ARTIFICIAL INTELLIGENCE
ReadMe

Rakesh Kumar Mahato
RKM190000

Instructions:

1. The Knowledge Base axioms are in the file “KB_axioms.txt”.
2. Paste the Assumptions in assumptions box and Goals in goal box.
3. The Goals are commented with a “%” sign.
4. Only one goal can be uncommented for running at a time.
5. Comment the previous run goal if a new goal is to be run
6. The 9 questions of 12.5 and 6 questions of 12.6 are already written in goals

Assumptions:

```
all x (Veggie(x) -> FoodItem(x)).
all x (Beef(x) -> Meat(x)).
all x (Meat(x) -> FoodItem(x)).
all x (FoodItem(x) -> Item(x)).

all x (PersonalCare(x) -> Item(x)).
all x (Child(x) -> Person(x)).
all x (Adult(x) -> Person(x)).
all x all y (Eats(x,y) & Meat(y) -> -Vegetarian(x)).

all x all y all q ((Purchase(x,y,q)) & Item(y) -> Buys(x,y) & Has(x,y) &
Quantity(x,y,q)).
all x all y all q(Purchase(x,y,q) & FoodItem(y) -> Eats(x,y)).
all x all y (Buys(x,y) -> (Bring(x,m) & Money(m)) | (Bring(x,c)&
CreditCard(c))).
%all x all y (Buys(x,y) -> (Bring(x,c)& CreditCard(c))).
all x all y (Buys(x,y) -> FitInCarTrunkOf(y,x)).

%1b. Does John now have at least two tomatoes?
all x all y (AtLeast(x,y,5) -> AtLeast(x,y,4)).
all x all y (AtLeast(x,y,4) -> AtLeast(x,y,3)).
all x all y (AtLeast(x,y,3) -> AtLeast(x,y,2)).
all x all y (AtLeast(x,y,2) -> AtLeast(x,y,1)).
all x all y all q (Purchase(x,y,q) & Item(y) -> AtLeast(x,y,q)).

AtLeast(Safeway,Tomato,20).

Location(Safeway,NorthBerkley).

SuperMarket(Safeway).
PersonalCare(Deodrant).
FoodItem(Pizza).
FoodItem(Cake).
Meat(Chicken).
%SuperMarket can own items in 2 ways to sell. Either make in-house or purchase
from outside market and dealers.
all x all y all q (Purchase(x,y,q) & SuperMarket(x) -> Owns(x,y,q) & -
MadeIn(x,y)).
all x all y all q (MadeInhouse(x,y,q) & SuperMarket(x) -> Owns(x,y,q) &
MadeIn(x,y)).
all x all y all q (SuperMarket(x) & Item(y) & Owns(x,y,q) -> Sells(x,y)).
Purchase(Safeway, Deodrant,20).
Purchase(Safeway, Tomato,30).
```

```
Purchase(Safeway, GroundBeef,15).
Purchase(Safeway, Chicken,15).
MadeInhouse(Safeway, Pizza, 10).
MadeInhouse(Safeway, Cake, 10).

all x all y all q (Owns(x,y,q) & SuperMarket(Safeway) ->
owns(SafewayCorporation,y)).

%1d. If Mary was buying tomatoes at the same time as John, did he see her?
Purchase(Mary, Tomato,4).
SameTime(John, Mary).
all x all y all z (SameTime(x,y) & Buys(x,z) & Buys(y,z) -> Sees(x,y) &
Sees(y,x)).

%1i. Does John have less money after going to the supermarket?
all x all y all q (Purchase(x,y,q) & HasMoney(x,m1) & Money(m1) ->
HasMoney(x,m2) & Money(m2) & LessThan(m2,m1)).
%Bring(John,money).
CreditCard(cc).
Money(m).
HasMoney(John,m1).
Money(m1).

%2a. Are there other people in Safeway while John is there?
all x all y exists z ( Buys(x,y) -> OtherPeople(x,z) & Staff(z)).
ShopFrom(John,Safeway).

%2e Does the Shell station next door have any gas?
all x exists y (SuperMarket(x) -> NextDoor(x,y) & GasStation(y)).
HasGas(ShellStation).
GasStation(ShellStation).

Beef(GroundBeef).
Person(John).
Person(Mary).
Veggie(Tomato).

Purchase(John, GroundBeef,1).
Purchase(John, Tomato,2).

%1a. Is John a child or an adult?
all x (Child(x) -> Person(x)).
all x (Adult(x) -> Person(x)).
all x (Person(x) -> Child(x) | Adult(x)).

all x (Child(x) <-> -Adult(x)).
all x (-Child(x) <-> Adult(x)).

all x (Child(x) -> CanCarryMaxQty(x,1)).
all x (Adult(x) -> CanCarryMaxQty(x,5)).

all x (CanCarryMaxQty(x,1) -> -CanCarry(x,2) & CanCarry(x,1)).
all x (CanCarryMaxQty(x,2) -> -CanCarry(x,3) & CanCarry(x,2)).
all x (CanCarryMaxQty(x,3) -> -CanCarry(x,4) & CanCarry(x,3)).
all x (CanCarryMaxQty(x,4) -> -CanCarry(x,5) & CanCarry(x,4)).
all x (CanCarryMaxQty(x,5) -> -CanCarry(x,6) & CanCarry(x,5)).

all x (CanCarry(x,6) -> CanCarry(x,5)).
all x (CanCarry(x,5) -> CanCarry(x,4)).
all x (CanCarry(x,4) -> CanCarry(x,3)).
all x (CanCarry(x,3) -> CanCarry(x,2)).
all x (CanCarry(x,2) -> CanCarry(x,1)).
```

```
all x (-CanCarry(x,1) -> -CanCarry(x,2)).
all x (-CanCarry(x,2) -> -CanCarry(x,3)).
all x (-CanCarry(x,3) -> -CanCarry(x,4)).
all x (-CanCarry(x,4) -> -CanCarry(x,5)).
all x (-CanCarry(x,5) -> -CanCarry(x,6)).

all x all y all q (Purchase(x,y,q) & Item(y) ->  CanCarry(x,q)).
all x all q(CanCarryMax(x,q) -> CanCarry(x,q)).
```

Goals:

```
%12.5
%1a. Is John a child or an adult?
%Child(John).
-Child(John).
%Adult(John).
%-Adult(John).

%1b. Does John now have at least two tomatoes?
%Has(John, Tomato) & AtLeast(John,Tomato,2).

%1c. Did John buy any meat?
%exists x (Buys(John, x) & Meat(x)).

%1d. If Mary was buying tomatoes at the same time as John, did he see her?
%Sees(Mary,John).

%1e. Are the tomatoes made in the supermarket?
%MadeIn(Safeway, Tomato).

%1f. What is John going to do with the tomatoes?
%Eats(John, Tomato).

%1g. Does Safeway sell deodorant?
%Sells(Safeway, Deodrant).

%1h. Did John bring some money or a credit card to the supermarket?
%Bring(John,m) | Bring(John,cc).

%1i. Does John have less money after going to the supermarket?
%HasMoney(John, m2) & LessThan(m2,m1).


%12.6
%2a. Are there other people in Safeway while John is there?
%exists x(ShopFrom(John,Safeway) & OtherPeople(John, x)).

%2b.Is John a vegetarian?
%-Vegetarian(John).

%2c. Who owns the deodorant in Safeway?
%-owns(SafewayCorporation,Deodrant).

%2d. Did John have an ounce of ground beef?
%Has(John,GroundBeef) & Quantity(John, GroundBeef,1).

%2e. Does the Shell station next door have any gas?
%exists y(NextDoor(Safeway,y) & GasStation(ShellStation)&
HasGas(ShellStation)).

%2f. Do the tomatoes fit in John's car trunk?
%FitInCarTrunkOf(Tomato,John).
```

Example Output (to show that John is not a child)

```
      ============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 24432 was started by userr on Rakesh-PC,
Thu Apr 29 19:06:30 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-
win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.05) seconds.
% Length of proof is 20.
% Level of proof is 5.
% Maximum clause weight is 6.
% Given clauses 24.

1 (all x (Veggie(x) -> FoodItem(x))) # label(non_clause).  [assumption].
4 (all x (FoodItem(x) -> Item(x))) # label(non_clause).  [assumption].
31 (all x (Child(x) -> CanCarryMaxQty(x,1))) # label(non_clause).
[assumption].
33 (all x (CanCarryMaxQty(x,1) -> -CanCarry(x,2) & CanCarry(x,1))) #
label(non_clause).  [assumption].
48 (all x all y all q (Purchase(x,y,q) & Item(y) -> CanCarry(x,q))) #
label(non_clause).  [assumption].
50 -Child(John) # label(non_clause) # label(goal).  [goal].
51 Veggie(Tomato).  [assumption].
52 -Veggie(x) | FoodItem(x).  [clausify(1)].
60 -FoodItem(x) | Item(x).  [clausify(4)].
63 FoodItem(Tomato).  [resolve(51,a,52,a)].
75 -Child(x) | CanCarryMaxQty(x,1).  [clausify(31)].
76 Child(John).  [deny(50)].
99 Purchase(John,Tomato,2).  [assumption].
100 -Purchase(x,y,z) | -Item(y) | CanCarry(x,z).  [clausify(48)].
145 Item(Tomato).  [resolve(63,a,60,a)].
182 -Item(Tomato) | CanCarry(John,2).  [resolve(100,a,99,a)].
222 -CanCarryMaxQty(x,1) | -CanCarry(x,2).  [clausify(33)].
237 CanCarryMaxQty(John,1).  [resolve(76,a,75,a)].
266 CanCarry(John,2).  [resolve(182,a,145,a)].
270 $F.  [resolve(237,a,222,a),unit_del(a,266)].

============================== end of proof ==========================
```