2. <u>Artificial Intelligence</u> <u>Shashank Kumar</u>

<u>1BM18CS098</u>

```
def processRule(rule):
        rule = rule.replace('~', 'not')
        rule = rule.replace('^', 'and')
        rule = rule ~~f~~.replace('V', 'or')
def formatRule(rule, P, Q, R):
        P, Q, R = str(P), str(Q), str(R)
    rule = rule.replace('P', P)
    rule = rule.replace('Q', Q)
    rule = rule.replace('R', R)
    return rule
def checkEntailment(rule, query):
    models = [ (False, False, False),
               (False, False, True),
               (False, True, False),
               (False, True, True),
               (True, False, False),
               (True, False, ~~False~~ True),
               (True, True, False)
               (True, True, True)
             ]
             rule = processRule(rule)
           entails = True
```

```
for P,Q,R in models:
    formattedRule = formatRule(rule,P,Q,R)
    print(f'Evaluating: {formattedRule}')
    KB= eval(formattedRule})
    ~~ks~~ -query=R if query=='R' else P if
            query == 'P' else Q
     print(f'Knowledge Base: {KB}
                         Query: {_query}')

    if KB:
       entails ~~x~~ =KB and _query
  if entails:
       print('Knowledge Base entails the query')
  else
       print(' Knowledge Base doesn't
             entail the query')
 rule, query='(RV~P)V(RV~Q)^(~RVP)^
              (~PVQ)','R'

 checkEntailment(rule, query)
```