**Task 1 & 2 – Design and Create a Knowledge Base & Convert KB to First Order Logic**

1. all x (Tomato(x) -> FoodItem(x)).

All objects x that belong to Tomato Class, is a part of the food item class.

2. all x (Tomato(x) -> Veggie(x)).

All items x belonging to the class Tomato also belong to the class Veggie.

Veggie here has been used for simplicity to convey the information that a person that consumes an item that belongs to the Veggie(x) class will be a Vegetarian(x).

3. all x (Veggie(x) -> Item(x)).

All objects x that belong to the Veggie Class, is a part of the Item Class.

4. all x (Beef(x) -> Meat(x)).

All objects x that belong to the Beef Class, is a part of the Meat Class.

5. all x (Meat(x) -> FoodItem(x)).

All objects x that belong to the Meat Class, is a part of the food item class.

6. all x (Deodorant(x) -> PersonalCare(x)).

All objects x that belong to the Deodorant Class, belong to the PersonalCare Class.

7. all x (PersonalCare(x) -> Item(x)).

All objects x that belong to the PersonalCare Class, belong to the Item Class.

8. all x all y (Eats(x, y) & FoodItem(y) & Meat(y) -> -Vegetarian(x)).

All people who eat meat are non-vegetarians.

9. Person(John). [Instantiation]

John is a person.

10. all x all y all q (SuperMarket(x) & Item(y) & Owns(x, y, q) & Quantity(q) -> Sells(x, y, q)).

If the supermarket owns something, then they sell it.

11. Has(x, y, q).

Entity x has an object of item class y and the amount owned is represented by q which is an object of class Quantity.

12. all x all y all q (Buys(x, y, q) & Item(y) -> Owns(x, y, q)).

If someone buys an object of item class, then they own it.

13. all x all y all q (Buys(x, y, q) & FoodItem(y) -> Eats(x, y)).

If someone buys something and it is a food item, then they eat it.

14. all x all y all q (Buys(x, y, q) & HasMoney(a, m1) & Money(m1) -> HasMoney(x, m2) & Money(m2) & LessThan(m2, m1)).

If x buys y and x has money m then after the transaction x will have less money than before.

15. Location(Safeway, NorthBerkeley).

The location of Safeway is in NorthBerkeley as per question statement.

16. all x all y all q (Buys(x, y, q) & FoodItem(y) & Buys(x, y, q) -> Adult(x)).

For all x that buys a food item and has a quantity is an adult.

17. Tomatoweight(2). [Instantiation]

Tomato weight is 3.

18. Beefweight(1). [Instantiation]

Beef weight is 3.

19. assign(max\_weight, 3).

Maximum weight is equal to 3.

20. all x (Customer(x) & max\_weight(3) -> Adult(x)).

For every x there is a customer with a max\_weight of 3, who is an Adult.

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

For every x if they are an adult, they are not a child.

**Report**

**Problems**

I first started to write my own knowledge base for this project, without using the axioms given. This quickly got out of control – meaning, I wasn’t able to keep them from conflicting with each other to solve any proofs. My thinking was to first just go off of the scenario from problem 12.5 and then work my way into reverse engineering the questions that needed proved. This was a mistake. After a few days of not making any progress, I then switched over to using the axioms given and adding on from there.

This was a feeble attempt because I am just not too familiar with the Prover9 syntax. I wasn’t able to figure out how to add constants. This was my biggest problem because I couldn’t solve any of the weight or tomato and beef questions. And without that knowledge it seemed impossible from there. The hardest part of this whole assignment was not fully understanding the axioms given enough to keep adding non-conflicting ones; not knowing how to create simple math equations inside of Prover9; and then not being able to ask the proper goals at the end to solve the problems.

**Strengths**

I was able to solve if John was an adult or not, but I had to do it by making quantity a constant equal to 3. This was exciting at first until I realized that almost every other question that needed proved was not going to be because if the quantity were a constant, I couldn’t differentiate between buying tomatoes, beef, or any other items. Then came the point of exhaustion where I seemingly tried to create every possible axiom I could think of and it ended up becoming more deleting them then adding to the final solution. I kept the way I was able to solve the first question and from prover9 and that is what the lone out file turned in is – along with its input file – inside the solved folder.

**Conclusion**

My attempt to change the axioms, as seen on first page, went through many, many, many different evolutions. Each making it harder and harder to figure out how to count in Prover9. It seems like it should be easy, but I was unable to capture the intent. I do believe however, that I would be able to do better with a little more practice in Prover9 syntax.

As I progressed through the project, I realized that I was able to create more simplified axioms, but every time I ran a solution, it seemed I took a step backwards. My goal was to use as little instantiations as possible – maybe to my detriment. And I also realized that going through the solutions first is NOT the way to approach this problem.

I’ve learned that being as simple as possible with your knowledge base leads to less tie-ups within the solution sets to reach the ultimate proofs. Also, be as vague as possible and make the variables way easier to understand because the more axioms you have, the harder it is to keep them from conflicting. First Order Logic (FOL) is more of an art then a science. And although it may look like I failed to implement much, I do feel with practice, that I could do way better in the future – for what it’s worth. It wasn’t the understanding of the logic or material within the homework that crippled me, but the implementation of the Prover9 system – and that makes me feel better about my experience.