Assignment 1

Question 3

1. food(bread, X) = Food(Y, soup)

Error (Food is not a valid operator)

2. Bread = soup

Unify (Bread = soup)

3. Bread = Soup

Unify

4. food(bread, X, milk) = food(Y, salad, X)

Does Not Unify (X can’t be both milk and salad)

5. manager(X) = Y

Unify (X is instantiated then the entire thing will be unified to Y)

6. meal(healthyFood(bread), drink(milk)) = meal(X,Y)

Unify (X= healthyFood(bread), Y = drink(milk)

7. meal(eat(Z), drink(milk)) = [X]

Does Not Unify (LHS isn’t a list)

8. [eat(Z), drink(milk)] = [X, Y | Z]

Unify (X = eat(Z) = eat([]), Y = drink(milk), Z = [])

9. f(X, t(b, c)) = f(l, t(Z, c))

Unify (X=l, Z=b)

10. ancestor(french(jean), B) = ancestor(A, scottish(joe))

Unify (A = french(jean), B = scottish(joe))

11. meal(healthyFood(bread), Y) = meal(X, drink(water))

Unify (X = healthyFood(bread), Y = drink(water))

12. [H|T] = [a, b, c]

Unify (H = a, T = [b,c])

13. [H, T] = [a, b, c]

Does Not Unify (LHS has 2 terms, RHS has 3 terms)

14. breakfast(healthyFood(bread), egg, milk) = breakfast(healthyFood(Y), Y, Z)

Does Not Unify. (Y cannot be both bread and egg).

15. dinner(X, Y, Time) = dinner(jack, cook(egg, oil), Evening)

Unify (X = jack, Y = cook(egg,oil), Time = Evening)

16. k(s(g), Y) = k(X, t(k))

Unify (X = s(g), Y = t(k))

17. equation(Z, f(x, 17, M), L*M, 17) = equation(C, f(D, D, y), C, E)

Does Not Unify (D cannot be both x and 17)

18. a(X, b(c, d), [H|T]) = a(X, b(c, X), b)

Does Not Unify (b is not a list, so cannot unify with [H|T])

Question 4

1. ? field(hit_transfer,engineering).

ground

2. ? lab_number(fine_arts,X).

non-ground

3. ? field(computer, literature).

ground

4. ? course(X,Y).

non-ground

5. ? student(adrian).

ground

6. ? student(anna, engineering).

ground

7. ? student(X, engineering).

Non-ground

8. ? student(X,fine-arts), course(fine_arts, Y).

non-ground

9. ? field(_,X).

non-ground

10. ? lab_number(_,X), field(X,Y).

non-ground

11. ? lab_number(X,15), field(X,Y).

non-ground

12. ? student(X), !, student(X,_). % note to cut here

non-ground

13. ? student(X), student(X,_), !.

non-ground

14. ? course(X,_), \+ student(_,X). % \+ is for negation (not)

non-ground