  
  
  
  
  
  
  
  
  
  
  
  
  
**Part A. Conceptual Questions (Short Answer) (20 points)**

**5**

**Predicate Logic**

**Name: Shaila Patrice D. Avellaneda**

**Section BSCS 3A**

1. Define Unification and explain why it is important in predicate logic inference.

* Unification is the process of finding a set of substitutions that make two logical expressions the same. It allows inference rules—such as Generalized Modus Ponens, resolution, and forward or backward chaining—to connect general rules with specific facts or goals by replacing variables with appropriate terms. In simpler terms, unification helps us apply general logical rules to particular situations so that reasoning can take place.

1. Differentiate between Forward Chaining and Backward Chaining. Provide one practical application of each.

* **Forward Chaining** is a method starts with the information already known and applies rules whose conditions are fulfilled to create new conclusions, continuing until the target goal is achieved.For example, it can be applied in smart home systems that automatically adjust lighting or temperature based on sensor readings and user preferences. **Backward Chaining** is an approach begins with a specific goal and works backward by identifying which rules could lead to that goal, then checking whether their requirements can be proven true. For an instance, It can be used in medical diagnostic systems that start with a possible illness and trace backward through symptoms and test results to confirm or rule it out.

1. What is Generalized Modus Ponens (GMP)? Give an example in predicate logic.

* GMP generalizes classical Modus Ponens to predicates with variables. If you have a rule ∀x (P(x) ∧ Q(x) → R(x)) and facts P(a) and Q(a), GMP allows you to infer R(a) after applying the substitution {x/a}.

**Concrete example:**

* + Rule: ∀x (Dog(x) → Animal(x))
  + Fact: Dog(Fido)  
    Applying GMP with substitution {x/Fido} gives Animal(Fido).

1. Explain in your own words what Resolution is and why it is powerful in automated theorem proving.

* Resolution is a rule of inference used in logic that works by combining two statements with opposing parts (like P and ¬P) to form a new statement, and repeating this process can lead to a contradiction. It is powerful because it is both sound and complete—meaning it can reliably find contradictions when a conclusion logically follows—making it a key method used in automated reasoning and theorem-proving systems.

**Part B. Translation & Reasoning**

1. “All humans are mortal.”

“Socrates is a human”

→ Prove that Socrates is mortal.

**Predicate Logic Translation:**

* ∀x (Human(x) → Mortal(x))
* Human(Socrates)

**Reasoning**

* From (1), by substituting x = Socrates: Human(Socrates) → Mortal(Socrates)
* Given (2), Human(Socrates) is true.
* By Modus Ponens, **Mortal (Socrates)** is true.

1. “Every student who studies passes the exam.”

“Juan is a student and he studies”

→ Prove that Juan passes the exam.

**Predicate Logic Translation:**

* ∀x ((Student(x) ∧ Studies(x)) → PassesExam(x))
* Student(Juan)
* Studies(Juan)

**Reasoning**

* From (1), substitute x = Juan: (Student(Juan) ∧ Studies(Juan)) → PassesExam(Juan)
* From (2) and (3), both conditions are true.
* By Modus Ponens, **PassesExam(Juan)** is true.

1. “If a person is a teacher, then they advise some students.”

“Mark is a teacher.”  
→ Prove that Mark advises at least one student.

**Predicate Logic Translation:**

* ∀x (Teacher(x) → ∃y (Student(y) ∧ Advises(x, y)))
* Teacher(Mark)

**Reasoning**

* From (1), substitute x = Mark: Teacher(Mark) → ∃y (Student(y) ∧ Advises(Mark, y))
* Given (2), Teacher(Mark) is true.
* By Modus Ponens, **∃y (Student(y) ∧ Advises(Mark, y))** is true — meaning **Mark advises at least one student.**