Republic of the Philippines
Laguna State Polytechnic University
Main Campus, Sta. Cruz, Laguna

ADVANCED PRESENTATION AND REASONING
Machine Problem 1

Name: Ramos, Jezreel R
Year/Section: BSCS3A **Instructor: Mark Bernardino**

Part A — Conceptual Questions

1. What is Unification and why is it important in predicate-logic inference?

Unification is the process of finding a substitution (a mapping from variables to terms) that makes two predicate expressions syntactically identical.
Example substitution: {x → Socrates} unifies Human(x) with Human(Socrates).

Why it matters: Unification is the mechanism that lets inference rules (like Generalized Modus Ponens and resolution) match the antecedents of rules with known facts. Without unification you couldn't apply general rules to specific instances or propagate variables correctly during automated reasoning.

2. Forward chaining vs Backward chaining

- **Forward Chaining (data-driven):** Start from known facts and repeatedly apply inference rules to derive new facts until the goal is reached (or no new facts). Good when you have lots of facts and want all consequences (e.g., real-time monitoring systems, production rule systems, sensor-data interpretation).
- **Backward Chaining (goal-driven):** Start from a goal and work backwards: find rules that could produce the goal and try to establish their premises (subgoals). Good for query answering where you only want to prove a particular goal (e.g., Prolog query resolution, expert-system question answering).

3. What is Generalized Modus Ponens (GMP)? Give an example.

GMP extends propositional modus ponens to first-order logic using unification. If you have a universally quantified implication ∀x (P(x) ⇒ Q(x)) and a fact P(a), then by unifying P(x) with P(a) using {x→a} you can infer Q(a).

Example:

- Rule: ∀x (Human(x) ⇒ Mortal(x))
- Fact: Human(Socrates)
- Unification {x → Socrates} yields Mortal(Socrates).

4. What is Resolution and why is it powerful?

Resolution is a single inference rule used for automated theorem proving. It operates on clauses in Conjunctive Normal Form (CNF): given two clauses that contain complementary literals L and ¬L (after suitable unification), you can infer a new clause that is the disjunction of the remaining literals. Resolution + refutation works by adding the negation of the goal and deriving the empty clause (contradiction).

Why it's powerful:

- It is (refutation-)complete for first-order logic: if a set of sentences logically implies a goal, resolution (with proper transformations like skolemization and unification) will eventually derive a contradiction.
- Uses a single mechanical rule (plus unification), which makes it easy to implement and automate.
- Works uniformly on many problems by reduction to clause manipulation.

Part B — Translation & Reasoning (proofs)

For each item I'll: (a) give a predicate translation, (b) convert to clauses (CNF/skolemize where needed), (c) show a resolution or GMP-style proof.

B.1

Natural-language premises:

1. "All humans are mortal."
2. "Socrates is a human."
   Goal: Socrates is mortal.

Translation (predicates):

- Human(x) — x is a human.
- Mortal(x) — x is mortal.
  Premises:

1. ∀x (Human(x) ⇒ Mortal(x))
2. Human(Socrates)

Proof (GMP / simple instantiation):

- From (1) and universal instantiation: take x = Socrates → Human(Socrates) ⇒ Mortal(Socrates).
- From (2) Human(Socrates).
- By Modus Ponens (or GMP with substitution {x→Socrates}) infer Mortal(Socrates). ✅

(If using resolution: convert (1) to clause ¬Human(x) ∨ Mortal(x), assert Human(Socrates). Resolve Human(Socrates) with ¬Human(Socrates) ∨ Mortal(Socrates) → Mortal(Socrates).)

B.2

Natural-language premises:

1. "Every student who studies passes the exam."
2. "Juan is a student and he studies."
   Goal: Juan passes the exam.

Translation:

- Student(x) — x is a student.
- Studies(x) — x studies.
- Passes(x) — x passes the exam.

Premise 1: ∀x ((Student(x) ∧ Studies(x)) ⇒ Passes(x))
Premise 2: Student(Juan) ∧ Studies(Juan)

Proof (GMP / direct):

- From (1) by universal instantiation: (Student(Juan) ∧ Studies(Juan)) ⇒ Passes(Juan).
- From (2) we have Student(Juan) and Studies(Juan), so the antecedent Student(Juan) ∧ Studies(Juan) is true.
- By Modus Ponens infer Passes(Juan). ✅

(Resolution version: clause form: ¬Student(x) ∨ ¬Studies(x) ∨ Passes(x) plus facts Student(Juan) and Studies(Juan). Resolve Student(Juan) with ¬Student(Juan) ∨

$\neg$Studies(Juan) $\lor$ Passes(Juan) $\rightarrow$ $\neg$Studies(Juan) $\lor$ Passes(Juan). Resolve Studies(Juan) $\rightarrow$ Passes(Juan).)

B.3

Natural-language premises:

1. "If a person is a teacher, then they advise some students."
2. "Mark is a teacher."
   Goal: Mark advises at least one student — i.e. $\exists y$ (Student(y) $\land$ Advises(Mark, y)).

Translation:

- Teacher(x) — x is a teacher.
- Student(y) — y is a student.
- Advises(x,y) — x advises y.

Premise 1: $\forall x$ (Teacher(x) $\Rightarrow$ $\exists y$ (Student(y) $\land$ Advises(x,y)))
Premise 2: Teacher(Mark)
Goal: $\exists y$ (Student(y) $\land$ Advises(Mark,y))

Reasoning (using skolemization + resolution / constructive instantiation):

A direct constructive way (informal / GMP view): apply the rule to x = Mark. The implication says that for Mark there exists at least one student he advises. So conclude $\exists y$ (Student(y) $\land$ Advises(Mark,y)). That already proves the goal.

To show it in the standard resolution style (with skolemization):

1. Start with $\forall x$ (Teacher(x) $\Rightarrow$ $\exists y$ (Student(y) $\land$ Advises(x,y))).
   - Convert implication: $\forall x$ ($\neg$Teacher(x) $\lor$ $\exists y$ (Student(y) $\land$ Advises(x,y))).
   - Skolemize the existential y that depends on x by introducing a Skolem function f(x). After Skolemization we get a universal clause with no existential: $\forall x$ ($\neg$Teacher(x) $\lor$ Student(f(x))) and $\forall x$ ($\neg$Teacher(x) $\lor$ Advises(x, f(x))). (Equivalently combine as two clauses.)
   - Clause set from (1):
     (A) $\neg$Teacher(x) $\lor$ Student(f(x))
     (B) $\neg$Teacher(x) $\lor$ Advises(x, f(x))

2. From premise (2): Teacher(Mark) → clause (C) Teacher(Mark).
3. Instantiate (A) and (B) with x = Mark (i.e., apply substitution {x → Mark}):
   - From (A): ¬Teacher(Mark) ∨ Student(f(Mark)). Resolve with (C) Teacher(Mark) → eliminates ¬Teacher(Mark) and yields Student(f(Mark)).
   - From (B): ¬Teacher(Mark) ∨ Advises(Mark, f(Mark)). Resolve with (C) → yields Advises(Mark, f(Mark)).
4. From steps above we have both Student(f(Mark)) and Advises(Mark, f(Mark)). Combine: Student(f(Mark)) ∧ Advises(Mark, f(Mark)). That shows a concrete witness f(Mark) exists for the existential ∃y (Student(y) ∧ Advises(Mark,y)). ✅

So resolution/Skolemization produces a Skolem constant/function that witnesses the existential — proving that Mark advises at least one student.