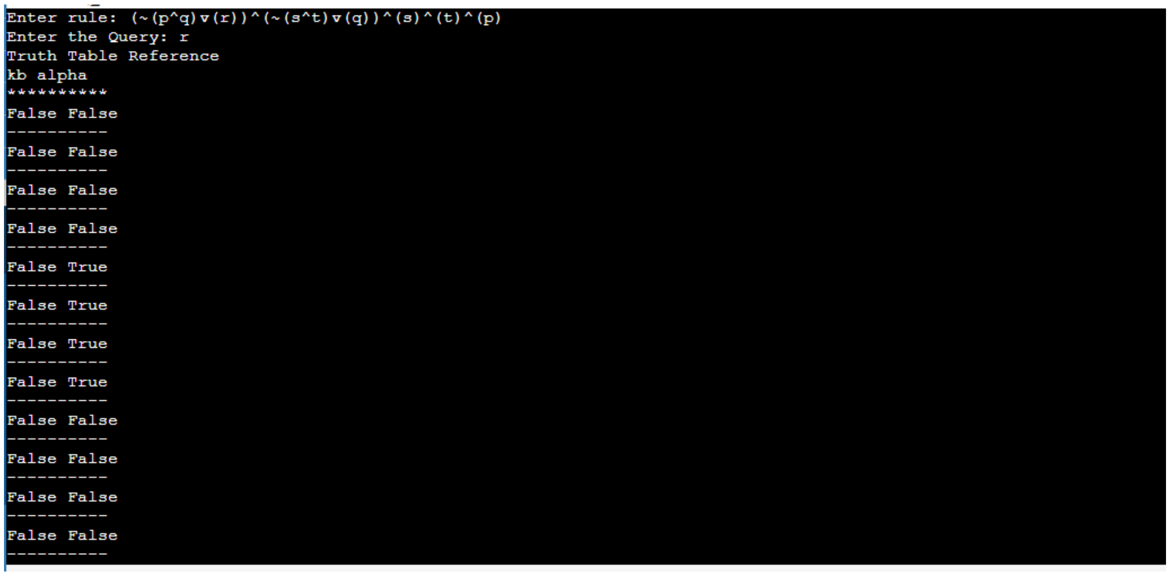
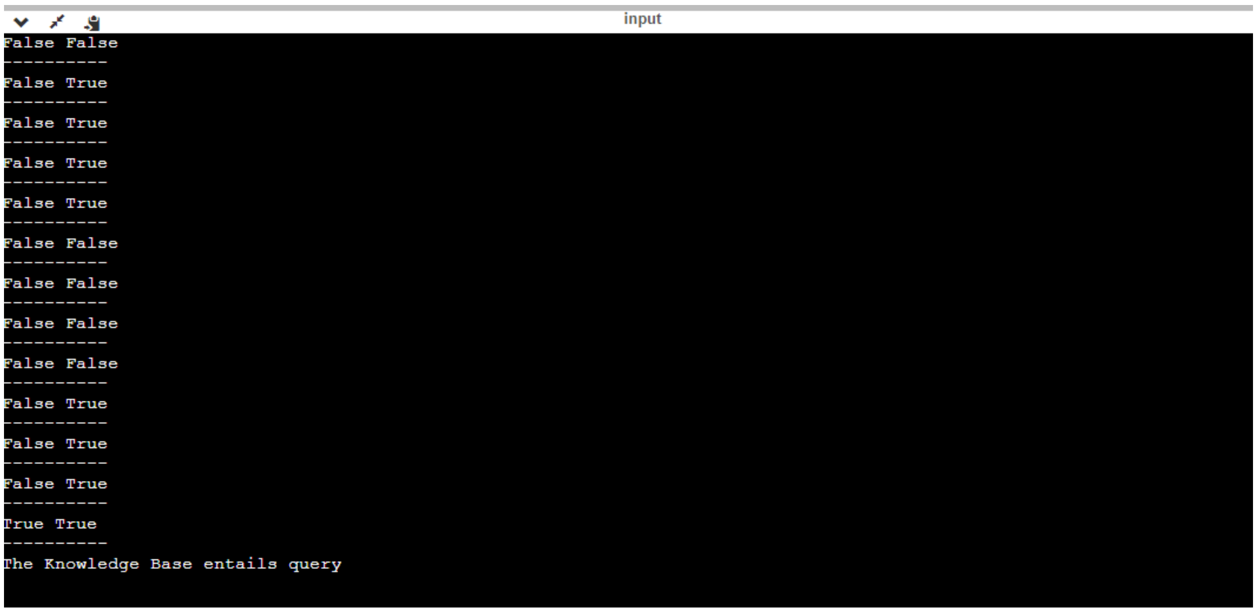
LAKSHMI S KUMAR
1BM19CS078
AI LAB TEST -2

```
combinations=[[False, False, False, False, False], [False, False, False, False, True], [False,
False, False, True, False], [False, False, False, True, True], [False, False, True, False, False],
[False, False, True, False, True], [False, False, True, True, False], [False, False, True, True,
True], [False, True, False, False, False], [False, True, False, False, True], [False, True, False,
True, False], [False, True, False, True, True], [False, True, True, False, False], [False, True,
True, False, True], [False, True, True, True, False], [False, True, True, True, True], [True, False,
False, False, False], [True, False, False, False, True], [True, False, False, True, False], [True,
False, False, True, True], [True, False, True, False, False], [True, False, True, False, True],
[True, False, True, True, False], [True, False, True, True, True], [True, True, False, False, False],
[True, True, False, False, True], [True, True, False, True, False], [True, True, False, True, True],
[True, True, True, False, False], [True, True, True, False, True], [True, True, True, True, False],
[True, True, True, True, True]]
variable={'p':0,'q':1, 'r':2, 's':3, 't':4}
kb=''
q=''
priority={'~':3,'v':1,'^':2}
def input_rules():
    global kb, q
    kb = (input("Enter rule: "))
    q = input("Enter the Query: ")
def entailment():
    global kb, q
    print('*'*10+"Truth Table Reference"+'*'*10)
    print('kb','alpha')
    print('*'*10)
    for comb in combinations:
        s = evaluatePostfix(toPostfix(kb), comb)
        f = evaluatePostfix(toPostfix(q), comb)
        print(s, f)
        print('-'*10)
        if s and not f:
            return False
    return True
def isOperand(c):
    return c.isalpha() and c!='v'
```

```
def isLeftParanthesis(c):
    return c == '('

def isRightParanthesis(c):
    return c == ')'

def isEmpty(stack):
    return len(stack) == 0

def peek(stack):
    return stack[-1]

def hasLessOrEqualPriority(c1, c2):
    try:
        return priority[c1]<=priority[c2]
    except KeyError:
        return False
def toPostfix(infix):
    stack = []
    postfix = ''
    for c in infix:
        if isOperand(c):
            postfix += c
        else:
            if isLeftParanthesis(c):
                stack.append(c)
            elif isRightParanthesis(c):
                operator = stack.pop()
                while not isLeftParanthesis(operator):
                    postfix += operator
                    operator = stack.pop()
            else:
                while (not isEmpty(stack)) and hasLessOrEqualPriority(c, peek(stack)):
                    postfix += stack.pop()
                stack.append(c)
    while (not isEmpty(stack)):
        postfix += stack.pop()

    return postfix
def evaluatePostfix(exp, comb):
    stack = []
    for i in exp:
        if isOperand(i):
            stack.append(comb[variable[i]])
        elif i == '~':
            val1 = stack.pop()
            stack.append(not val1)
        else:
            val1 = stack.pop()
            val2 = stack.pop()
            stack.append(_eval(i,val2,val1))
    return stack.pop()
def _eval(i, val1, val2):
    if i == '^':
        return val2 and val1
    return val2 or val1
#Test 1
input_rules()
ans = entailment()
if ans:
    print("The Knowledge Base entails query")
else:
    print("The Knowledge Base does not entail query")
```



```
Enter rule: (~(p^q)v(r))^(~(s^t)v(q))^(s)^(t)^(p)
Enter the Query: r
Truth Table Reference
kb alpha
**********
False False
----------
False False
----------
False False
----------
False False
----------
False True
----------
False True
----------
False True
----------
False True
----------
False False
----------
False False
----------
False False
----------
False False
----------
```



```
False False
----------
False True
----------
False True
----------
False True
----------
False True
----------
False False
----------
False False
----------
False False
----------
False False
----------
False True
----------
False True
----------
False True
----------
True True
----------
The Knowledge Base entails query
```