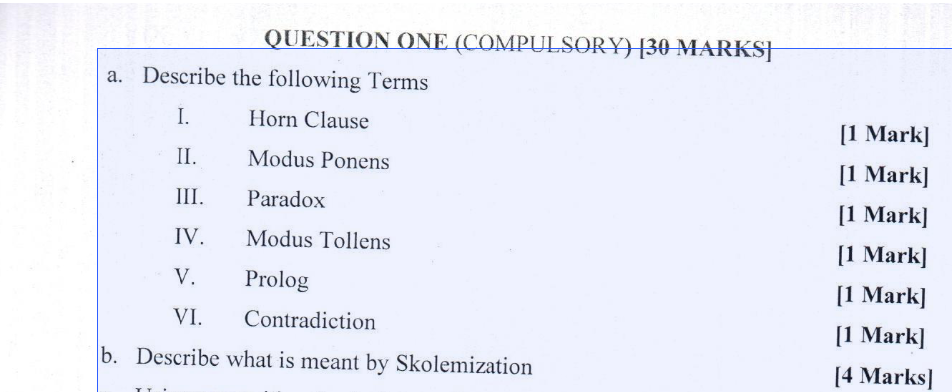
CSC227 Logic Programming  
8th Oct ‘21

Question One



I. **Horn Clause:** A logical formula consisting of a disjunction of literals, with at most one positive literal. It's named after the logician Alfred Horn and commonly used in logic programming and automated theorem proving.

II. **Modus Ponens:** A rule of inference in propositional logic where from a conditional statement ("if P then Q") and the affirmation of the antecedent ("P"), one can infer the consequent ("Q").

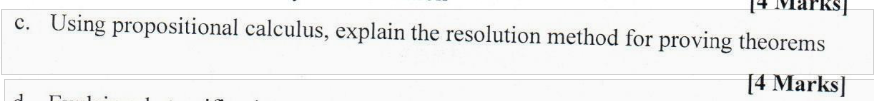
III. **Paradox**: A statement or situation that leads to a contradictory or logically unacceptable conclusion, challenging common sense or intuition.

IV. **Modus Tollens** :A valid form of argumentation where from a conditional statement ("if P then Q") and the negation of the consequent ("not Q"), one can infer the negation of the antecedent ("not P").

V. **Prolog**: A logic programming language used for symbolic and non-numeric computations, particularly in artificial intelligence and computational linguistics. It operates based on formal logic and allows declarative programming.

VI. **Contradiction**: A situation where two or more propositions, statements, or conclusions are incompatible or mutually exclusive, leading to logical inconsistency.

**Skolemization:** Skolemization is a technique used in mathematical logic and automated theorem proving to eliminate existential quantifiers from a logical formula. It replaces existential quantifiers with Skolem functions or Skolem constants, thus converting a formula into an equivalent one in prenex normal form. This process facilitates reasoning and automated deduction by removing existential dependencies.



The resolution method is a fundamental technique in propositional calculus used for proving theorems. It is particularly effective for proving statements in first-order logic. Here's a concise explanation of the resolution method:

1. Statement Representation:

- Express the given statements or theorems in propositional logic, where propositions are represented by variables, and logical connectives (AND, OR, NOT) are used.

2. Conversion to CNF (Conjunctive Normal Form):

- Convert the propositional logic statements into conjunctive normal form (CNF), which is a conjunction of disjunctions of literals. This involves applying various logical equivalences and transformations.

3. Clause Set Formation:

- Represent each clause (disjunction of literals) as a separate element in a set, forming the clause set.

4. Resolution Rule:

- Apply the resolution rule iteratively until either a contradiction (empty clause) is derived or no further resolutions are possible.

- The resolution rule states that if there are two clauses with complementary literals (one clause contains a literal, and the other its negation), then their resolution results in a new clause formed by the union of the remaining literals from both clauses, excluding the complementary literals.

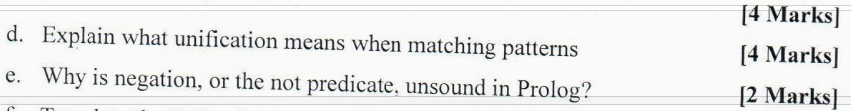
5. Contradiction Check:

- Check if the empty clause is derived. If it is, then the original set of clauses is inconsistent, and the theorem is proven.

6. Termination:

- If no contradiction is reached and no further resolutions are possible, conclude that the original set of clauses is consistent, and the theorem is not proven.

The resolution method is sound and complete for propositional logic, meaning that if a contradiction exists, the method will eventually find it, and if there is no contradiction, the method will correctly conclude consistency. It serves as a foundation for more complex automated reasoning systems, including those applied in artificial intelligence and theorem proving.



d. Unification in matching patterns refers to the process of finding a substitution that makes two patterns identical. In Prolog, unification is a fundamental operation used to match queries with clauses in the knowledge base. It involves recursively matching corresponding terms, variables, and structures until a consistent assignment is found. If successful, unification produces a substitution that allows the query to be satisfied by binding variables to appropriate values.

e. Negation, or the not predicate, is unsound in Prolog because it violates the closed-world assumption. Prolog operates under the closed-world assumption, meaning that anything not known to be true is considered false. However, negation as failure (the not predicate) in Prolog doesn't fit neatly into this assumption. It doesn't necessarily imply falsity but rather the absence of evidence for a proposition. Therefore, using negation in Prolog can lead to unintended interpretations and contradictions, making reasoning unsound.

