1. **Translate sentences into First-Order Logic**
    a. **bowling balls are sporting equipment**
        i. $\forall x\ isBowlingBall(x) \rightarrow isSportingEquipment(x)$
    b. **horses are faster than frogs (there are many ways to say this in FOL; try expressing it this way: "all horses have a higher speed than any frog")**
        i. $\forall h \forall f\ isHorse(f) \land isFrog(f) \rightarrow fasterThan(h, f)$
    c. **all domesticated horses have an owner**
        i. $\forall h\ isHorse(h) \land isDomesticated(h) \rightarrow \exists o\ isOwner(o, h)$
    d. **the rider of a horse can be different than the owner**
        i. $\forall h \exists r \exists o\ isHorse(h) \land isRider(r, h) \land isOwner(o, h) \rightarrow (r = o \lor r \neq o)$
    e. **a finger is any digit on a hand other than the thumb**
        i. $\forall x\ isDigit(x) \land onHand(x) \land \neg isThumb(x) \rightarrow isFinger(x)$
    f. **an isosceles triangle is defined as a polygon with 3 edges connected at 3 vertices, where 2 (but not 3) edges have the same length**
        i. $\forall t\ isIsoscelesTriangle(t) \leftrightarrow isPolygon(t) \land hasEdges(t, 3) \land hasVertices(t, 3) \land \exists e1 \exists e2 \exists e3\ isEdge(e1, t) \land isEdge(e2, t) \land isEdge(e3, t) \land haveEqualLength(e1, e2) \land \neg haveEqualLength(e1, e3) \land \neg haveEqualLength(e2, e3)$

2. **Convert FOL sentence into CNF**

$\forall x\ person(x) \land [\exists z\ petOf(x, z) \land \forall y\ petOf(x, y) \rightarrow dog(y)] \rightarrow doglover(x)$

1. Eliminate biconditionals and implications

$\neg[\forall x\ person(x) \land [\neg[\exists z\ petOf(x, z) \land \forall y\ petOf(x, y)] \lor dog(y)]] \lor doglover(x)$

2. Move ¬ inwards

$\neg[\forall x\ person(x) \land [\exists z\ \neg petOf(x, z) \lor \forall y\ \neg petOf(x, y) \lor dog(y)]] \lor doglover(x)$

$[\forall x\ \neg person(x) \lor \exists z\ petOf(x, z) \lor \forall y\ petOf(x, y) \lor \neg dog(y)] \lor doglover(x)$

$\forall x\ \neg person(x) \lor \exists z\ petOf(x, z) \lor \forall y\ petOf(x, y) \lor \neg dog(y) \lor doglover(x)$

3. Skolemize

$\forall x\ \neg person(x) \lor \exists z\ petOf(x, z) \lor \forall y\ petOf(x, y) \lor \neg dog(y) \lor doglover(x)$

$\forall x\ \neg person(x) \lor petOf(x, f(x)) \lor \forall y\ petOf(x,y) \lor \neg dog(y) \lor doglover(x)$

4. Drop universal quantifiers

$\neg person(x) \lor petOf(x, f(x)) \lor petOf(x,y) \lor \neg dog(y) \lor doglover(x)$

5. Distribute ∧ over ∨

Not needed already done

3. **Determine whether or not the following pairs of predicates are unifiable. If they are, give the most-general unifier and show the result of applying the substitution to each predicate. If they are not unifiable, indicate why**
    a. **owes(owner(X), citibank, cost(X)) and owes(owner(Ferrari), Z, cost(Y))**

owner(X) and owner(Ferrari) are unifiable: {X / Ferrari}
Citibank and Z are unifiable: {Z / citibank}
cost(X) and cost(Y) are unifiable: {Y / X}
Combine to get the most-general unifier: {X / Ferrari, Z / citibank, Y / X}

If we apply the MGU to each predicate we get:
- owes(owner(Ferrari), citibank, cost(Ferrari))
- owes(owner(Ferrari), citibank, cost(Ferrari))

    b. **gives(bill, jerry, book21) and gives(X, brother(X), Z)**

Bill and X are unifiable: {X / bill}
Jerry and brother(X) are not unifiable because jerry is a constant, while brother(X) is a function symbol applied to a variable - so they are not unifiable.
Therefore the predicates are not unifiable.

    c. **opened(X, result(open(X), s0))) and opened(toolbox, Z)**

X and toolbox are unifiable: {X / toolbox}
result(open(X), s0) and Z are unifiable: {Z / result(open(toolbox, s0))}
Combine to get the most-general unifier: {X / toolbox, Z / result(open(toolbox, s0))}

If we apply the MGU to each predicate we get:
- opened(toolbox, result(open(toolbox), s0))
- opened(toolbox, result(open(toolbox), s0))

4. **Consider the following situation:**
    a. **Translate the sentences into first-order logic**

- Marcus is a Pompeian
    - isPompeian(Marcus)
- All Pompeians are Romans
    - $\forall x$ (isPompeian(x) → isRoman(x))
- Caesar is a ruler
    - isRuler(Caesar)
- All Romans are either loyal to Caesar or hate Caesar (but not both)
    - $\oplus$ is XOR
    - $\forall x$ (isRoman(x) → (LoyalTo(x, Caesar) $\oplus$ Hates(x, Caesar)))
- Everyone is loyal to someone
    - $\forall x \; \exists y$ (isLoyalTo(x, y))
- People only try to assassinate rulers they are not loyal to
    - $\forall x \; \exists y$ ((tryToAssassinate(x, y) ∧ isRuler(y)) → ¬isLoyalTo(x, y))
- Marcus tries to assassinate Caesar
    - tryToAssassinate(Marcus, Caesar)

    b. **Prove that Marcus hates Caesar using Natural Deduction. Label all derived sentences with the ROI and which prior sentences and unifier were used.**

1. isPompeian(Marcus)
    a. Premise
2. $\forall x$ (isPompeian(x) → isRoman(x))
    a. Premise
3. isRoman(Marcus)
    a. From 1, 2 by Universal Instantiation (UI) and Implication Elimination (IE)
4. isRuler(Caesar)
    a. Premise
5. $\forall x$ (isRoman(x) → (LoyalTo(x, Caesar) $\oplus$ Hates(x, Caesar)))
    a. Premise
6. LoyalTo(x, Caesar) $\oplus$ Hates(x, Caesar)
    a. From 3, 5 by UI and IE
7. tryToAssassinate(Marcus, Caesar)
    a. Premise
8. $\forall x \; \exists y$ ((tryToAssassinate(x, y) ∧ isRuler(y)) → ¬isLoyalTo(x, y))
    a. Premise
9. tryToAssassinate(Marcus, Caesar) ∧ isRuler(Caesar)
    a. From 4, 7 by And Introduction (AI)
10. ¬isLoyalTo(Marcus, Caesar)
    a. From 8, 9 by UI and IE
11. Hates(Marcus, Caesar)
    a. From 6, 10 by negating the isLoyalTo part of the XOR

Therefore Marcus Hates Caesar

c. Convert all the sentences into CNF

- isPompeian(Marcus)
    - Already CNF
- ∀x (isPompeian(x) → isRoman(x))
    - ∀x (¬isPompeian(x) ∨ isRoman(x))
    - ¬isPompeian(x) ∨ isRoman(x)
- isRuler(Caesar)
    - Already CNF
- ∀x (isRoman(x) → (LoyalTo(x, Caesar) ⊕ Hates(x, Caesar)))
    - ∀x (¬isRoman(x) ∨ (LoyalTo(x, Caesar) ⊕ Hates(x, Caesar)))
    - ¬isRoman(x) ∨ (LoyalTo(x, Caesar) ⊕ Hates(x, Caesar))
    - (¬isRoman(x) ∨ (LoyalTo(x, Caesar) ∧ ¬Hates(x, Caesar))) ∨ (¬isRoman(x) ∨ (¬LoyalTo(x, Caesar) ∧ Hates(x, Caesar)))
- ∀x ∃y (isLoyalTo(x, y))
    - ∀x (isLoyalTo(x, f(x)))
    - isLoyalTo(x, f(x))
- ∀x ∃y ((tryToAssassinate(x, y) ∧ isRuler(y)) → ¬isLoyalTo(x, y))
    - ∀x ∃y (¬(tryToAssassinate(x, y) ∧ isRuler(y)) ∨ ¬isLoyalTo(x, y))
    - ∀x ∃y (¬tryToAssassinate(x, y) ∨ ¬isRuler(y) ∨ ¬isLoyalTo(x, y))
    - ∀x (¬tryToAssassinate(x, f(x)) ∨ ¬isRuler(f(x)) ∨ ¬isLoyalTo(x, f(x)))
    - ¬tryToAssassinate(x, f(x)) ∨ ¬isRuler(f(x)) ∨ ¬isLoyalTo(x, f(x))
- tryToAssassinate(Marcus, Caesar)
    - Already CNF

d. Prove that Marcus hates Caesar using Resolution Refutation.

Negate Hates(Marcus, Caesar) and demonstrate that it leads to a contradiction.

CNF sentences:

1. isPompeian(Marcus)
2. ¬isPompeian(x) ∨ isRoman(x)
3. isRuler(Caesar)
4. (¬isRoman(x) ∨ (LoyalTo(x, Caesar) ∧ ¬Hates(x, Caesar))) ∨ (¬isRoman(x) ∨ (¬LoyalTo(x, Caesar) ∧ Hates(x, Caesar)))
5. isLoyalTo(x, f(x))
6. ¬tryToAssassinate(x, f(x)) ∨ ¬isRuler(f(x)) ∨ ¬isLoyalTo(x, f(x))
7. tryToAssassinate(Marcus, Caesar)
8. ¬Hates(Marcus, Caesar)

Refutation Resolution:

a. Resolve 1, 2
    i. isRoman(Marcus)
b. Resolve 3, 6, 7
    i. ¬isLoyalTo(Marcus, Caesar)
c. Resolve a, 4
    i. isLoyalTo(Marcus, Caesar)
d. Resolve b, c
    i. Contradiction

We derived a contradiction, so the negation of our hypothesis (¬Hates(Marcus, Caesar)) cannot be true, therefore Hates(Marcus, Caesar) must be true, which proves that Marcus hates Caesar.

5. **Write a KB in First-Order Logic with rules/axioms for…**
    a. **Map-coloring – every state must be exactly 1 color, and adjacent states must be different colors. Assume possible colors are states are defined using unary predicate like color(red) or state(WA). To say a state has a color, use a binary predicate, e.g. 'color(WA,red)'.**

1. Each state is at least 1 color
    a. ∀s isState(s) → ∃c isColor(c) ∧ hasColor(s, c)
2. Each state is at most 1 color
    a. ∀s ∀c ∀d isState(s) ∧ hasColor(s, c) ∧ hasColor(s, d) → c = d
3. Neighboring states cannot have matching colors
    a. ∀s ∀t ∀c isState(s) ∧ isState(t) ∧ areNeighbors(s, t) ∧ hasColor(s, c) → ¬hasColor(s, c)

    b. **Sammy's Sport Shop – include implications of facts like obs(1,W) or label(2,B), as well as constraints about the boxes and colors. Use predicate 'cont(x,q)' to represent that box x contains tennis balls of color q (where q could be W, Y, or B).**

1. If you observe a ball, the contents are either that ball's color or both
    a. ∀x ∀c obs(x, c) → contents(x, c) ∨ contents(x, BOTH)
2. If a box is labeled something, its contents are not that something
    a. ∀x labeled(x, YELLOW) → ¬contents(x, YELLOW)
    b. ∀x labeled(x, WHITE) → ¬contents(x, WHITE)
    c. ∀x labeled(x, BOTH) → ¬contents(x, BOTH)
3. Exactly 1 box of each type
    a. ∃x contents(x, YELLOW) ∧ ∃x contents(x, WHITE) ∧ ∃x contents(x, BOTH)

c. **Wumpus World - (hint start by defining a helper concept 'adjacent(x,y,p,q)' which defines when a room at coordinates (x,y) is adjacent to another room at (p,q). Don't forget rules for 'stench', 'breezy', and 'safe'.**

1. Adjacent rooms
    a. $\forall x\ \forall y\ \forall p\ \forall q$ adjacent(x, y, p, q) $\leftrightarrow (x = p \land |y - q| = 1) \lor (y = q \land |x - p| = 1)$
2. Stench
    a. $\forall x\ \forall y\ \forall p\ \forall q$ stench(x, y) $\leftrightarrow$ ($\exists p,\ \exists q$ adjacent(x, y, p, q) $\land$ wumpus(p, q))
3. Breeze
    a. $\forall x\ \forall y\ \forall p\ \forall q$ breezy(x, y) $\leftrightarrow$ ($\exists p,\ \exists q$ adjacent(x, y, p, q) $\land$ pit(p, q))
4. Safe room
    a. $\forall x\ \forall y$ safe(x, y) $\leftrightarrow \neg$pit(x, y) $\land\ \neg$wumpus(x, y)
5. Gold
    a. $\forall x\ \forall y$ gold(x, y) $\rightarrow$ safe(x, y)

d. **4-Queens – assume row(1)…row(4) and col(1)…col(4) are facts; write rules that describe configurations of 4 queens such that none can attack each other, using 'queen(r,c)' to represent that there is a queen in row r and col c.**

1. No 2 queens can share same row or column
    a. $\forall r1\ \forall c1\ \forall r2\ \forall c2$ ((isQueen(r1, c1) $\land$ isQueen(r2, c2)) $\rightarrow$ ((r1 $\neq$ r2) $\land$ (c1 $\neq$ c2)))
2. No 2 queens can be on the same diagonal
    a. $\forall r1,\ \forall c1,\ \forall r2,\ \forall c2$ ((isQueen(r1, c1) $\land$ isQueen(r2, c2) $\land$ (r1 $\neq$ r2)) $\rightarrow$ (|r1−r2| $\neq$|c1−c2|))
3. There is a queen in each row and in each column
    a. $\forall r\ \exists c$ isQueen(r, c)
    b. $\forall c\ \exists r$ isQueen(r, c)