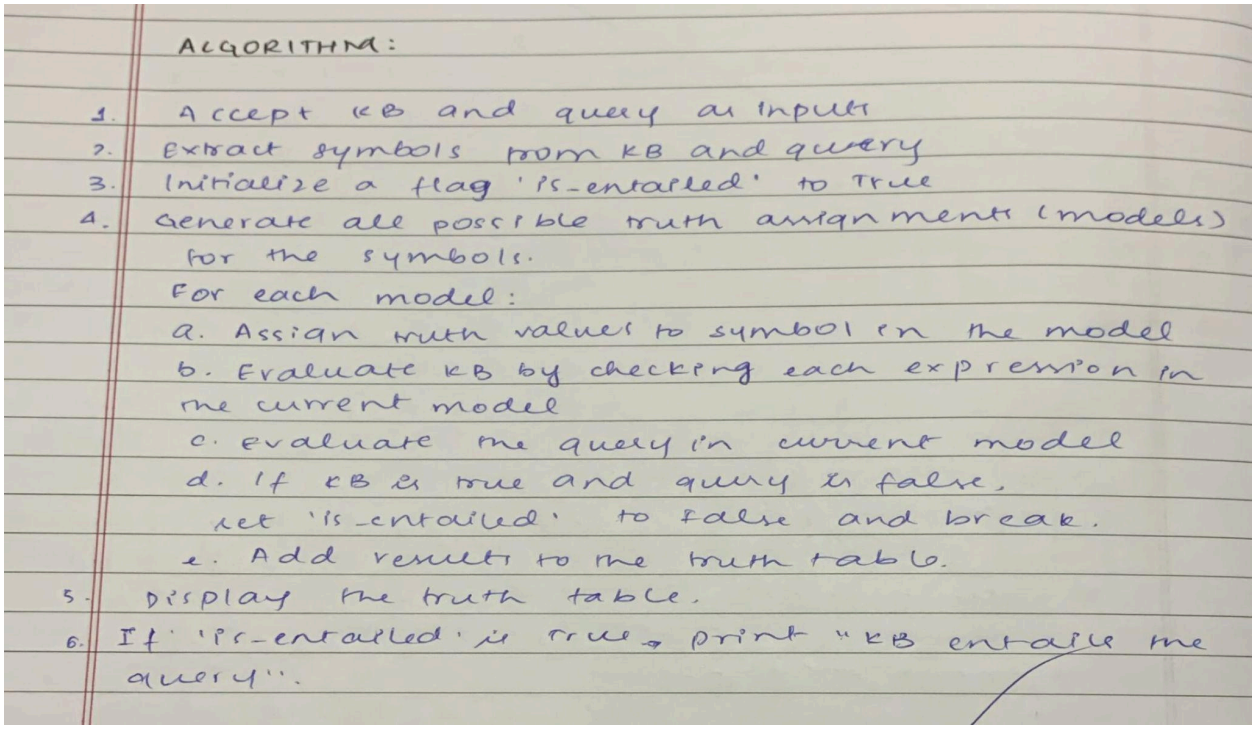
12-11-2024 WEEK - 07

Create a knowledge base using propositional logic and show that the query entails the knowledge base or not.

ALGORITHM / PSEUDOCODE -



ALGORITHM:

1. Accept KB and query as inputs
2. Extract symbols from KB and query
3. Initialize a flag 'is_entailed' to True
4. Generate all possible truth assignments (models) for the symbols.
   For each model:
   a. Assign truth values to symbol in the model
   b. Evaluate KB by checking each expression in the current model
   c. Evaluate the query in current model
   d. If KB is true and query is false, set 'is_entailed' to false and break.
   e. Add results to the truth table.
5. Display the truth table.
6. If 'is_entailed' is true, print "KB entails the query".

CODE -

```
import itertools

def display_truth_table(KB, query, symbols):
    """
    Display the truth table for the KB and query, and determine if the query is entailed by the KB.
    """
    # Generate all possible models (combinations of truth values)
    all_models = list(itertools.product([True, False], repeat=len(symbols)))
    is_entailed = True  # Assume entailment until proven otherwise
    table = []  # For storing table rows

    print("Truth Table:")
```

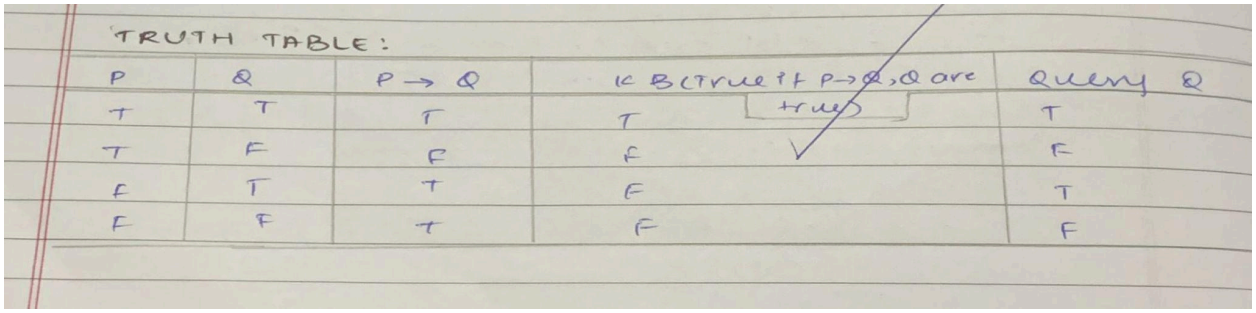
```
    print(" | ".join(symbols) + " | KB | Query | Entailment")
    print("-" * (len(symbols) * 4 + 26))

    # Iterate over all possible models
    for model in all_models:
        # Assign truth values to symbols in the model
        model_dict = dict(zip(symbols, model))

        # Evaluate the KB and query in the current model
        KB_true = all(eval(expr, {}, model_dict) for expr in KB)
        query_true = eval(query, {}, model_dict)
        entailment = "Yes" if KB_true and query_true else "No"

        # Print the row
        row = [model_dict[symbol] for symbol in symbols] + [KB_true, query_true,
entailment]
        table.append(row)
        print(" | ".join(str(val) for val in row))

        # Check for entailment
        if KB_true and not query_true:
            is_entailed = False

    print("\nDoes KB entail the query?", "Yes" if is_entailed else "No")
    return is_entailed

# Accept user input for Knowledge Base (KB) and query
print("Enter the propositions in the Knowledge Base (KB), separated by commas
(e.g., '(not P or Q)', 'P'):")
KB_input = input().split(',')

# Trim whitespace from each KB entry
KB = [prop.strip() for prop in KB_input]

print("Enter the query proposition (e.g., 'Q'):")
```

```
query = input().strip()

# Accept symbols dynamically based on KB and query
symbols = sorted(set(sym for expr in KB + [query] for sym in expr if
sym.isalpha()))

# Display truth table and check entailment
display_truth_table(KB, query, symbols)
```

OUTPUT-

```
Enter the propositions in the Knowledge Base (KB), separated by commas (e.g., '(not P or Q)', 'P'):
P , Q
Enter the query proposition (e.g., 'Q'):
P
Truth Table:
P | Q | KB | Query | Entailment
---------------------------------
True | True | True | True | Yes
True | False | False | True | No
False | True | False | False | No
False | False | False | False | No

Does KB entail the query? Yes
True
```

TRUTH TABLE -



TRUTH TABLE:

P	Q	P → Q	KB (True if P→Q, Q are true)	Query Q
T	T	T	T	T
T	F	F	F	F
F	T	T	F	T
F	F	T	F	F