AI - Lab Test 2
01.01.2021

Shaan Subbaiah B C
1BM18CS096

Question 2)

```
import re

~~define~~
def negate(term):
    return f"~{term}" if term[0] != "~" else term[1]


def reverse(clause):
    if len(clause) > 2:
        t = split_terms(clause)
        return f"{t[1]}v{t[0]}"
    return ""

def split-terms(rule):
    exp = "(~*[PQRS])"
    terms = re.findall(exp, rule)
    return terms

def contradiction(query, clause):
    contradictions = [f"{query}v{negate(query)}",
                      f"{negate(query)}v{query}"]

    return clause in contradictions or reverse(clause)
        in contradictions

def resolve(Kb, query):
    temp = Kb.copy()
    temp += [negate(query)]
```

[illegible]

```
steps = dict()
for rule in temp:
    steps [rule] = "Given"
steps[negate(query)] = "Negated conclusion"

i=0
while i < len(temp):
    n = len(temp)
    j = (i+1) % n
    clauses = []
    while j != i :
        terms1 = split_terms ( temp [i])
        terms 2 = split_terms ( temp [j])
        for c in terms_1 :
            if negate(c) in terms2 :
                t1 = [t for t in terms 1 if t != c]
                t2 = [t for t in terms 2 if t != negate(c)]
                gen = t1 + t2
                if len(gen) == 1 :
                    clauses += [f "{gen[0]}"]
                elif len(gen) == 2 :
                    if gen[0] != negate(gen[1]) :
                        clauses +=
                            [f" {gen[0]}v{gen[1]}"]

                else :

                    if contradiction (query, f"{gen[0]}
                    v {gen[1]}"):
                    temp. append (f" {gen[0]}v
                                 {gen[1]})
```

[illegible]

```
                    steps[""] = f"Resolved {temp[i]} and
                    {temp[j]} to {temp[-1]} which is
                      null.
                    return steps

            elif len(gen) == 1 :
                    clauses += [f"{gen[0]}"]

            else :
                if contradiction (query, f"{terms 1[0]}v
                        {terms2[0]}") :
                    temp.append ( f"terms 1[0] v {terms2[0]}")
                    steps[" "] = f "Resolved {temp[i]}
                    and {temp[j]} to {temp[-1]} which
                    is null
                    return steps

        ~~foa~~ for clause in clauses :
            if ( clause  not  in  temp
                and  clause  != reverse (clause)
                and  reverse (clause)  not  in  temp ) :
                    temp.append (clause)
                    steps[clause] = f"Resolved from
                      {temp[i]} and {temp[j]"

            j = ( j + 1 ) % n

        i += 1
return steps

def  resolution ( kb, query ):
    kb  =  kb . split ("  ")
    steps   =   resolve ( kb , query )
    print (" Step  | Clause | Derivation ")
```

[illegible]

```
    print ("-" * 30)
    i = 1
     for step in steps :
         print (f "{i}  | {step} | steps[{step}] ")
         i+=1

def main :
    print ("Enter the kb : ")
    kb = input()
    print ("Enter the query: ")
    query = input ()
    resolution (kb, query )
```

[illegible]