QS solve - Mahbub Sir

> নিচের প্রায় সবগুলো উত্তরই LLM generated, তবে যদি কোনোটা নিয়ে সংশয় সৃষ্টি হয়, feel free to ask!

Equivalence Check

> একটা কথা মাথায় রাখতে হবে যে $\models$ এই চিহ্নটা হলো logical entails, বাম পাশে সত্য আর ডান পাশে মিথ্যা এরকম কখনোই হতে পারবে না ;)

> আর == এটা হলো equivalence যেটা আমরা `tpointtech` এ পড়েছি

i. $(A \land B) \models (A \Leftrightarrow B)$

This statement is correct.

- **Clarification:** The symbol $\models$ means "logically entails." This statement asserts that in every situation where $(A \land B)$ is true, $(A \Leftrightarrow B)$ must also be true.
  - For $(A \land B)$ to be true, both **A** and **B** must be true.
  - The expression $(A \Leftrightarrow B)$ is true if and only if **A** and **B** have the same truth value.
  - Therefore, in the specific case where both **A** and **B** are true, $(A \Leftrightarrow B)$ is also true. This confirms the entailment.

  We can also see this with a truth table:

A	B	A∧B	A⇔B
T	T	**T**	**T**
T	F	F	F
F	T	F	F
F	F	F	T

The only row where $(A \land B)$ is true is the first row, and in that row, $(A \Leftrightarrow B)$ is also true.

---

ii. $A \Leftrightarrow B \models \neg A \lor B$

This statement is correct.

- **Clarification:** We need to check if $\neg A \lor B$ is true in all cases where $A \Leftrightarrow B$ is true.
  - $A \Leftrightarrow B$ is true under two conditions:
    1. Both A and B are true.

2. Both A and B are false.

- Let's check the expression $\neg A \lor B$ (which is equivalent to $A \Rightarrow B$) for these two conditions:
    1. If A=True and B=True, then $\neg T \lor T \Rightarrow F \lor T$, which is **True**.
    2. If A=False and B=False, then $\neg F \lor F \Rightarrow T \lor F$, which is **True**.
- Since $\neg A \lor B$ is true in all cases where $A \Leftrightarrow B$ is true, the entailment holds.

---

iii. $(A \lor B) \land (\neg C \lor \neg D \lor E) \models (A \lor B \lor C) \land (B \land C \land D \Rightarrow E)$

This statement is correct.

- **Clarification:** Let the left-hand side (LHS) be our premises and the right-hand side (RHS) be our conclusion. The LHS gives us two premises:
    1. $P_1 : (A \lor B)$
    2. $P_2 : (\neg C \lor \neg D \lor E)$

  The RHS is a conjunction, so we must be able to derive both parts from the premises.
    - **Part 1 of RHS:** $(A \lor B \lor C)$
        - We can derive this directly from P_1 using the **Rule of Addition** (or Disjunction Introduction), which states that if we have P, we can infer $P \lor Q$. Since we have $(A \lor B)$, we can validly infer $(A \lor B) \lor C$.
    - **Part 2 of RHS:** $(B \land C \land D \Rightarrow E)$
        - First, convert the implication to its equivalent disjunctive form: $\neg(B \land C \land D) \lor E$, which simplifies to $(\neg B \lor \neg C \lor \neg D \lor E)$ using De Morgan's laws.
        - We can derive this from P_2 using the **Rule of Addition**. Since we have $(\neg C \lor \neg D \lor E)$, we can validly infer $\neg B \lor (\neg C \lor \neg D \lor E)$.

  Since both parts of the RHS conclusion can be logically derived from the LHS premises, the entire statement is correct.

---

iv. $(A \land B) \Rightarrow C \models (A \Rightarrow C) \lor (B \Rightarrow C)$

This statement is correct.

- **Clarification:** This is a case of logical equivalence. If two expressions are logically equivalent, they also entail each other. We can prove this by converting both sides into a standard form.
    - **Left-Hand Side (LHS):** $(A \land B) \Rightarrow C$
        - $\equiv \neg(A \land B) \lor C$ (Implication Equivalence)
        - $\equiv (\neg A \lor \neg B) \lor C$ (De Morgan's Law)

- $\equiv \neg A \vee \neg B \vee C$
  - **Right-Hand Side (RHS):** $(A \Rightarrow C) \vee (B \Rightarrow C)$
    - $\equiv (\neg A \vee C) \vee (\neg B \vee C)$ (Implication Equivalence)
    - $\equiv \neg A \vee C \vee \neg B \vee C$
    - $\equiv \neg A \vee \neg B \vee C$ (by Idempotent Law, where $C \vee C \equiv C$)

Since both the LHS and RHS simplify to the exact same logical expression $(\neg A \vee \neg B \vee C)$, they are logically equivalent, and the entailment is correct.

---

v. $A \Leftrightarrow B \models A \vee B$

❌ **Incorrect.**

- **Reasoning:** The entailment fails when **A** and **B** are both false.
  - If A=False and B=False, then $(A \Leftrightarrow B)$ is **true**.
  - However, $(A \vee B)$ becomes $(F \vee F)$, which is **false**.
  - Since we found a case where the premise is true but the conclusion is false, the statement is incorrect.

---

vi. $A \Rightarrow B \models \neg A \vee B$

✅ **Correct.**

- **Reasoning:** This is a fundamental logical equivalence. The expression $A \Rightarrow B$ (A implies B) is defined as being logically equivalent to $\neg A \vee B$. Since they are equivalent, they entail each other.

---

iv. $(A \vee B) \wedge (\neg C \vee \neg D \vee E) \models (A \vee B \vee C) \wedge (B \wedge C \wedge D \Rightarrow E)$

✅ **Correct.**

- **Reasoning:** Both parts of the conclusion on the right side can be logically derived from the premises on the left side using the **Rule of Addition**, which allows you to add any proposition to an existing one with an 'OR' operator. For example, since we know $(A \vee B)$ is true, we can infer that $(A \vee B \vee C)$ must also be true.

---

v. $(A \vee B) \wedge (\neg C \vee \neg D \vee E) \models (A \vee B) \wedge (\neg D \vee E)$

❌ Incorrect.

- **Reasoning:** This statement claims that if $(\neg C \vee \neg D \vee E)$ is true, then $(\neg D \vee E)$ must also be true. This is not valid.
    - **Counterexample:** Let C=False, D=True, and E=False.
    - The premise $(\neg C \vee \neg D \vee E)$ becomes $(T \vee F \vee F)$, which is **true**.
    - However, the conclusion $(\neg D \vee E)$ becomes $(F \vee F)$, which is **false**.

---

vi. $(A \vee B) \wedge \neg(A \Rightarrow B)$ **is satisfiable**

✅ Correct.

- **Reasoning:** An expression is **satisfiable** if there's at least one assignment of truth values that makes it true. Let's simplify the expression:
    - $(A \vee B) \wedge \neg(\neg A \vee B)$
    - $(A \vee B) \wedge (A \wedge \neg B)$ (Using De Morgan's Law)
- For this entire expression to be true, the part $(A \wedge \neg B)$ must be true, which means **A must be True** and **B must be False**.
- This assignment (A=T, B=F) makes the entire expression true, so it is satisfiable.

---

vii. $(A \wedge B) \Rightarrow C \models (A \Rightarrow C) \vee (B \Rightarrow C)$

✅ Correct.

- **Reasoning:** The expressions on the left and right are logically equivalent. Both can be simplified to the same form: $\neg A \vee \neg B \vee C$. Since they are equivalent, the entailment is correct.

First Order Logic

A. Every cat loves its mother or father.

- **(i) ∀x Cat(x) ⇒ Loves(x, Mother(x) ∨ Father(x))**
    - **(2) Syntactically invalid.** The arguments to a predicate like `Loves` must be terms (variables, constants, or functions). The expression `Mother(x) v Father(x)` is a logical disjunction, not a term. You cannot "love" a logical "or".
- **(ii) ∀x ¬Cat(x) ∨ Loves(x, Mother(x)) ∨ Loves(x, Father(x))**
    - **(1) Correctly expresses the sentence.** This is logically equivalent to the standard form `∀x Cat(x) ⇒ (Loves(x, Mother(x)) v Loves(x, Father(x)))`. The equivalence rule used here is that `P ⇒ Q` is the same as `¬P v Q`.
- **(iii) ∀x Cat(x) ∧ (Loves(x, Mother(x)) ∨ Loves(x, Father(x)))**

- This expression is syntactically valid, but it **does not correctly express the sentence**. It claims that everything in the universe is a cat (∀x Cat(x)), which is incorrect. The proper logical connective to use with a universal quantifier (∀) for "Every..." statements is implication (⇒), not conjunction (∧).

B. Every dog who loves one of its brothers is happy.

- **(i) ∀x Dog(x) ∧ (∃y Brother(y, x) ∧ Loves(x, y)) ⇒ Happy(x)**
  - **(2) Syntactically invalid.** The scope of the universal quantifier ∀x is not correctly applied to the entire sentence. As written, the x in Happy(x) is a "free variable" because it's outside the scope of ∀x. The entire expression should be enclosed in brackets after the ∀x.
- **(ii) ∀x [Dog(x) ∧ (∃y Brother(y, x) ∧ Loves(x, y))] ⇒ Happy(x)**
  - **(1) Correctly expresses the sentence.** This translates to: "For any x, if x is a dog AND there exists a y such that y is a brother of x AND x loves y, THEN x is happy." This perfectly captures the sentence's meaning.
- **(iii) ∀x Dog(x) ∧ [∀y Brother(y, x) ⇒ Loves(x, y)] ⇒ Happy(x)**
  - **(2) Syntactically invalid.** This has the same scope issue as expression (i), where the x in Happy(x) is a free variable. Additionally, it incorrectly uses a universal quantifier (∀y), which would mean the dog loves **all** of its brothers, not just **one** of them.

C. No dog bites a child of its owner.

- **(i) ∀x Dog(x) ⇒ ¬Bites(x, Child(Owner(x)))**
  - **(2) Syntactically invalid.** The expression Child(Owner(x)) is being used as a function that returns a single object ("the child"), but the original sentence implies any child. Logic requires a predicate Child(y, z) (y is a child of z) and a quantifier for the child (y).
- **(ii) ¬∃x, y Dog(x) ∧ Child(y, Owner(x)) ∧ Bites(x, y)**
  - **(1) Correctly expresses the sentence.** This reads: "There does not exist an x and a y such that x is a dog, y is a child of x's owner, and x bites y." This is an accurate way to state the original sentence.
- **(iii) ∀x Dog(x) ⇒ (∀y Child(y, Owner(x)) ⇒ ¬Bites(x, y))**
  - **(1) Correctly expresses the sentence.** This is a logically equivalent and correct formulation. It reads: "For any x, if x is a dog, then for any y, if y is a child of x's owner, then x does not bite y."
- **(iv) ¬∃x Dog(x) ⇒ (∃y Child(y, Owner(x)) ∧ Bites(x, y))**

- **(2) Syntactically invalid.** The scope of `¬∃x` only covers `Dog(x)` . The variables `x` that appear later in the expression (`Owner(x)` and `Bites(x, y)`) are free and not bound by any quantifier.

D. Everyone's zip code within a state has the same first digit.

- **(i) ∀x, s, z1 [State(s) ∧ LivesIn(x, s) ∧ Zip(x)=z1] ⇒ [∀y, z2 LivesIn(y, s) ∧ Zip(y)=z2 ⇒ Digit(1, z1)=Digit(1, z2)]**
  - **(1) Correctly expresses the sentence.** Although it's written in a complex way by introducing `z1` and `z2` to represent the zip codes, it is logically sound. It states that for any person `x` in a state `s` with zip `z1` , any other person `y` in the same state with zip `z2` will have the same first digit.
- **(ii) ∀x, s [State(s) ∧ LivesIn(x, s) ∧ ∃z1 Zip(x)=z1] ⇒ [∀y, z2 LivesIn(y, s) ∧ Zip(y)=z2 ∧ Digit(1, z1)=Digit(1, z2)]**
  - **(2) Syntactically invalid.** The variable `z1` is introduced with an existential quantifier (`∃z1`) in the antecedent (the `if` part), but it is then used in the consequent (the `then` part). A variable's scope does not extend outside of the clause where its quantifier is located.
- **(iii) ∀x, y, s State(s) ∧ LivesIn(x, s) ∧ LivesIn(y, s) ⇒ Digit(1, Zip(x)=Zip(y))**
  - **(2) Syntactically invalid.** The function `Digit(...)` expects a zip code as its second argument. The expression `Zip(x)=Zip(y)` is a logical comparison that evaluates to true or false; it is not a zip code and cannot be an argument for the `Digit` function.
- **(iv) ∀x, y, s (State(s) ∧ LivesIn(x, s) ∧ LivesIn(y, s)) ⇒ Digit(1, Zip(x))=Digit(1, Zip(y))**
  - **(1) Correctly expresses the sentence.** This is the most clear and direct translation. It says: "For any two people x and y and any state s, if s is a state and both x and y live in s, then the first digit of x's zip code is equal to the first digit of y's zip code."

First order Logic - Translation

📝 Predicate Definitions

প্রশ্ন উত্তরে সুবিধার্থে কিছু function ব্যবহার করা যেতে পারে, এটা optional, যার যেভাবে খুশি ধরে নেয়া যাবে...

- **Gardener(x)**: x is a gardener.
- **Likes(x, y)**: x likes y.
- **TheSun**: A constant representing the sun.
- **Person(x)**: x is a person.
- **Time(t)**: t is a time.

- **CanFool(x, t)**: Person x can be fooled at time t.
- **Mushroom(x)**: x is a mushroom.
- **Purple(x)**: x is purple.
- **Poisonous(x)**: x is poisonous.
- **Clinton**: A constant representing Clinton.
- **Tall(x)**: x is tall.
- **Man(x)**: x is a man.
- **Fool(x)**: x is a fool.
- Surgeon(x) : x is a surgeon.
- Lawyer(x) : x is a lawyer.
- Actor(x) : x is an actor.
- Doctor(x) : x is a doctor.
- HoldsAnotherJob(x) : x holds a job in addition to one already mentioned.
- BossOf(x, y) : x is the boss of y.
- CustomerOf(x, y) : x is a customer of y.

Set - I

i. Every gardener likes the sun.

- **Answer:** $\forall x (Gardener(x) \rightarrow Likes(x, TheSun))$

ii. You can fool some of the people all of the time.

- **Answer:** $\exists x (Person(x) \land \forall t (Time(t) \rightarrow CanFool(x, t)))$

iii. You can fool all of the people some of the time.

- **Answer:** $\forall x (Person(x) \rightarrow \exists t (Time(t) \land CanFool(x, t)))$

iv. All purple mushrooms are poisonous.

- **Answer:** $\forall x ((Mushroom(x) \land Purple(x)) \rightarrow Poisonous(x))$

v. No purple mushroom is poisonous.

- **Answer:** $\forall x ((Mushroom(x) \land Purple(x)) \rightarrow \neg Poisonous(x))$

vi. There are exactly two purple mushrooms.

- **Answer:**
  $\exists x \exists y ((Mushroom(x) \land Purple(x)) \land (Mushroom(y) \land Purple(y)) \land (x \neq y) \land \forall z ((Mushroom($

vii. Clinton is not tall.

- **Answer:** $\neg Tall(Clinton)$

viii. All men are not fool.

- **Answer:** $\exists x(Man(x) \wedge \neg Fool(x))$

  > **Note:** This translation assumes the sentence means "Not all men are fools." The English is slightly ambiguous. If it were to mean "No man is a fool," the translation would be $\forall x(Man(x) \rightarrow \neg Fool(x))$.

Set - II

i. Emily is either a surgeon or a lawyer.

- **Answer:** `Surgeon(Emily) v Lawyer(Emily)`

ii. Joe is an actor, but he also holds another job.

- **Answer:** `Actor(Joe) ∧ HoldsAnotherJob(Joe)`

  > **Note:** "But" is translated as a logical AND (∧). The concept of "another job" is represented with a dedicated predicate for simplicity.

iii. All surgeons are doctors.

- **Answer:** $\forall x(Surgeon(x) \rightarrow Doctor(x))$

iv. Joe does not have a lawyer (i.e., is not a customer of any lawyer).

- **Answer:** $\neg \exists x(Lawyer(x) \wedge CustomerOf(Joe, x))$
  - This can also be written equivalently as: $\forall x(Lawyer(x) \rightarrow \neg CustomerOf(Joe, x))$

v. Emily has a boss who is a lawyer.

- **Answer:** $\exists x(BossOf(x, Emily) \wedge Lawyer(x))$

vi. There exists a lawyer all of whose customers are doctors.

- **Answer:** $\exists x(Lawyer(x) \wedge \forall y(CustomerOf(y, x) \rightarrow Doctor(y)))$

vii. Every surgeon has a lawyer.

- **Answer:** $\forall x(Surgeon(x) \rightarrow \exists y(Lawyer(y) \wedge CustomerOf(x, y)))$