Tổng hợp bài tập lập trình Prolog

* **Tìm phần tử thứ nhất**

|  |  |
| --- | --- |
| 1 | timdautien([Dau|Duoi],Dau). |

Ví dụ : Gọi vị từ  timdautien([12,4,2,3,4,5],Kq) thì kết quả trả về là Kq=12.

**Diễn đạt bằng lời** :

*Truyền vào danh sách lấy phần tử đầu gán cho kết quả*

* **Tìm phần tử thứ hai**

|  |  |
| --- | --- |
| 1 | timhai([\_,Y|\_],Y). |

Ví dụ : Gọi vị từ  timhai([4,3,4,5],Kq) thì kết quả trả về là Kq=3.

**Diễn đạt bằng lời** :

*Truyền vào danh sách không quan tâm thằng thứ nhất là ai cả (để nó là vô danh) lấy phần tử thứ 2 gán cho kết quả.*

* **Tìm phần tử thứ cuối**

|  |  |
| --- | --- |
| 1  2 | timcuoi([X],X).  timcuoi([\_|Duoi],Kq):-timcuoi(Duoi,Kq). |

Ví dụ : Gọi vị từ  timcuoi([4,3,4,5],Kq) thì kết quả trả về là Kq=5.

**Diễn đạt bằng lời** :

*Truyền vào danh sách không quan tâm thằng thứ nhất là ai cả (để nó là vô danh) ,chưa đến cuối cùng của danh sách thì cứ tiếp tục tìm. (Dòng code 2).*

*Nếu X là thằng cuối cùng thì gán Kq là X (Dòng code 1).*

* **Lấy ra phần tử đầu tiên trùng với X**

|  |  |
| --- | --- |
| 1  2  3 | layphantudau([],\_,[]).  layphantudau([X|Duoi],X,Duoi):-!.  layphantudau([Dau|Duoi],X,[Dau|Kq]):-layphantudau(Duoi,X,Kq). |

Ví dụ : Gọi vị từ  layphantudautien([1,1,2,3,4,5],1,Kq) thì kết quả trả về là Kq=[1,2,3,4,5].

**Diễn đạt bằng lời** :

*Nếu trong danh sách phần tử đầu không trùng với X thì đi tìm ở danh sách đuôi và kết quả phải có đầu (nhớ phần tử đầu cho kết quả trả về) (Dòng code 3)*

*Nếu gặp phần tử nào đó trùng thì trong kết quả không có X (không nhớ X) dừng và đệ quy ngược lại lấy các phần tử trước đó đã nhớ ghép vào danh sách .(Dòng code 2)*

*Nếu đi hết danh sách mà không có phần tử nào trùng với X thì sao nhỉ thì danh sách kết quả sẽ bằng rỗng và đệ quy ngược lại lấy các phần tử trước đó đã nhớ để trả về danh sách ban đầu (không xóa được phần tử nào ).(Dòng code 1)*

* **Lấy ra tất cả các phần tử trùng với X**

Bài này cũng giống như bài ở trên nhưng có biến đổi 1 tí

|  |  |
| --- | --- |
| 1  2  3 | layratatcaphantu([],\_,[]).  layratatcaphantu([X|Duoi],X,Kq):-layratatcaphantu(Duoi,X,Kq),!.  layratatcaphantu([Dau|Duoi],X,[Dau|Kq]):-layratatcaphantu(Duoi,X,Kq). |

Ví dụ : Gọi vị từ layratatcaphantu([1,1,2,3,1,4,5],1,Kq) thì kết quả trả về là Kq=[2,3,4,5].

**Diễn đạt bằng lời** :

*Nếu gặp phần tử đầu trùng với X thì trong kết quả sẽ không có X tiếp tục đi tìm lời giải ở phần đuôi . Nếu phần tử đầu không trùng với X thì không thỏa điều kiện gặp nhát cắt dừng việc tìm kiếm lại (Dòng code 2), đi xuống dòng code 3 đưa phần tử đầu mà không trùng với X đó vào kết quả và tiếp tục đi tìm lời giải ở đuôi.(Dòng code 3)*

Nếu danh sách đuôi tìm kiếm đã hết thì kết quả bằng rỗng và đệ quy ngược trở lại ghép những phần tử đã được đưa vào kết quả trước đó.

* **Xuất ra các cặp phần tử kế tiếp nhau trong danh sách**

|  |  |
| --- | --- |
| 1  2 | xuatcacphantuketiepnhau([]).  xuatcacphantuketiepnhau([\_]):-!.  xuatcacphantuketiepnhau([X,Y|Duoi]):-write(X), write(‘—‘), write(Y), nl,  xuatcacphantuketiepnhau([Y|Duoi]). |
|  |  |

Ví dụ : Gọi vị từ xuatcacphantuketiepnhau([1,2,3,4,5]) thì kết quả trả về là :

1–2

2–3

3–4

4–5

**Diễn đạt bằng lời** :

*Nếu có phần tử X và Y bất kỳ ở đầu thì thực hiện việc xuất X và Y sau đó tiếp tục đi về phần đuôi để thực hiện việc xuất nhưng ở phần đuôi không được bỏ qua Y ,xuất X và Y cho đến khi nào danh sách rỗng.*

* **Tạo danh sách tất cả các phần tử ở vị trí chẵn của danh sách dữ liệu L**

♦ Giữ nguyên trật tự xuất hiện trong L

|  |  |
| --- | --- |
| 1  2  3 | phantuovitrichan([],[]).  phantuovitrichan([\_],[]).  phantuovitrichan([\_,Chan|Duoi],[Chan|KQ]):-phantuovitrichan(Duoi,KQ). |

Ví dụ : gọi vị từ phantuovitrichan([1,2,3,4,5,6,7],Kq) thì kết Kq=[2,4,6]

**Diễn đạt bằng lời :**

*Phần tử đầu là phần tử ở vị trí lẻ không quan tâm nó ,quan tâm phần tử sau nó là phần tử chẵn cho vào kết quả cần tìm ,nếu danh sách ban đầu vẫn còn phần tử thì đi về đuôi để tiếp tục lấy phần tử chẵn.(Dòng 3)*

*Nếu danh sách là rỗng (số phần tử của danh sách là chẵn) hoặc danh sách chỉ còn đúng 1 phần tử (số phần tử của danh sách là lẻ) thì kết quả bằng rỗng dừng việc đệ quy và quay ngược trở lại để lấy các phần tử chẵn trước đó cho kết quả .(Dòng 1 ,2).*

♦ Đão ngược trật tự

|  |  |
| --- | --- |
| 1  2  3 | phantuovitrichandao([],Phu,Phu).  phantuovitrichandao([\_],Phu,Phu).  phantuovitrichandao([\_,Chan|Duoi],Phu,Kq):-PhuMoi=[Chan|Phu],  phantuovitrichandao(Duoi,PhuMoi,Kq). |

Ví dụ : gọi vị từ phantuovitrichandao([1,2,3,4,5,6,7],[],Kq) thì kết Kq=[6,4,2]

**Diễn đạt bằng lời :**

*Ban đầu ta truyền vào Phụ bằng rỗng ,Lấy phần tử ở vị trí chẵn đưa vào đầu của Phụ ta có được Phụ mới sau khi có được phụ mới ta tiếp tục tìm ở phần Đuôi của danh sách với Phụ là Phụ mới (Dòng 3).*

*Tìm các phần tử ở vị trí chẵn cho đến khi danh sách chỉ còn lại một phần tử (Danh sách với số phần tử lẻ ) hoặc đã duyệt qua tất cả các phần tử danh sách duyệt còn lại rỗng (Danh sách với số phần tử chẵn), dừng lại và gán Phụ đã tìm được từ các bước trước cho kết quả Kq=Phu.(Dòng 1 , 2).*

* **Xây dựng danh sách mà các phần tử là căn bậc hai của các phần tử tương ứng trong danh sách L các số thực cho trước**

|  |  |
| --- | --- |
| 1  2 | danhsachcanbachai([],[]).  danhsachcanbachai([Dau|Duoi],[SoCanBacHai|Kq]):-SoCanBacHai is sqrt(Dau),  danhsachcanbachai(Duoi,Kq). |

Ví dụ : gọi vị từ danhsachcanbachai([2,4],Kq) thì kết Kq=[1.4142135624,2].

**Diễn đạt bằng lời :**

*Duyệt qua danh sách ban đầu tính căn bậc hai của từng giá trị rồi cho vào kết quả .*

* **Trộn danh sách xen kẽ**

|  |  |
| --- | --- |
| 1  2  3 | tronxenke([],[X],[X]).  tronxenke([X],[],[X]).  tronxenke([], [], []).  tronxenke([Dau1|Duoi1],DS2,[Dau1|Kq]):-tronxenke(DS2,Duoi1,Kq). |

Ví dụ : gọi vị từ tronxenke([1,2,3],[4,5,6],Kq) thì kết Kq=[1,4,2,5,3,6].

**Diễn đạt bằng lời :**

*Duyệt danh sách thứ 1 và thứ 2 xen kẻ nhau cho đến khi danh sách thứ nhất chỉ còn 1 phần tử và danh sách thứ 2 rỗng hoặc danh sách thứ 1 rỗng và danh sách thứ 2 chỉ còn một phần tử thì kết quả chứa phần tử cuối cùng đó , quay trở ngược lại gom các phần tử đã trộn xen kẻ trước đó cho kết quả hoàn chỉnh .*

* **Ai nói thật ai nói láo.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | sinh2(0).  sinh2(1).  sinhDS([],0):-!.  sinhDS([Dau|Duoi],N):-sinh2(Dau),N1=N-1,sinhDS(Duoi,N1).    tong([],0).  tong([Dau|Duoi],Kq):-tong(Duoi,KQDuoi),Kq=KqDuoi+Dau.    anoi([1|Duoi]):-tong([1|Duoi],SoThat),SoThat=1.  anoi([0|Duoi]):-tong([0|Duoi],SoThat),SoThat1.    cnoi([A,B,1]):-tong([A,B,1],SoThat),SoThat=0.  cnoi([A,B,0]):-tong([A,B,0],SoThat),SoThat>0.    Suyluan(L,N):- sinhDS(L,N),anoi(L),cnoi(L). |

* **Pythagore điều kiện :  0<x<y<z , x^2+y^2=z^2,UCLN(x,y,1) : Ước chung lớn nhất của x và y bằng 1.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | ucln(X,X,X):-!.  ucln(X,Y,KQ):-X>0,Y>0,X>Y,X1=X-Y,!,ucln(X1,Y,KQ).  ucln(X,Y,KQ):-X>0,Y>0,Y1=Y-X,ucln(X,Y1,KQ).    sinhso(I,I,\_).  sinhso(I,KQ,ChanTren):-I1=I+1,I1<ChanTren,sinhso(I1,KQ,ChanTren).    pytago(X,Y,Z):-sinhso(1,Z,100),sinhso(1,X,Z),  sinhso(1,Y,Z),X<Y,ucln(X,Y,1),X\*X+Y\*Y=Z\*Z. |

* **Selection Sort**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | min([X],X).  min([X,Y|T],KQ):-X>Y,!,min([Y|T],KQ).  min([X,\_|T],KQ):-min([X|T],KQ).    delete(\_,[],[]).  delete(X,[X|T],KQ):-!,delete(X,T,KQ).  delete(X,[H|T],[H|KQ]):-delete(X,T,KQ).    selection\_sort([],[]).  selection\_sort(L,[Min|KQ]):-min(L,Min),  delete(Min,L,L1),  selection\_sort(L1,KQ). |

Run code : [Here](http://swish.swi-prolog.org/p/Selection%20Sort%20Prolog.pl)

* **Insertion Sort**

|  |  |
| --- | --- |
| 1  2  3  4  5 | insert(X,[H|T],[H|KQ]):-X>=H,!,insert(X,T,KQ).  insert(X,T,[X|T]).    insertion\_sort([],[]).  insertion\_sort([H|T],KQ):-insertion\_sort(T,KQ1),insert(H,KQ1,KQ). |

* **Quick Sort**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | noi([],P,P).  noi([D|C],P,Result):-noi(C,[D|P],Result).    ph(\_,[],[],[]).  ph(Moc,[D|C],[D|LBe],LLon):-D<=Moc,!,  ph(Moc,C,LBe,LLon).  ph(Moc,[D|C],LBe,[D|LLon]):-ph(Moc,C,Lbe,LLon).    qs([],[]).  qs([D|C],Result):-ph(D,C,LBe,LLon),  qs(LBe,Be),qs(LLon,Lon),noi(Be,[D],Tam),  noi(Tam,Lon,Result). |

Tổng hợp bài tập lập trình Prolog (TT)

* **Kiểm tra số tự nhiên N có phải là số nguyên tố hay không ? Tìm N số nguyên tố đầu tiên.**

***Số nguyên tố****là*[*số tự nhiên*](https://vi.wikipedia.org/wiki/S%E1%BB%91_t%E1%BB%B1_nhi%C3%AAn)*chỉ có hai ước số dương phân biệt là 1 và chính nó.*

Viết vị từ kiểm tra một số tự nhiên có phải là số nguyên tố hay không

|  |  |
| --- | --- |
| 1  2 | kiemtranguyento(N,I):-N>1,N=I.  kiemtranguyento(N,I):-N>1,Du=N mod I , Du <> 0,I1=I+1,kiemtranguyento(N,I1). |

Ví dụ gọi vị từ kiemtranguyento(5,2) thì kết quả trả về là **Yes**.

**Diễn đạt bằng lời** :

***N****là số ta muốn kiểm tra có phải nguyên tố hay không.*

***I****là giá trị chạy từ I đến  N , nhiệm vụ để kiểm tra xem trong khoảng từ I cho đến N thì N có chia hết cho số nào trong khoảng đó hay không ,nếu chia hết mà lúc này I=N tức là trong khoảng từ I đến N-1 không có bất kỳ số nào mà N chia hết thì đó là số nguyên tố ,còn ngược lại trong khoảng đó N có chia hết cho số nào đó thì N không phải nguyên tố.*

Viết vị từ songuyentonhohonnsapgiam để liệt kê các số nguyên tố mà nhỏ hơn hoặc bằng N sắp theo chiều giảm dần.

|  |  |
| --- | --- |
| 1  2  3 | songuyentonhohonnsapgiam(2,[2]):-!.  songuyentonhohonnsapgiam(N,[N|Kq]):-kiemtranguyento(N,2),!,N1=N-1,  songuyentonhohonnsapgiam(N1,Kq).  songuyentonhohonnsapgiam(N,Kq):-N1=N-1,songuyentonhohonnsapgiam(N1,Kq). |

Ví dụ : Gọi vị từ  songuyentonhohonnsapgiam(5,Kq) thì kết quả trả về là Kq=[5,3,2].

**Diễn đạt bằng lời** :

*Ban đầu kiểm tra xem N truyền vào có bằng 2 hay không nếu bằng 2 thì trả về kết quả là 2 (Dòng 1)*

*Nếu đi kiểm tra N là số nguyên tố thì trong kết quả sẽ có N và đi tìm tiếp N-1 có phải là số nguyên tố hay không (Dòng 2)*

*Nếu N không phải là số nguyên tố thì trong kết quả sẽ không có N và đi tìm tiếp N-1 có phải là nguyên tố hay không (Dòng 3)*

*Đi tìm cho đến khi nào N=2 thì dừng và trả về kết quả mảng các số nguyên tố.*

Viết vị từ songuyentonhohonnsaptang để liệt kê các số nguyên tố mà nhỏ hơn hoặc bằng N sắp theo chiều tăng dần.

|  |  |
| --- | --- |
| 1  2  3 | songuyentonhohonnsaptang(X,N,[]):-X>N,!.  songuyentonhohonnsaptang(X,N,[X|Kq]):-kiemtranguyento(X,2),!,X1=X+1,  songuyentonhohonnsaptang(X1,N,Kq).  songuyentonhohonnsaptang(X,N,Kq):-X1=X+1,songuyentonhohonnsaptang(X1,N,Kq). |

Ví dụ : Gọi vị từ songuyentonhohonnsaptang (2,5,Kq) thì kết quả trả về là Kq=[2,3,5].

**Diễn đạt bằng lời** :

***X****là số tăng từ 2 cho đến N  . X<=N*

*Kiểm tra xem X có phải là nguyên tố hay không nếu X là nguyên tố thì trong kết quả có X và đi kiểm tra X+1 có phải là nguyên tố hay không.(Dòng 2)*

*Nếu X không phải là nguyên tố thì vẫn đi tìm X+1 có phải là nguyên tố hay không (Dòng 3)*

*Tìm cho đến khi X>N thì dừng lại và trả về kết q*uả .(Dòng 1).

***Còn có cách  khác như sau :***

|  |  |
| --- | --- |
| 1  2  3 | songuyentonhohonnsaptang2(1,Phu,Phu):-!.  songuyentonhohonnsaptang2(X,Phu,Kq):-kiemtranguyento(X,2),!,X1=X-1,  PhuMoi=[X|Phu],  songuyentonhohonnsaptang2(X1,PhuMoi,Kq).  songuyentonhohonnsaptang2(X,Phu,Kq):-X1=X-1,  songuyentonhohonnsaptang2(X1,Phu,Kq). |

* **Đão ngược trật tự mọi phần tử của một danh sách**

|  |  |
| --- | --- |
| 1  2  3  4 | daonguocdanhsach(DS,Kq):-daonguocdanhsachx(DS,[],Kq).    daonguocdanhsachx([],Phu,Phu).  daonguocdanhsachx([Dau|Duoi],Phu,Kq):-PhuMoi=[Dau|Phu],  daonguocdanhsachx(Duoi,PhuMoi,Kq). |

Ví dụ : Gọi vị từ  daonguocdanhsach([1,2,3,4,5],Kq) thì kết quả trả về là Kq=[5,4,3,2,1].

Hoặc có thể gọi vị từ daonguocdanhsachx([1,2,3,4,5],[],Kq) thì kết quả trả về là Kq=[5,4,3,2,1].

**Diễn đạt bằng lời** :

*Khi gọi vị từ daonguocdanhsachx thì Phụ ban đầu là rỗng sau đó gán phần tử đầu vào vị trí đầu tiên của Phụ được Phụ Mới ,duyệt như vậy cho đến khi danh sách rỗng thì Kq bằng danh sách Phụ đã tìm được trước đó .(Dòng 2, Dòng 3)*

*Khi gọi vị từ daonguocdanhsach thì nó sẽ gọi đến daonguocdanhsachx với Phu là rỗng rồi duyệt danh sách như giải thích bên trên.*

* **Xây dựng vị từ thuộc trong đó có thêm một đối số để trả vệ trị TRUE nếu phần tử x thuộc danh sách l cho trước và FALSE trong trường hợp ngược lại .**

|  |  |
| --- | --- |
| 1  2  3 | thuoc(\_,[],"FALSE").  thuoc(X,[X|\_],"TRUE"):-!.  thuoc(X,[\_|Duoi],Kq):-thuoc(X,Duoi,Kq). |

Ví du gọi vị từ thuoc(2,[1,2,3,4],Kq) thì Kq=TRUE

Hoặc thuoc(5,[1,2,3,4],Kq) thì kq=FALSE

**Diễn đạt bằng lời** :

*Nếu phần tử X xuât hiện tại vị trí nào đó trong danh sách thì Kq=”TRUE” còn ngược lại thì duyệt danh sách về Đuôi (Dòng 1 , Dòng 2) nếu duyệt đến khi danh sách đã rỗng thì Kq=”FALSE” (Dòng 1)*

* **Xây dựng quy tắc NằmNgoài(x,L) để kiểm tra xem x không thuộc danh sách L**

|  |  |
| --- | --- |
| 1  2  3 | namngoai(\_,[]).  namngoai(X,[X|\_]):-!,fail.  namngoai(X,[\_|Duoi]):-namngoai(X,Duoi). |

Ví du gọi vị từ namngoai(2,[1,2,3]) thì kết quả là **No**

Hoặc gọi vị từ namtrong(5,[1,2,3]) thì thì kết quả là ***Yes***

**Diễn đạt bằng lời** :

*Nếu duyệt danh sách phần tử đầu của danh sách không trùng với X thì duyệt về phần đuôi của danh sách cho đến khi danh sách rỗng thì trả về****Yes****(Dòng 1 ,Dòng 3)*

Nếu trong khi duyệt danh sách có phần tử trùng với X thì sử dụng nhát cắt để dừng việc tìm kiếm và đồng thời trả về fail ,kết quả là **No.**(Dòng 2).

* **Xây dựng quy tắc khacnhau(L) để kiểm tra danh sách L không có phần tử trùng nhau**

|  |  |
| --- | --- |
| 1  2 | khacnhau([]).  khacnhau([Dau|Duoi]):-namngoai(Dau,Duoi),!,khacnhau(Duoi). |

Ví du gọi vị từ khacnhau([1,2,2,3]) thì kết quả là **No**

khacnhau([1,2,3]) thì kết quả là **Yes**

**Diễn đạt bằng lời** :

*Nếu phần tử đầu  nằm ngoài danh sách đuôi (tức là không có phần tử nào thuộc danh sách đuôi ) thì tiếp tục đi về đuôi để kiểm tra .*

*Nếu như phần tử đầu trùng với phần tử nào đó trong đuôi gặp nhát cắt dừng chương trình và trả kết quả về sai (No) (Dòng 2)*

*Nếu duyệt hết danh sách mà tất cả đều thỏa thì trả về kết quả đúng (True) (Dòng 1)*

* **Xây dựng quy tắc phandau(l1,l2) để kiểm tra danh sách l1 là phần đầu của danh sách l2**

|  |  |
| --- | --- |
| 1  2 | phandau([],\_).  phandau([Dau1|Duoi1],[Dau2|Duoi2]):-Dau1=Dau2,phandau(Duoi1,Duoi2). |

Ví du gọi vị từ khacnhau([1,2,2],[1,2,2,3]) thì kết quả là **Yes**

khacnhau([1,2,4],[1,2,3,4]) thì kết quả là **No**

**Diễn đạt bằng lời** :

*Duyệt 2 danh sách nếu phần tử đầu của danh sách 1 trùng với phần tử đầu của danh sách 2 thì duyệt tiếp phần đuôi của danh sách 1 và đuôi của danh sách 2 ,ngược lại thì dừng và trả về kết quả sai.(Dòng 2)*

*Nếu duyệt hết danh sách 1 thì kết quả trả về đúng (Các phần tử trong danh sách 1 đều thỏa điều kiện ) (Dong 1).*

* **Xây dựng quy tắc namtronglientiep(l1,l2) để kiểm tra mọi phần tử của l1 đều nằm liên tiếp trong danh sách l2**

|  |  |
| --- | --- |
| 1  2  3 | namtronglientiep([],\_).  namtronglientiep([Dau1|Duoi1],[Dau2|Duoi2]):-Dau1=Dau2,!,  namtronglientiep(Duoi1,Duoi2).  namtronglientiep([Dau1|Duoi1],[\_|Duoi2]):-namtronglientiep([Dau1|Duoi1],Duoi2). |

Ví du gọi vị từ namtronglientiep([1,2,3],[5,6,7,1,2,3]) thì kết quả là **Yes**

namtronglientiep([1,2,4],[1,2,3,4]) thì kết quả là **No**

**Diễn đạt bằng lời** :

*Nếu phần tử đầu của danh sách 1 bằng phần tử đầu của danh sách 2 thì duyệt về phần đuôi của danh sách 1 và danh sách 2 ,ngược lại nhảu xuống Dòng 3 (Dòng 2)*

*Xét phần tử đầu của danh sách 1 có nằm trong phần đuôi của danh sách 2 hay không (Dòng 3)*

*Nếu đã duyệt hết danh sách thứ 1 mà mọi điều kiện thỏa thì trả về kết quả đúng ( Dòng 1).*

* **Xây dựng quy tăc phân hoạch(L,U,V,W) để phân hoạch danh sách L thành 3 danh sách U,V,W .**

|  |  |
| --- | --- |
| 1  2  3  4 | phanhoach([],[],[],[]).  phanhoach([Dau|Duoi],U,[Dau|V],W):-Dau=10,!,phanhoach(Duoi,U,V,W).  phanhoach([Dau|Duoi],[Dau|U],V,W):-Dau<=10,!,phanhoach(Duoi,U,V,W).  phanhoach([Dau|Duoi],U,V,[Dau|W]):-phanhoach(Duoi,U,V,W). |

Ví dụ : gọi vị từ phanhoach([1,2,3,4,5,6,7,8,9,10,11,12,13,10],U,V,W) thì kết quả là U=[1,2,3,4,5,6,7,8,9] , V=[10,10] , W=[11,12,13] .

**Diễn đạt bằng lời** :

*Nếu phần tử đầu của danh sách Dau =10 thì nó được cho vào danh sach V  và duyệt tiếp phần đuôi của danh sách L nếu điều kiện không thỏa thì sử dụng nhát cắt dừng việc tìm kiếm kết quả đi xuống dòng tiếp theo (Dòng 2)*

*Nếu phần tử đầu của danh sách Dau <=10 thì nó được cho vào danh sach U và duyệt tiếp phần đuôi của danh sách L nếu điều kiện không thỏa thì sử dụng nhát cắt dừng việc tìm kiếm kết quả đi xuống dòng tiếp theo (Dòng 3)*

*Nếu đã đi qua các điệu kiện trên Dau =10 và Dau <=10 mà không thỏa thì phần đầu lúc này sẽ Dau > 10 và cho vào danh sách V (Dòng 4).*

*Nếu duyệt hết danh sách thì các danh sách U , V ,W là rỗng sau đó quay lui trở lại để lấy kết quả của mỗi danh sách mà lúc đầu nếu thỏa điều kiện đã gán vào danh sách đo (Dòng 1)*

* **Xuất ra mọi hoán vị của những phần tử của danh sách cho trước**

|  |  |
| --- | --- |
| 1  2  3  4  5 | noi([],L2,L2).  noi([X|Duoi1],L2,[X|Kq]):-noi(Duoi1,L2,Kq).    hoanvi([],[]).  hoanvi([Dau|Duoi],Kq):-hoanvi(Duoi,KqDuoi),  noi(L1,L2,KqDuoi),  noi(L1,[Dau],Tam),  noi(Tam,L2,Kq). |

[Sàng Eratosthene](https://voxuanphong.wordpress.com/2017/05/03/sang-eratosthene/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | %trace  domains    i=integer  list=i\*    predicates    boboi(i,list,i,list)  taodanhsach(i,list)  sang(i,list,list)  eras(i,list)  clauses    boboi(VT,[D|Duoi],I,[D|DuoiKq]):-Du=VT mod I ,Du >0,!,  VT1=VT+1,boboi(VT1,Duoi,I,DuoiKq).    boboi(VT,[\_|Duoi],I,[0|DuoiKq]):-VT1=VT+1,boboi(VT1,Duoi,I,DuoiKq).    boboi(\_,[],\_,[]).    taodanhsach(2,[]):-!.  taodanhsach(N,[1|Duoi]):-N1=N-1,taodanhsach(N1,Duoi).    sang(\_,[],[]).  sang(I,[Dau|DuoiSang],[I|DuoiKq]):-Dau=1,!,VT=I+1,  boboi(VT,DuoiSang,I,DuoiSangMoi),  sang(VT,DuoiSangMoi,DuoiKq).      sang(I,[\_|DuoiSang],DuoiKq):-VT=I+1,sang(VT,DuoiSang,DuoiKq).    eras(N,Kq):-taodanhsach(N,Sang),sang(2,Sang,Kq). |

[Ngựa vằn](https://voxuanphong.wordpress.com/2017/05/03/ngua-van/)

* Houses logical puzzle: who owns the zebra and who drinks water?

1) Five colored houses in a row, each with an owner, a pet, cigarettes, and a drink.  
2) The English lives in the red house.  
3) The Spanish has a dog.  
4) They drink coffee in the green house.  
5) The Ukrainian drinks tea.  
6) The green house is next to the white house.  
7) The Winston smoker has a serpent.  
8) In the yellow house they smoke Kool.  
9) In the middle house they drink milk.  
10) The Norwegian lives in the first house from the left.  
11) The Chesterfield smoker lives near the man with the fox.  
12) In the house near the house with the horse they smoke Kool.  
13) The Lucky Strike smoker drinks juice.  
14) The Japanese smokes Kent.  
15) The Norwegian lives near the blue house.

Who owns the zebra and who drinks water?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51 | %trace  domains    s=symbol  i=integer  listi=i\*  lists=s\*    predicates    cungcot(s,lists,s,lists)  sinhds(i,lists)  xuat(lists)  phai(s,lists,s,lists)  canh(s,lists,s,lists)    nha    clauses    cungcot(X,[X|\_],Y,[Y|\_]).  cungcot(X,[\_|DuoiX],Y,[\_|DuoiY]):-cungcot(X,DuoiX,Y,DuoiY).    sinhds(0,[]):-!.  sinhds(N,[\_|Duoi]):-N1=N-1,sinhds(N1,Duoi).    xuat([]).  xuat([D|Duoi]):-free(D),!,writef("%15s","\_"),xuat(Duoi).  xuat([D|Duoi]):-writef("%15s",D),xuat(Duoi).    phai(X,[X|\_],Y,[\_,Y|\_]):-!.  phai(X,[\_|DuoiX],Y,[\_|DuoiY]):-phai(X,DuoiX,Y,DuoiY).    canh(X,LX,Y,LY):-phai(X,LX,Y,LY);phai(Y,LY,X,LX).    nha:-Nha=["1","2","3","4","5"],  sinhds(5,QT),sinhds(5,Mau),sinhds(5,Vat),  sinhds(5,Uong),sinhds(5,Hut),cungcot(anh,QT,do,Mau),  cungcot(oldgold,Hut,sen,Vat),  cungcot(tbnha,QT,cho,Vat),  cungcot(cafe,Uong,xlcay,Mau),  cungcot(ukrain,QT,tra,Uong),  cungcot(kool,Hut,vang,Mau),  cungcot(gitanes,Hut,ruou,Uong),  cungcot(nhat,QT,caraven,Hut),  cungcot(anh,QT,do,Mau),  cungcot(tbnha,QT,cho,Vat),  cungcot(sua,Uong,"3",Nha),  cungcot(nauy,QT,"1",Nha),  phai(trang,Mau,xlcay,Mau),  canh(fied,Hut,cao,Vat),  canh(kool,Hut,ngua,Vat),  canh(nauy,QT,xdtroi,Mau),nl,  cungcot(\_,QT,"NUOC",Uong),  cungcot(\_,QT,"NGUAVAN",Vat),  xuat(QT),nl,xuat(Mau),nl,  xuat(Vat),nl,xuat(Uong),nl,xuat(Hut),nl.    goal  nha,nl,readchar(\_). |

[Bài Tập Các Phép Toán Trên Tập Hợp ProLog](https://voxuanphong.wordpress.com/2017/05/02/bai-tap-cac-phep-toan-tren-tap-hop-prolog/)

* **Xây dựng các phép toán trên tập hợp Hội ,Giao,Hiệu Thuộc,Bao Hàm.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52 | domains    i=integer  list=i\*    predicates    nhap(list)  xuat(list)  giao(list,list,list)  thuoc(i,list)  hop(list,list,list)  noi(i,list,list)  loaitrung(list,list)  hieu(list,list,list)  baoham(list,list)  clauses    nhap([X|T]):-readint(X),X>=0,!,nhap(T).  nhap([]).    xuat([]).  xuat([X|T]):-write(X),write(" "),xuat(T).    thuoc(X,[X|\_]).  thuoc(X,[\_|T]):-thuoc(X,T).    giao([],\_,[]).  giao([H1|T1],L2,[H1|Result]):-thuoc(H1,L2),!,giao(T1,L2,Result).  giao([\_|T1],L2,Result):-giao(T1,L2,Result).    noi(X,[],[X]).  noi(X,[H|T],[H|Result]):-noi(X,T,Result).    hop([],L2,L2).  hop([H1|T1],L2,Result):-noi(H1,L2,L2KQ),hop(T1,L2KQ,Result).    loaitrung([],[]).  loaitrung([H|T],Result):-thuoc(H,T),!,loaitrung(T,Result).  loaitrung([H|T],[H|Result]):-loaitrung(T,Result).    hieu([],\_,[]).  hieu([H|T],L2,[H|Result]):-not(thuoc(H,L2)),!,hieu(T,L2,Result).  hieu([\_|T],L2,Result):-hieu(T,L2,Result).    baoham(\_,[]).  baoham(L,[H|T]):-thuoc(H,L),!,baoham(L,T).  goal    clearwindow,  write("Nhap vao Tap A : "),nl,nhap(A),loaitrung(A,A1),nl,  write("Xuat Tap A :"),nl,xuat(A1),nl,    write("Nhap vao Tap B : "),nl,nhap(B),loaitrung(B,B1),nl,  write("Xuat Tap B :"),nl,xuat(B),nl,    write("A giao B :"),nl,giao(A1,B1,G),xuat(G),nl,  write("A hop B :"),nl,hop(A1,B1,H),loaitrung(H,H1),xuat(H1),nl,    write("A hieu B :"),nl,hieu(A1,B1,N),nl,xuat(N). |

**Diễn đạt bằng lời :**

*Với vị từ nhap(L) ta tiến hành nhập phần tử và gán vào danh sách ,nếu phần tử nhập vào X>=0 thì vẫn gọi vị từ nhập để nhập danh sách còn ngược lại thì gặp nhát cắt và dừng việc nhập lại đi xuống dòng tiếp theo và trả về danh sách rỗng sau đó đệ quy ngược lại để gom các phần tử trước đó đã cho vào danh sách .*

*Với vị từ xuat(L) Duyệt danh sách và xuất phần tử cho đến khi nào danh sách rỗng thì dừng.*

*Với vị từ thuoc(X,L) Nếu phần tử đầu của danh sách trùng với X thì trả về kết quả đúng còn ngược lại khi duyệt hết danh sách mà không có phần tử X nào trong danh sách L thì trả về kết quả sai.*

*Vị từ noi(X,L,R) duyệt qua danh sách L gán các phần tử của L cho R cho đến khi danh sách L là rỗng thì gán X vào danh sách R.*

*Vị từ hop(L1,L2,R) nối từng phần tử của danh sách L1 vào L2 được danh sách R*