Homework 6

1) (a). {x/A, y/B, z/C}
(b). Not unifiable since x cannot be both A and B.
(c). {x/A, y/A}
(d). {y/John, x/John}
(e). Not unifiable since y cannot be Father(x) and x (unless Father(x) = x in some strange world. In that case it is unifiable where {x/y, Father(x)/y}).

2) (a). First-order logic:
- (A x) (Food(x) => Likes(John, x))
- Food(Apples)
- Food(Chicken)
- (A x, y) (Eats(x, y) & ~Killed(y, x) => Food(y))
- (A x, y) (Killed(y, x) => ~Alive(x))
- Eats(Bill, Peanuts) & Alive(Bill)
- (A x) (Eats(Bill, x) => Eats(Sue, x))

(b).

| **First-Order** | **CNF** |
|---|---|
| (A x) (Food(x) => Likes(John, x)) | (~Food(x) \| Likes(John, x)) |
| Food(Apples) | Food(Apples) |
| Food(Chicken) | Food(Chicken) |
| (A x, y) (Eats(x, y) & ~Killed(y, x) => Food(y)) | (~Eats(x, y) \| Killed(y, x) \| Food(y)) |
| (A x, y) (Killed(y, x) => ~Alive(x)) | (~Killed(y, x) \| ~Alive(x)) |
| Eats(Bill, Peanuts) & Alive(Bill) | Eats(Bill, Peanuts) & Alive(Bill) |
| (A x) (Eats(Bill, x) => Eats(Sue, x)) | (~Eats(Bill, x) \| Eats(Sue, x)) |

(c). Proof by Resolution: Likes(John, Peanuts)
Prove that KB & ~α is unsatisfiable.
~α = ~Likes(John, Peanuts)
- Eats(Bill, Peanuts) & Alive(Bill) & (~Killed(y, x) | ~Alive(x)) unify only when ~Killed(y, Bill) where y applies to all objects, including Peanuts so {x/Bill}
- Eats(Bill, Peanuts) & Alive(Bill) & ~Killed(y, Bill) & (~Eats(x, y) | Killed(y, x) | Food(y)) unify only when Food(Peanuts) so {x/Peanuts, y/Bill}
- ~Likes(John, Peanuts) & (~Food(x) | Likes(John, x)) unify only when ~Food(Peanuts) so {x/Peanuts}

We have reached a contradiction with Food(Peanuts) & ~Food(Peanuts) so we have proved that Likes(John, Peanuts) which means that John likes Peanuts.

(d). “What food does Sue eat?”
Eats(Bill, Peanuts) & Alive(Bill) & (~Eats(Bill, x) | Eats(Sue, x)) unify only when Eats(Sue, Peanuts) so {x/Peanuts}. Through resolution we have showed that Sue eats Peanuts.

(e). “What food does Sue eat?”
- If you don’t eat, you die. (A x) (((A y) (~Eats (x,y)) => Dies(x))
- If you die, you are not alive. (A x) (Dies(x) => ~Alive(x))
- Bill is alive. Alive(Bill)

We cannot use resolution to answer this question with the above statements. Since we are not given any axioms indicating what Bill or Sue eat, we cannot use resolution to prove what Sue eats. There is simply not enough information.