

Date: 19/11/24

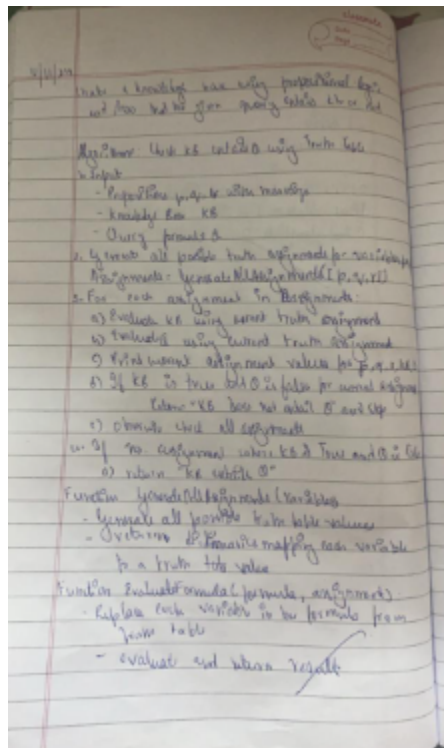
Program Title:

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

i. For given KB and Query you should write truth table values to demonstrate whether the given query entails KB or not.

ii. Implementation of truth-table enumeration algorithm for deciding propositional entailment.

Algorithm:



kb formula = context
 p.d. is raining
 q.d. the ground is wet
 r. I have an umbrella

 kb. (-p) & (-q) & (-r)
 either p is not raining or the ground is not wet or the ground is not wet, or I have an umbrella

 Query (-p & r): either p is not raining, or I have an umbrella

 output
 Print the meaning of the query proposition
 Enter the meaning of proposition p. Is it raining
 Enter the meaning of proposition q. The ground is wet
 Enter the meaning of proposition r. I have an umbrella

 Now, after the knowledge base (kb) and query consulting
 Print proposition
 You can use 'q', 'r', 'not', 'or', 'and',
 and parentheses in your formula

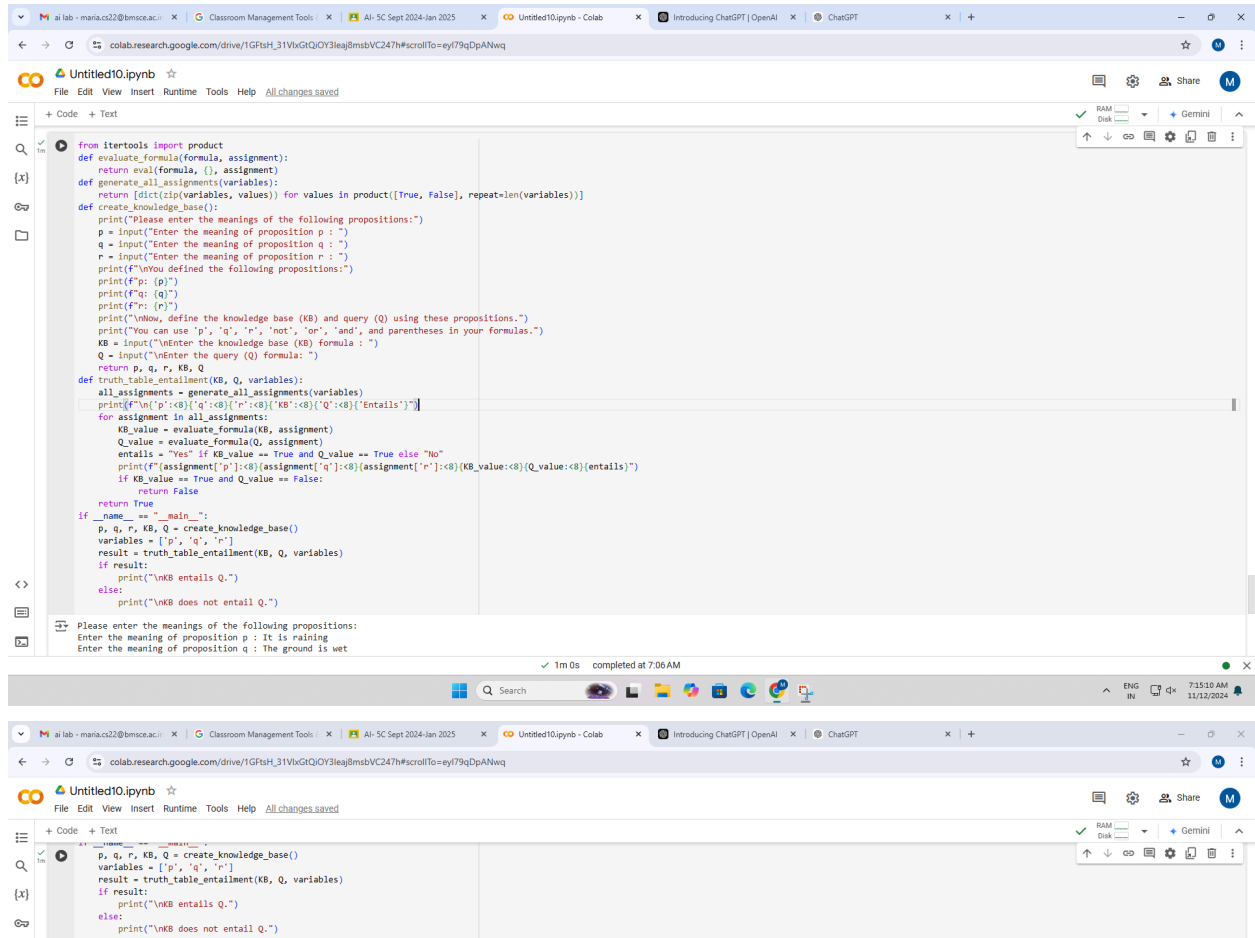
 Enter the knowledge base (kb) formula
 (not p or q) and (not q or r)

 Enter the query (q) formula (not p and r)

p	q	r	kb	0	1	kb
1	1	1	1	1	1	Yes
1	1	0	0	0	0	No
1	0	1	0	1	1	No
1	0	0	0	0	0	No
0	1	1	1	1	1	Yes
0	1	0	0	1	1	No
0	0	1	1	1	1	Yes
0	0	0	1	1	1	Yes

kb entered
 Answer

Code:



```
from itertools import product
def evaluate_formula(formula, assignment):
    return eval(formula, {}, assignment)
def generate_all_assignments(variables):
    return [dict(zip(variables, values)) for values in product([True, False], repeat=len(variables))]
def create_knowledge_base():
    print("Please enter the meanings of the following propositions:")
    p = input("Enter the meaning of proposition p : ")
    q = input("Enter the meaning of proposition q : ")
    r = input("Enter the meaning of proposition r : ")
    print(f"\nYou defined the following propositions:")
    print(f"p: {p}")
    print(f"q: {q}")
    print(f"r: {r}")
    print("\nNow, define the knowledge base (KB) and query (Q) using these propositions.")
    print("You can use 'p', 'q', 'r', 'not', 'or', 'and', and parentheses in your formulas.")
    KB = input("\nEnter the knowledge base (KB) formula : ")
    Q = input("\nEnter the query (Q) formula : ")
    return p, q, r, KB, Q
def truth_table_entailment(KB, Q, variables):
    all_assignments = generate_all_assignments(variables)
    print(f"\n{'p':<8}{'q':<8}{'r':<8}{'KB':<8}{'Q':<8}{'Entails':<8}")
    for assignment in all_assignments:
        KB_value = evaluate_formula(KB, assignment)
        Q_value = evaluate_formula(Q, assignment)
        entails = "Yes" if KB_value == True and Q_value == True else "No"
        print(f"{assignment['p']:<8}{assignment['q']:<8}{assignment['r']:<8}{KB_value:<8}{Q_value:<8}{entails}")
        if KB_value == True and Q_value == False:
            return False
    return True
if __name__ == "__main__":
    p, q, r, KB, Q = create_knowledge_base()
    variables = ['p', 'q', 'r']
    result = truth_table_entailment(KB, Q, variables)
    if result:
        print("\nKB entails Q.")
    else:
        print("\nKB does not entail Q.")
```

1m 0s completed at 7:06 AM

1m 0s completed at 7:06 AM

```
from itertools import product
def evaluate_formula(formula, assignment):
    return eval(formula, {}, assignment)
def generate_all_assignments(variables):
    return [dict(zip(variables, values)) for values in product([True, False], repeat=len(variables))]
def create_knowledge_base():
    print("Please enter the meanings of the following propositions:")
    p = input("Enter the meaning of proposition p : ")
    q = input("Enter the meaning of proposition q : ")
    r = input("Enter the meaning of proposition r : ")
    print(f"\nYou defined the following propositions:")
    print(f"p: {p}")
    print(f"q: {q}")
    print(f"r: {r}")
    print("\nNow, define the knowledge base (KB) and query (Q) using these propositions.")
    print("You can use 'p', 'q', 'r', 'not', 'or', 'and', and parentheses in your formulas.")
    KB = input("\nEnter the knowledge base (KB) formula : ")
    Q = input("\nEnter the query (Q) formula : ")
    return p, q, r, KB, Q
```

```

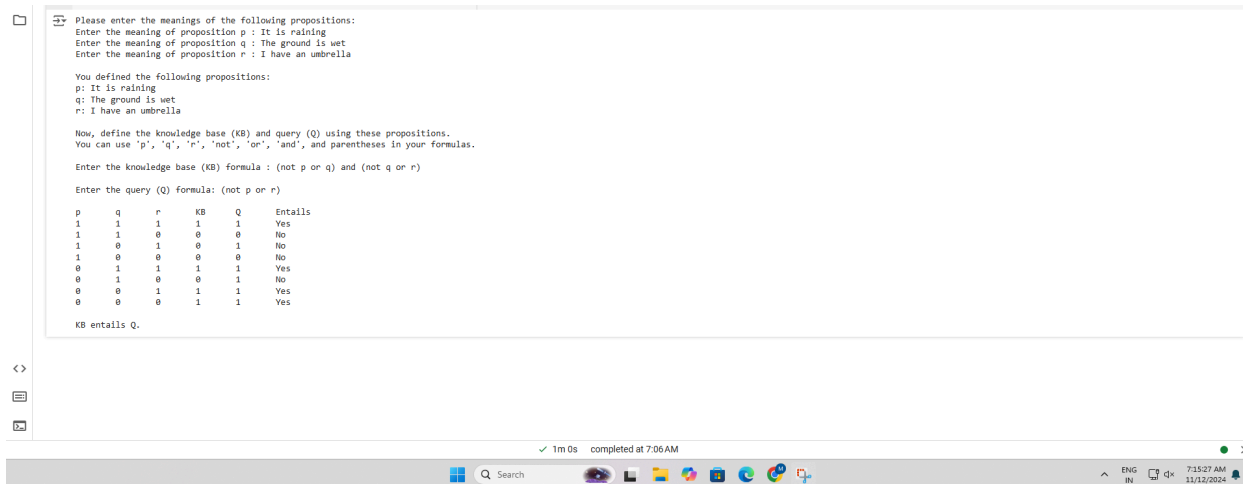
def truth_table_entailment(KB, Q, variables):
    all_assignments = generate_all_assignments(variables)
    print(f'\n{p':<8}{q':<8}{r':<8}{KB':<8}{Q':<8}{Entails}")
    for assignment in all_assignments:
        KB_value = evaluate_formula(KB, assignment)
        Q_value = evaluate_formula(Q, assignment)
        entails = "Yes" if KB_value == True and Q_value == True else "No"

    print(f'{assignment[p]':<8}{assignment[q]':<8}{assignment[r]':<8}{KB_value:<8}{Q_value:<8}{en
tails}")
    if KB_value == True and Q_value == False:
        return False
    return True

if __name__ == "__main__":
    p, q, r, KB, Q = create_knowledge_base()
    variables = ['p', 'q', 'r']
    result = truth_table_entailment(KB, Q, variables)
    if result:
        print("\nKB entails Q.")
    else:
        print("\nKB does not entail Q.")

```

Snapshot of the output:



```

Please enter the meanings of the following propositions:
Enter the meaning of proposition p : It is raining
Enter the meaning of proposition q : The ground is wet
Enter the meaning of proposition r : I have an umbrella

You defined the following propositions:
p: It is raining
q: The ground is wet
r: I have an umbrella

Now, define the knowledge base (KB) and query (Q) using these propositions.
You can use 'p', 'q', 'r', 'not', 'or', 'and', and parentheses in your formulas.

Enter the knowledge base (KB) formula : (not p or q) and (not q or r)
Enter the query (Q) formula: (not p or r)

p    q    r    KB    Q    Entails
1    1    1    1    1    Yes
1    1    0    0    0    No
1    0    1    0    1    No
1    0    0    0    0    No
0    1    1    1    1    Yes
0    1    0    0    1    No
0    0    1    1    1    Yes
0    0    0    1    1    Yes

KB entails Q.

```

1m 0s completed at 7:06 AM

7:13:37 AM 11/12/2024

