**PROLOG**

**List (nho dau cham “.”)**

indanhsach(L,H,T) :- L = [H|T].

**//Dem so phan tu cua 1 danh sach**

dem([], 0) :- !.

dem([\_|T], X) :- dem(T,X1), X is X1 + 1.

**//Tinh tong cac phan tu trong danh sach**

tong([],0) :- !.

tong([H|T],X :- tong(T,X1), X is X1 + H.

**//Kiem tra so nguyen co nam trong danh sach khong**

ptu(H,[H|\_]) :- !.

ptu(H,[\_|T]) :- ptu(H,T).

**//Xac dinh phan tu thu n cua danh sach**

ptn([H|\_]),1,H) :- !.

ptn([\_|T],N,X) :- N1 = N – 1, ptn(T,N1,X).

**//Tao ra danh sach gom cac phan tu le cua danh sach ban dau**

ptle([], []) :- !.

ptle([H|T],[H|T1]) :- H mod 2 = 1, ptle(T,T1), !.

ptle([\_|T],T1) :- ptle(T|T1).

**III. Bai tap sv tu hoc**

**1.**

thich('mary','food').

thich('mary','wine').

thich('john','wine').

thich('john','mary').

**2.**

nuoitrongnha(X) :-thunuoinho(X).

thunuoi(cho).

thunuoi(meo).

thunuoinho(X) :-cholongxu(X).

cholongxu('Pluto').

**3.**

nguoi(marcus).

pompeian(marcus).

lama(X) :- pompeian(X).

lanhchua(ceasar).

amsat(X,Y) :- lama(X),khongtrungthanh(X,Y),lanhchua(Y).

khongtrungthanh(marcus,ceasar).

**4.**

men(john).

men(fred).

men(harry).

women(mary).

women(julie).

women(susan).

women(anne).

hair(john,blonde).

hair(fred,dark).

hair(harry,dark).

hair(julie,blonde).

hair(susan,blonde).

hair(mary,brunette).

hair(anne,brunette).

gold(fred).

gold(julie).

rich(X) :-gold(X).

like(john,X) :- rich(X),women(X).

like(harry,X) :-rich(X),women(X).

like(john,X) :-hair(X,blonde),women(X).

like(fred,X) :-hair(X,brunette),women(X).

like(mary,X) :-hair(X,dark),men(X).

like(julie,X) :-hair(X,dark),men(X).

like(julie,X) :-rich(X), men(X).

thichlannhau(X,Y) :-like(X,Y),like(Y,X).

car(john).

house(anne).

**5.**

hanhphuc(X) :-khongngheo(X),thongminh(X).

thongminh(X) :-docsach(X).

docsach(an).

khongngheo(an).

soidong(X) :-hanhphuc(X).

**6.**

thanhvien(long).

thanhvien(trinh).

thanhvien(hung).

thanhvien(anh).

cuoi(long,trinh).

anhtrai(hung,anh).

vo(X,Y) :-cuoi(Y,X).

thamgia(X) :-vo(X,Y),thanhvien(Y).

cuochopdienratai(long).

**7.**

hpsudungcongnghe(tc333).

hpde(X) :-hpsudungcongnghe(X).

hpkho(trituenhantao).

like(tuan,X) :- hpde(X).

**8.**

% Marcus là một người đàn ông.

nguoi\_dan\_ong(marcus).

% Marcus là người Pompeian.

nguoi\_pompeian(marcus).

% Marcus được sinh ra vào năm 40 (sau công nguyên).

nam\_sinh(marcus, 40).

% Mọi người đều phải chết.

phai\_chet(Nguoi) :- nguoi\_dan\_ong(Nguoi).

% Tất cả người Pompeian đều bị chết khi núi lửa phun trào vào năm 79 (sau công nguyên).

chay\_nui\_lua(Nguoi) :- nguoi\_pompeian(Nguoi), nam\_sinh(Nguoi, NamSinh), NamSinh < 79.

% Không cái gì phải chết sống lâu hơn 150 năm.

khong\_chet\_sau\_150\_nam(Nguoi) :- nguoi\_dan\_ong(Nguoi), nam\_sinh(Nguoi, NamSinh), NamSinh + 150 >= 2000.

% Bây giờ là năm 2000.

nam\_hien\_tai(2000).

% Còn sống nghĩa là không chết.

con\_song(Nguoi) :- nguoi\_dan\_ong(Nguoi), nam\_hien\_tai(HienTai), \+ chay\_nui\_lua(Nguoi), khong\_chet\_sau\_150\_nam(Nguoi), nam\_sinh(Nguoi, NamSinh), NamSinh < HienTai.

% Nếu một người bị chết, người đó sẽ chết từ đó về sau.

chet(Nguoi) :- \+ con\_song(Nguoi).

% Truy vấn kiểm tra liệu Marcus còn sống không trong năm 2000.

?- con\_song(marcus).

**Tong quan ve Prolog**

**2.**

thucan('ga').

thucan('tao').

thucan(Y) :- an(X,Y), consong(X).

consong('Bill').

an('Bill', 'dauphong').

an('John', X) :- thucan(X).

an('Sue', X) :- an('Bill', X).

**2 PROLOG**

**2.2 Cay gia pha**

male(jamesI).

male(charlesI).

male(charlesII).

male(jameII).

male(georgeI).

female(catherine).

female(elizabeth).

female(sophia).

parent(jamesI,charlesI).

parent(jamesI,elizabeth).

parent(charlesI,catherine).

parent(charlesI,charlesII).

parent(charlesI,jamesII).

parent(elizabeth,sophia).

parent(sophia,georgeI).

mother(X,Y) :- parent(X,Y),female(X).

father(X,Y) :- parent(X,Y),male(X).

sibling(X,Y) :- parent(P,X),parent(P,Y).

**2.3 HaNoi tower**

move(1, A, B, \_) :- print(A), print('-->'),print(B),nl.

move(N, A, B, C) :- N1 is N-1,

move(N1,A,C,B),

move(1,A,B,C), move(N1,C,B,A).

2.4 So hoc

**Giai thua**

giaithua(0,1).

giaithua(N,R) :- N>0 , N1 is N-1, giaithua(N1,R1), R is R1\*N.

**Fibonaci**

fibo(1,1).

fibo(2,1).

fibo(N,R) :- N>2, N1 is N-1, N2 is N-2, fibo(N1,R1), fibo(N2,R2), R is R1+R2.

**So nguyen to**

divide(N, P) :- P mod N =:= 0.

divide(N,P) :- P mod N\=0, N\*N<P, N1 is N+1, divide(N1,P).

prime(P):- \+divide(2,P).

**Uoc chung lon nhat**

gcd(A,0,A).

gcd(A,B,D) :- B>0, R is A mod B , gcd(B,R,D).

**Nguyen to cung nhau**

divide(N, P) :- P mod N =:= 0.

divide(N,P) :- P mod N\=0, N\*N<P, N1 is N+1, divide(N1,P).

prime(P):- \+divide(2,P).

gcd(A,0,A).

gcd(A,B,D) :- B>0, R is A mod B , gcd(B,R,D).

coprime(X,Y) :- gcd(X,Y,1), prime(X), prime(Y).

**phi**

% P34 (\*\*) Calculate Euler's totient function phi(m).

% Euler's so-called totient function phi(m) is defined as the number

% of positive integers r (1 <= r < m) that are coprime to m.

% Example: m = 10: r = 1,3,7,9; thus phi(m) = 4. Note: phi(1) = 1.

% totient\_phi(M,Phi) :- Phi is the value of the Euler's totient function

% phi for the argument M.

% (integer, integer) (+,-)

**:- ensure\_loaded(p33).**

**:- arithmetic\_function(totient\_phi/1).**

**totient\_phi(1,1) :- !.**

**totient\_phi(M,Phi) :- t\_phi(M,Phi,1,0).**

% t\_phi(M,Phi,K,C) :- Phi = C + N, where N is the number of integers R

% such that K <= R < M and R is coprime to M.

% (integer,integer,integer,integer) (+,-,+,+)

**t\_phi(M,Phi,M,Phi) :- !.**

**t\_phi(M,Phi,K,C) :-**

**K < M, coprime(K,M), !,**

**C1 is C + 1, K1 is K + 1,**

**t\_phi(M,Phi,K1,C1).**

**t\_phi(M,Phi,K,C) :-**

**K < M, K1 is K + 1,**

**t\_phi(M,Phi,K1,C).**

**Danh sach**

**1.Dem so phan tu cua 1 danh sach**

count([],0).

count([\_|T],N) :- count(T,N1), N is N1 + 1.

**2.Kiem tra 1 phan tu co nam trong danh sach khong**

member(X,[X|\_]).

member(X,[Y|L]) :- X\=Y,member(X,L).

**3.Them phan tu X vao cuoi danh sach**

push\_back(X,L,R).

push\_back(X,[],[X]).

push\_back(X,[H|T],R) :- push\_back(X,T,R1),R = [H|R1].

**4.Noi 2 danh sach**

noi([],L,L).

noi([X|L1],L2,[X|L3]) :- noi(L1,L2,L3).

**5.dao thu tu danh sach**

reverse([],[]).

reverse ([X|Y],R) :- reverse(Y,R1),append(R1,[X],R).

**6.dem so phan tu co gia tri la X**

count(\_,[],0).

count(X,[X|Y],Count) :- count(X,Y,SubCount),Count is SubCount+1.

count(X,[Y|Z],Count) :- X\=Y, count(X,Z,Count).

**7.loai bo phan tu trung nhau**

remove\_duplicates([],[]).

remove\_duplicates([X|Y],R) :- member(X,Y),remove\_duplicates(Y,R).

remove\_duplicates([X|Y],[X,R]) :- \+ member(X,Y),remove\_duplicates(Y,R).

**8.Uoc nguyen to**

is\_prime(2).

is\_prime(3).

is\_prime(p) :- p>3,p mod 2 =\= 0,\+ has\_factor(p,3).

has\_factor(n,factor) :- n mod factor =:= 0.

has\_factor(n,factor) :- factor \* factor < n,nextfactor is factor + 2,has\_factor(n,nextfactor).

prime\_factors(n,factors) :- prime\_factors(n,2,factors).

prime\_factors(1,\_,[]) :- !.

prime\_factors(n,\_,[n]) :- is\_prime(n),!.

prime\_factors(n,factor,[factor|factors]):- factor \* factor =< n,n mod factor =:= 0,nextn is n // factor,prime\_facors(nextn,factor,factors).

prime\_factors(n,factor,factors) :- nextfactor is factor + 1,prime\_factors(n,nextfactor,factors).

**Do thi**

**do thi**

edge(1,2).

edge(1,5).

edge(2,3).

edge(2,5).

edge(3,4).

edge(4,5).

edge(4,6).

edge(7,8).

canh(x,y) :- edge(x,y).

canh(x,y) :- edge(y,x).

lien\_thong(x,y) :- canh(x,y).

lien\_thong(x,y) :- lien\_thong(x,z),canh(z,y).

member(x,[x|\_]).

member(x,[y|t]) :- x \= y, member(x,t).

co\_duong\_di(y,y,\_).

co\_duong\_di(x,y,p) :- canh(x,z), \+member(z,p), co\_duong\_di(z,y,[z|p]).

lien\_thong(x,y) :- co\_duong\_di(x,y,[x]).

**bieu dien do thi nhu du lieu**

canh(x,y,[[x,y]|\_]).

canh(x,y,[[y,x]|\_]).

canh(x,y,[\_|t]) :- canh(x,y,t).

member1(x,[x|\_]).

member1(x,[y|t]) :- x\=y, member1(x,t).

co\_duong\_di(\_,y,y,\_).

co\_duong\_di(e,x,y,p) :- canh(x,z,e), \+member1(z,p), co\_duong\_di(e,z,y,[z|p]).

lien\_thong(e,x,y) :- co\_duong\_di(e,x,y,[x]).