warning: Use :- discontinuous ward/1. to suppress this message
Warning: c:/users/tungrg/desktop/ai lab 2/prolog/administration-in-vietnam.pl:64:
Warning: Clauses of commune/1 are not together in the source-file
Warning: Earlier definition at c:/users/tungrg/desktop/ai lab 2/prolog/administration-in-vietnam.pl:36

Warning: Current predicate: ward/1
Warning: Use :- discontinuous commune/1. to suppress this message
% c:/Users/tungrg/Desktop/AI lab 2/Prolog/Administration-in-VietNam.pl compiled 0.02 sec, 133 clauses
?- includes(camLo,X).
X = camHieu ;
X = camChinh ;
false.
?-
```

Hình 16. Kết quả câu truy vấn `?-includes(camLo,X)`.

## Phần 3. Cài đặt hệ thống suy diễn logic bằng ngôn ngữ lập trình

Ngôn ngữ lập trình : **Python**.

Phương pháp suy diễn : Suy diễn lùi.

### I. Cài đặt hệ suy diễn logic cho cơ sở tri thức phủ hệ hoàng gia Anh:

- **Input**

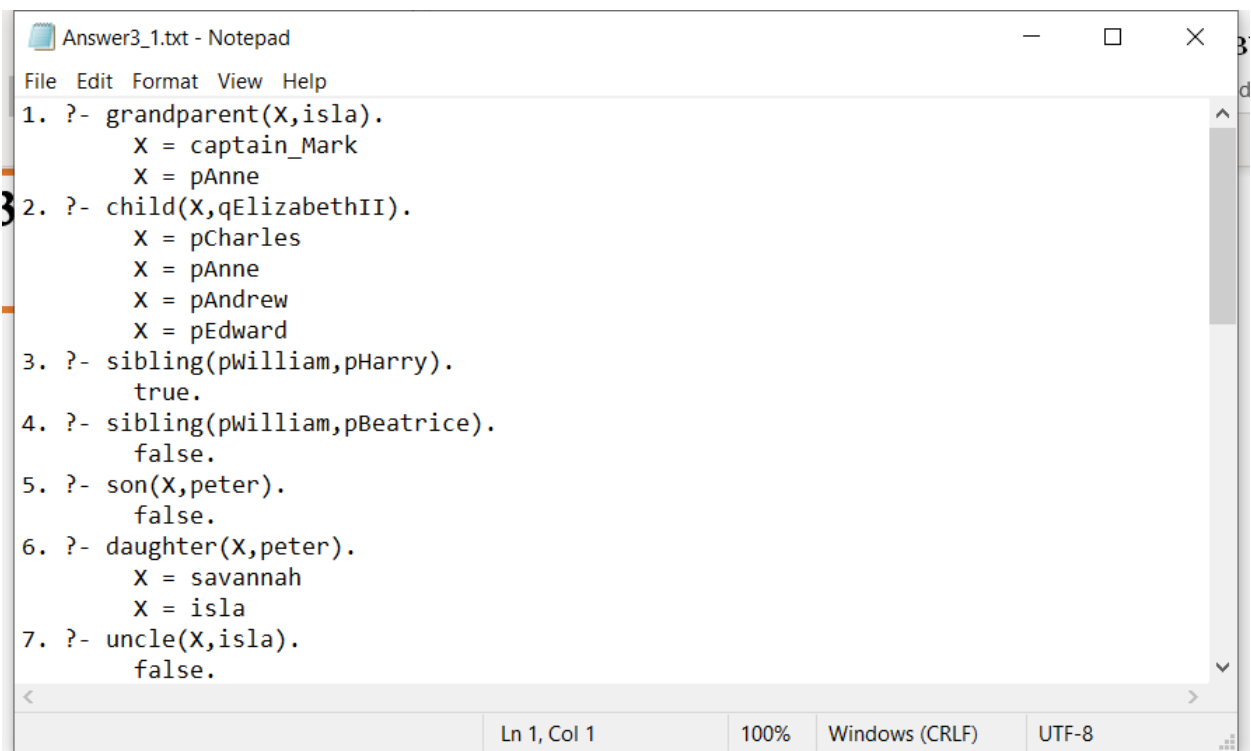
File mã nguồn : **source.py**

Bộ câu hỏi test: **20-Questions.txt**

File cơ sở tri thức: **cayphahe.pl**

- **Lưu ý:** để chạy chương trình thì các file **source.py**, **20-Questions.txt**, **cayphahe.pl** phải được để cùng một thư mục.
- **Các thực thi chương trình :** chạy file **source.py**.
- **Output:**

File **Answer3\_1.txt**, file này có cấu trúc như sau:



```
File Edit Format View Help
1. ?- grandparent(X,isla).
    X = captain_Mark
    X = pAnne
2. ?- child(X,qElizabethII).
    X = pCharles
    X = pAnne
    X = pAndrew
    X = pEdward
3. ?- sibling(pwilliam,pHarry).
    true.
4. ?- sibling(pwilliam,pBeatrice).
    false.
5. ?- son(X,peter).
    false.
6. ?- daughter(X,peter).
    X = savannah
    X = isla
7. ?- uncle(X,isla).
    false.
```

Hình 17. Kết quả file Answer3\_1.txt

Mỗi câu hỏi(truy vấn) đi kèm với đáp án được đánh số thứ tự.

## II. Cài đặt hệ suy diễn logic cho cơ sở tri thức phân cấp hành chính Việt Nam:

- Input:

File mã nguồn : **source2.py**

Bộ câu hỏi test: **20-Questions\_2.txt**

File cơ sở tri thức: **caphanhchinh.pl**

- **Lưu ý:** để chạy chương trình thì các file **source2.py**, **20-Questions\_2.txt**, **cayhanhchinh.pl** phải được để cùng một thư mục.
- **Các thực thi chương trình :** chạy file **source2.py**.
- **Output:**

File **Answer3\_2.txt**, file này có cấu trúc như sau:



```
File Edit Format View Help
1. ?- include(x,longKhanh).
    x = dongNai
2. ?- belongsto(kbang,X).
    X = giaLai
    X = tayNguyen
    X = vietNam
    false.
3. ?- includes(dongNai,longKhanh).
    true.
4. ?- includes(dongNai,bauSen).
    true.
5. ?- sibling(bauSen,xuanLap).
    true.
6. ?- sibling(longKhanh,X).
    X = bienHoa
    X = camMy
    false.
7. ?- sibling(dongNai,giaLai).
    true.
```

Hình 18, Kết quả file Answer3\_2.txt

Mỗi câu hỏi(truy vấn) đi kèm với đáp án được đánh số thứ tự.