

## BÀI 6: LÀM QUEN VỚI MÔI TRƯỜNG PROLOG

### I. MỤC TIÊU:

Sau khi thực hành xong, sinh viên nắm được:

- Viết 1 chương trình Prolog đơn giản, biên dịch và thực thi chương trình.

### II. TÓM TẮT LÝ THUYẾT:

Prolog (PROgramming in LOGic) là ngôn ngữ lập trình ký hiệu (symbolic programming) tương tự các ngôn ngữ lập trình hàm (functional programming), hay lập trình phi số (nonnumerical programming). Prolog do giáo sư người Pháp Alain Colmerauer và nhóm nghiên cứu của ông đề xuất lần đầu tiên tại trường Đại học Marseille đầu những năm 1970. Prolog rất thích hợp để giải quyết các bài toán liên quan đến các đối tượng (object) và mối quan hệ (relation) giữa chúng. Nguyên lý lập trình logic dựa trên các mệnh đề Horn (Horn logic). Một mệnh đề Horn biểu diễn một sự kiện hay một sự việc nào đó là đúng hoặc không đúng, xảy ra hoặc không xảy ra (có hoặc không có, v.v...).

Một chương trình Prolog là một cơ sở dữ liệu gồm các *mệnh đề* được xây dựng từ các vị từ (predicat). Một vị từ là một phát biểu nào đó về các đối tượng có chân trị *đúng* (true) hoặc *sai* (fail). Một vị từ có thể có các đối là *các nguyên logic* (logic atom).

Mỗi nguyên tử biểu diễn một quan hệ giữa các *hạng* (term). Do đó, hạng và quan hệ giữa các hạng tạo thành mệnh đề.

Hạng được xem là những đối tượng “dữ liệu” trong một chương trình Prolog. Hạng có thể là *hạng sơ cấp* (elementary term) gồm *hằng* (constant), *biến* (variable) và các *hạng phức hợp* (compound term).

Hạng phức hợp là một *hàm tử* (functor) có chứa *các đối* (argument) có dạng

$$\text{Tên\_hàm\_tử}(\text{Đối\_1}, \dots, \text{Đối\_n})$$

Tên hàm tử là một chuỗi chữ cái và/hoặc chữ số được bắt đầu bởi một chữ cái thường. các đối có thể là biến, hạng sơ cấp, hoặc hạng phức hợp. Trong Prolog, hàm tử đặc biệt “.” Biểu diễn cấu trúc danh sách.

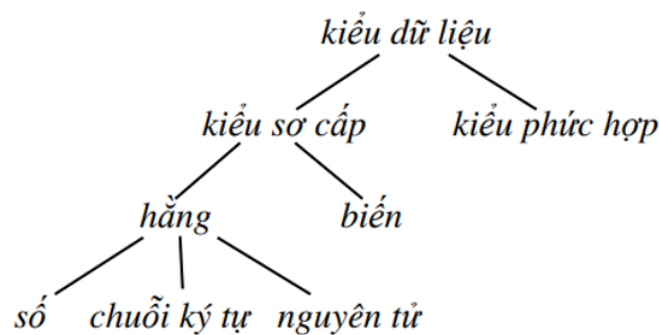
Mệnh đề có thể là một sự kiện, một luật (hay quy tắc), hay một câu hỏi.

Prolog quy ước viết sau mỗi mệnh đề một dấu chấm để kết thúc như sau:

- Sự kiện:  $\langle \dots \rangle$ . (tương ứng với luật  $\langle \dots \rangle \text{ :- true.}$ )
- Luật:  $\langle \dots \rangle \text{ :- } \langle \dots \rangle$ .
- Câu hỏi:  $?- \langle \dots \rangle$ .

1. **Link tải Prolog:** <https://www.swi-prolog.org/download/stable>

2. **Các kiểu dữ liệu của Prolog:**



Các kiểu dữ liệu của Prolog được xây dựng từ các ký tự ASCII:

- Các chữ cái in hoa  $A, B, \dots, Z$  và các chữ cái thường  $a, b, \dots, z$ .
- Các chữ số:  $0, 1, \dots, 9$ .
- Các ký tự đặc biệt:  $+ - * / < > = : . \& \_ \sim$

3. **Các kiểu dữ liệu sơ cấp:**

a. **Hằng số:** Prolog sử dụng cả số nguyên và số thực.

- Kiểu logic: true và false.
- Hằng chuỗi ký tự: các hằng là chuỗi các ký tự được đặt giữa hai dấu “ ”.
- Kiểu hằng nguyên tử: là chuỗi ký tự ở 1 trong 3 dạng như sau:
  - Chuỗi gồm chữ cái, chữ số và ký tự  $\_$  luôn được bắt đầu bằng một chữ cái in thường. Ví dụ:  $a\_$ ,  $x\_y$ ,  $x$
  - Chuỗi các ký tự đặc biệt:  $\langle - \rangle$ ,  $\langle ===== \rangle$ ,  $\langle \dots \rangle$ ,  $\langle ::= \rangle$

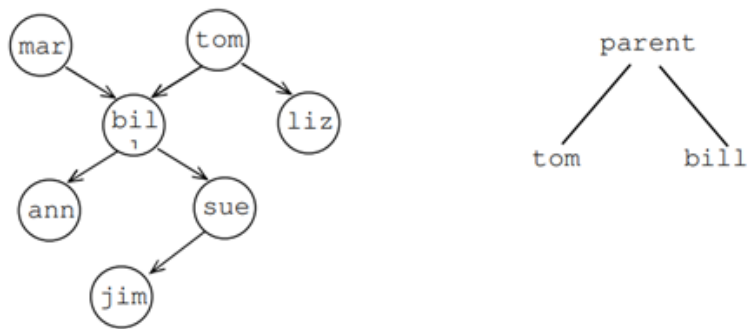
- Chuỗi đặt giữa 2 dấu nháy đơn được bắt đầu bằng chữ in hoa, dùng để phân biệt với các tên biến. Ví dụ: ‘Jerry’.

**b. Biến:** là một chuỗi ký tự gồm các chữ cái, chữ số và bắt đầu bởi chữ hoa hoặc dấu `_`. Ví dụ: `X, Y, Ketqua, _X, _x1, ...`

**c. Xây dựng 1 sự kiện:**

Ví dụ: Xây dựng cây gia hệ sau

Trong đó, các nút chỉ người còn các mũi tên chỉ quan hệ cha mẹ. Ví dụ: Tom là



cha mẹ của Bill và được viết thành 1 vị từ như sau:

*parent(tom, bill).*

Từ cây gia hệ trên, ta có 6 vị từ sau:

*parent(marry, bill).*

*parent(tom, bill).*

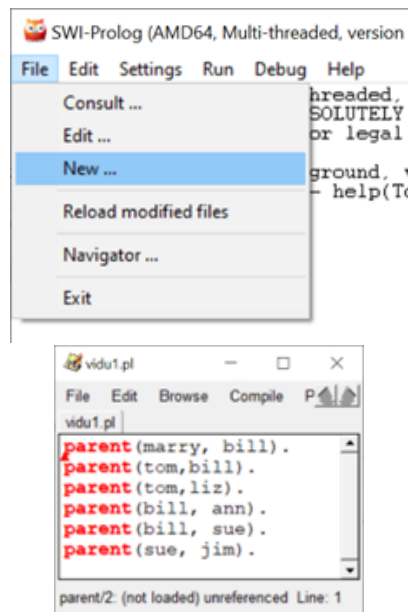
*parent(tom, liz).*

*parent(bill, ann).*

*parent(bill, sue).*

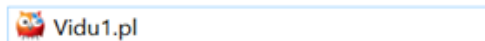
*parent(sue, jim).*

- Tạo cơ sở dữ liệu của 6 vị từ trên trong Prolog:

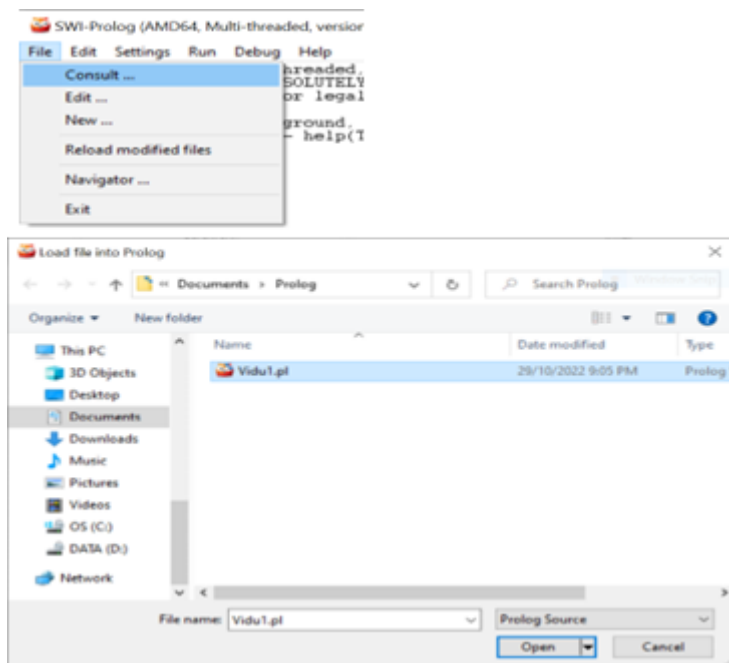


- Để chạy chương trình prolog:

\* Double click lên tập tin **Vidu1.pl**



\* Hoặc chạy phần mềm SWI-Prolog, và tham vấn (consult) tới tập tin chương trình.

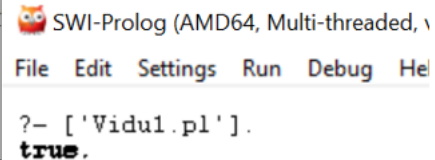


\* Hoặc sử dụng



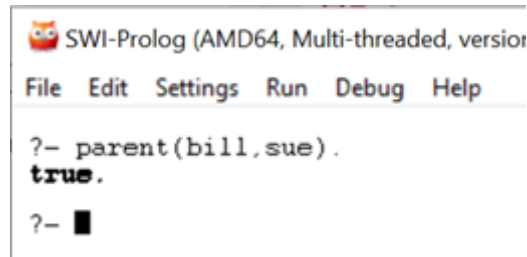
```
File Edit Settings Run Debug Help
?- consult('Vidu1.pl').
true.
```

hoặc



```
File Edit Settings Run Debug Help
?- ['Vidu1.pl'].
true.
```

- Sau đó, đưa ra câu hỏi mong muốn. Ví dụ: câu hỏi Bill có phải là cha mẹ của Sue được gõ vào trong hệ thống đối thoại Prolog.



```
File Edit Settings Run Debug Help
?- parent(bill,sue).
true.
?-
```

Ta tiếp tục đặt câu hỏi khác:

```
?- parent(liz,sue).
false.
```

Prolog không tìm thấy sự kiện Liz là cha mẹ của Sue. Prolog không trả lời true hay false mà đưa ra một giá trị của X.

```
?- parent(bill,X).
X = ann ,
```

Để biết ai là con của Bill, ta chỉ cần viết:

```
?- parent(X,liz).
X = tom.
```

Tổng quát hơn, tìm X và Y sao cho X là cha mẹ của Y. để biết được câu trả lời tiếp theo, ta sử dụng dấu chấm phẩy (;) và sử dụng dấu chấm (.) hoặc Enter để chấm dứt giữa chừng luồng trả lời.

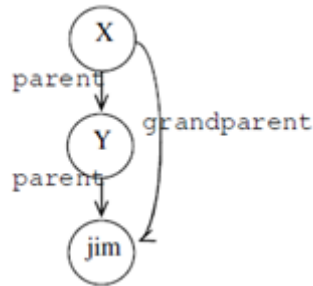
```

?- parent(X,Y).
X = marry,
Y = bill ;
X = tom,
Y = bill ;
X = tom,
Y = liz ;
X = bill,
Y = ann ;
X = bill,
Y = sue .

```

Ta tiếp tục đưa ra những câu hỏi phức tạp hơn, chẳng hạn ai là ông (bà) của Jim? Quan hệ ông bà (grandparent) cần phải phân tách câu hỏi này thành 2 phần sơ cấp hơn:

- Ai là cha mẹ của Jim? Giả sử có tên là Y.
- Ai là cha mẹ của Y? Giả sử có tên là X.



Lúc này ta viết trong Prolog như sau:

```

?- parent(Y,jim), parent(X,Y).
Y = sue,
X = bill.

```

Câu hỏi này tương ứng với: tìm  $X$  và  $Y$  thỏa mãn

$parent(Y,jim)$  và  $parent(X,Y)$ .

Nếu thay đổi thứ tự 2 thành phần của câu hỏi thì nghĩa logic vẫn không thay đổi

```

?- parent(X,Y), parent(Y,jim).
X = bill,
Y = sue .

```

Ta đặt câu hỏi *ai là cháu của Tom?*

```
?- parent(tom,X),parent(X,Y).
X = bill,
Y = ann ;
X = bill,
Y = sue ;
false.
```

Một câu hỏi khác: Ann và Sue có cùng cha mẹ không? Ta diễn đạt thành 2 giai đoạn:

- Tìm  $x$  là cha mẹ của Ann.
- $X$  tìm thấy được có là cha mẹ của Sue không?

```
?- parent(X,ann), parent(X,sue).
X = bill.
```

#### d. Xây dựng luật:

Từ chương trình trên, ta bổ sung thêm các sự kiện về giới tính (nam, nữ) của những người đã nêu trên trong quan hệ parent như sau:

*woman(marry).*

*man(tom).*

*man(bill).*

*woman(liz).*

*woman(sue).*

*woman(ann).*

*man(jim).*

Ta định nghĩa các quan hệ *đơn* (unary) *woman* và *man* vì chúng chỉ liên quan đến 1 đối tượng duy nhất, còn quan hệ *parent* là nhị phân vì chúng liên quan tới 1 cặp đối tượng. Như vậy, các quan hệ đơn dùng để thiết lập một thuộc tính của một đối tượng. Tuy nhiên, ta cũng có thể sử dụng quan hệ nhị phân để định nghĩa giới tính:

*sex(mary, female).*

*sex(tom, male).*

*sex(bill, male).*

...

Ta đưa ra quan hệ mới *child*, đối ngược với *parent* như sau:

$$child(liz, tom).$$

Từ đó, ta định nghĩa luật mới như sau:

$$child(Y, X) : \neg parent(X, Y).$$

Luật trên được hiểu là:

Với mọi  $X$  và  $Y$ ,

$Y$  là con của  $X$  nếu

$X$  là cha mẹ của  $Y$ .

hay

Với mọi  $X$  và  $Y$ ,

Nếu  $X$  là cha mẹ của  $Y$  thì

$Y$  là con của  $X$ .

Mỗi luật bao gồm 2 phần:

- Phần bên phải chỉ *điều kiện*, còn được gọi là *thân* (body) của luật.
- Phần bên trái chỉ *kết luận*, còn được gọi là *đầu* (head) của luật. Nếu điều kiện  $parent(X, Y)$  đúng thì  $child(Y, X)$  cũng đúng và là hệ quả logic của phép suy luận (inference). Câu hỏi sau đây giải thích cách Prolog sử dụng các luật: Liz có phải là con của Tom không?

```
| ?- child(liz, tom).  
| true .
```

Ta có thể sử dụng phép thế (substitution) bằng cách gán giá trị *liz* cho biến  $Y$  và *tom* cho  $X$  và thực hiện các truy vấn sau:

Ta tiếp tục bổ sung các quan hệ mới. Quan hệ *mother* được định nghĩa như sau (dấu phẩy chỉ phép hội hay phép và logic):

$$mother(X, Y) : \neg parent(X, Y), woman(X).$$



được hiểu là:

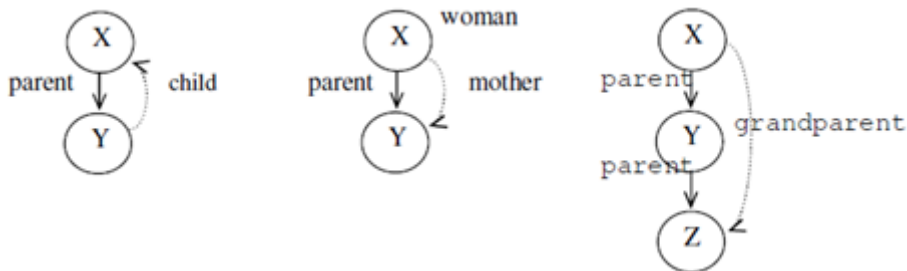
```
?- child(Y, X).  
Y = liz,  
X = tom .  
  
?- child(X, marry).  
X = bill.  
  
?- child(X, ann).  
false.
```

Với mọi  $X$  và  $Y$ ,

$X$  là mẹ của  $Y$  nếu

$X$  là cha mẹ của  $Y$  và  $X$  là nữ.

Đồ thị minh họa định nghĩa quan hệ *child*, *mother* và *grandparent*. Trong đó, các nút tương ứng với các đối tượng, các cung nối các nút tương ứng với các quan hệ nhị phân được định hướng từ đối tượng thứ nhất đến đối tượng thứ hai của quan hệ. Một quan hệ đơn được biểu diễn bởi tên quan hệ tương ứng với nhãn của đối tượng đó. Các quan hệ cần định nghĩa sẽ được biểu diễn bởi các cung có nét đứt. mỗi đồ thị được giải thích như sau: nếu các quan hệ được chỉ bởi các cung có nét liền được thỏa mãn, thì quan hệ biểu diễn bởi cung có nét đứt cũng được thỏa mãn.



như vậy, quan hệ *grandparent* được viết như sau:

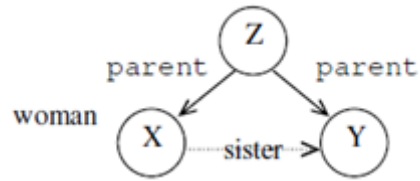
$$\text{grandparent}(X, Z) : \neg \text{parent}(X, Y), \text{parent}(Y, Z).$$

ta tiếp tục định nghĩa quan hệ chị em gái *sister* như sau:

với mọi  $X$  và  $Y$ ,  $X$  là một chị (em) gái của  $Y$  nếu

- (1)  $X$  và  $Y$  có cùng cha mẹ, và
- (2)  $X$  là nữ.

$sister(X, Y) : \neg parent(Z, X), parent(Z, Y), woman(X).$



Ann là nữ, Ann và Sue có cùng cha mẹ nên Ann là chị em gái của Sue:

```

?- sister(ann, sue).
true.

?- sister(X, sue).
X = ann ;
X = sue .
  
```

$\Rightarrow$  Xây dựng quan hệ different để được câu trả lời *Ann là chị em gái của Sue* ??

### III. NỘI DUNG THỰC HÀNH:

1. Cho quan hệ parent như trong phần II, cho biết kết quả của các câu hỏi sau:
  - a. ?- parent(jim, X).
  - b. ?- parent(X, jim).
  - c. ?- parent(marry, X), parent(X, part).
  - d. ?- parent(marry, X), parent(X, Y), parent(Y, jim).
2. Viết các mệnh đề Prolog diễn tả các câu hỏi liên quan đến quan hệ parent:
  - a. Ai là cha mẹ của Bill?
  - b. Marry có con không?
  - c. Ai là ông bà (grandparent) của Sue?