

% Pets Prolog

:- dynamic(known/3).

:- discontiguous pet\_type/1.

pet(dogs):- pet\_type(mammal), food(dog\_food), shelter(dog\_house), domestication(easy).

pet(cats):- pet\_type(mammal), food(cat\_food), shelter(cat\_foam), domestication(easy).

pet(rabbit):- pet\_type(mammal), food(vegetables), shelter(cage), domestication(moderate).

pet(hamster):- pet\_type(mammal), food(fruits), shelter(small\_cage), domestication(moderate).

pet(ferret):- pet\_type(mammal), food(kibble), shelter(small\_cage), domestication(easy).

pet(guienea\_pigs):- pet\_type(mammal), food(pellets), shelter(small\_cage), domestication(moderate).

pet(parrot):- pet\_type(bird), food(pelleted\_food), shelter(bird\_cage), domestication(easy).

pet(canaries):- pet\_type(bird), food(seed\_mix), shelter(bird\_cage), domestication(hard).

pet(finches):- pet\_type(bird), food(leafy\_greens), shelter(bird\_cage), domestication(moderate).

pet(cockatiels):- pet\_type(bird), food(sprouted\_seeds), shelter(bird\_cage), domestication(moderate).

pet(budgies):- pet\_type(bird), food(budgie\_seed\_mix), shelter(bird\_cage), domestication(hard).

pet(turtle):- pet\_type(reptile), food(vegetables), shelter(small\_swamp), domestication(hard).

pet(snake):- pet\_type(reptile), food(prey\_items), shelter(terrarium), domestication(hard).

pet(lizard):- pet\_type(reptile), food(insects), shelter(vivarium), domestication(hard).

pet(chameleon):- pet\_type(reptile), food(gut\_loading), shelter(vivarium), domestication(hard).

pet(tortoise):- pet\_type(reptile), food(pellets), shelter(swamp), domestication(moderate).

pet(gold\_fish):- pet\_type(fish), food(flakes), shelter(fish\_tank), domestication(moderate).

pet(beta\_fish):- pet\_type(fish), food(frozen\_foods), shelter(fish\_bowl), domestication(hard).

pet(guppies):- pet\_type(fish), food(pelleted\_food), shelter(fish\_pond), domestication(hard).

main :- show\_pet.

show\_pet :- retractall(known(\_, \_, \_)), pet(X), write('That Pet is a '), write(X), nl.

show\_pet :- write('I can''t identify that Pet'), nl.

pet(X) :- ask(pet, X).

pet\_type(X) :- ask(pet\_type, X).

skin(X) :- ask(skin, X).

cost(X) :- ask(cost, X).

comfort\_level(X) :- ask(comfort\_level, X).

food(X) :- ask(food, X).

shelter(X) :- ask(shelter, X).

domestication(X) :- menuask(domestication, X, [easy, moderate, hard]).

pet\_type(mammal) :- skin(furry), cost(low), comfort\_level(high).

pet\_type(bird) :- skin(feather), cost(high), comfort\_level(moderate).

pet\_type(reptile) :- skin(scale), cost(high), comfort\_level(low).

pet\_type(fish) :- skin(fins), cost(moderate), comfort\_level(low).

% Rest of code for ask/1, menuask/3, etc.

ask(Attribute, Value) :- known(yes, Attribute, Value), !.

ask(Attribute,Value):-known(\_,Attribute,Value), !, fail.

ask(Attribute,\_):- known(yes,Attribute,\_), !, fail.

ask(A,V):- write(A:V),

% if we get here, we need to ask

  write('? (yes or no): '), read(Y),

% get the answer

  asserta(known(Y,A,V)),

  Y = yes.

% succeed or fail based on answer

menuask(Attribute, Value, \_) :- known(yes, Attribute, Value), !.

menuask(Attribute, \_, \_) :-known(yes, Attribute, \_), !, fail.

menuask(Attribute,AskValue,Menu):-

  nl,write('What is the value for '),write(Attribute),write('?'),nl,

  display\_menu(Menu),

  write('Enter the number of choice> '),

  read(Num),nl,

  pick\_menu(Num,AnswerValue,Menu),

  asserta(known(yes,Attribute,AnswerValue)),

  AskValue = AnswerValue.

display\_menu(Menu):- disp\_menu(1,Menu), !.

disp\_menu(\_, []).

disp\_menu(N, [Item | Rest]) :- write(N), write(' : '), write(Item), nl, NN is N + 1, disp\_menu(NN, Rest).

pick\_menu(N,Val,Menu):- integer(N),

pic\_menu(1,N,Val,Menu), !.

pick\_menu(Val,Val,\_).

pic\_menu(\_,\_,none\_of\_the\_above,[]).

pic\_menu(N,N, Item, [Item|\_]).

pic\_menu(Ctr,N, Val, [\_|Rest]):- NextCtr is Ctr + 1, pic\_menu(NextCtr, N, Val, Rest).