1. (25 pts) For each pair of atomic sentences, provide the most general unifier if it exists:

(a) P(A, A, B), P(x, y, z)

θ = {x/A, y/A, z/B} → P(A, A, B), P(A, A, B)

(b) Q(y, G(A, B)), Q(G(x, x), y)

Cannot be unified because y = G(A,B) and G(A,B) = y means G(A,B) = G(x,x), where A = x and B = x, which is false, because A =/ B

(c) R(x, A, z), R(B, y, z)

θ = {x/B, y/A} → R(B, A, z), R(B, A, z)

(d) Older(Father(y), y), Older(Father(x), John)

Father(y) = Father(x) → y = x

y = John, x = John

θ = {x/John, y/John} → Older(Father(John), John), Older(Father(John), John)

(e) Knows(Father(y), y), Knows(x, x)

Cannot be unified because it would mean Father(y) = x, and y = x. And x cannot be both Father(y) and y, which is invalid.

2. (75 pts) Consider the following sentences:

(a) Translate these sentences into formulas in first-order logic.

- A x (Food(x) => Likes(John, x))
- Food(Apple)
- Food(Chicken)
- A x A y ((Eats(x, y) & ~KilledBy(x, y)) => Food(y))
- A x A y (KilledBy(x, y) => ~Alive(x))
- Eats(Bill, Peanuts) & Alive(Bill)
- A x (Eats(Bill, x) => Eats(Sue, x))

(b) Convert the formulas of part (a) into CNF (also called clausal form).

1. ~Food(x) | Likes(John, x)
2. Food(Apple)
3. Food(Chicken)
4. ~Eats(x, y) | KilledBy(x, y) | Food(y)

5. ~KilledBy(x, y) | ~Alive(x)
6. Eats(Bill, Peanuts)
7. Alive(Bill)
8. ~Eats(Bill, x) | Eats(Sue, x)

(c) Prove that John likes peanuts using resolution.

- 9. ~Likes(John, Peanuts)
- 10. ~Food(Peanuts) (1, 9) θ = {x/Peanuts}
- 11. KilledBy(Bill, Peanuts) | Food(Peanuts) (4, 6) θ = {x/Bill, y/Peanuts}
- 12. ~KilledBy(Bill, Peanuts) (5, 7) θ = {x/Bill, y/Peanuts}
- 13. Food(Peanuts) (11, 12)
- 14. () (10, 13)

Contradiction, therefore Likes(John, Peanuts) is true.

(d) Use resolution to answer the question, "What does Sue eat?"

- 9. Eats(Sue, Peanuts) (6, 8) θ = {x/Peanuts}

Since Sue eats the same thing as Bill, Sue also eats peanuts.

(e) Use resolution to answer (d) if, instead of the axiom marked with an asterisk (∗) above, we had:

• If you don't eat, you die.

• If you die, you are not alive.

• Bill is alive.

- 1. A x ((A y ~Eats(x, y)) => Dies(x))
    - CNF: A x (~(A y ~Eats(x, y)) => Dies(x))

      → A x (E y Eats(x, y)) => Dies(x)

      → A x (Eats(x, f(x)) => Dies(x))

      → CNF: Eats(x, f(x)) | Dies(x)

- 2. A x (Dies(x) => ~Alive(x))
    - CNF: ~Dies(x) | ~Alive(x)

- 3. Alive(Bill)
    - CNF: Alive(Bill)

Resolution:

- 1. ~Food(x) | Likes(John, x)
- 2. Food(Apple)
- 3. Food(Chicken)
- 4. ~Eats(x, y) | KilledBy(x, y) | Food(y)
- 5. ~KilledBy(x, y) | ~Alive(x)
- (NEW) 6. Eats(x, f(x)) | Dies(x)
- (NEW) 7. ~Dies(x) | ~Alive(x)
- (NEW) 8. Alive(Bill)
- 9. ~Eats(Bill, x) | Eats(Sue, x)
- 10. ~Dies(Bill) (7, 8) {x, Bill}
- 11. Eats(Bill, f(Bill)) (6, 10) {x, Bill}
- 12. Eats(Sue, f(Bill)) (9, 11) {x, f(Bill)}

Sue also eats f(Bill), which is some type of food that Bill eats.