Date - 1/01/2021

AI LAB-TEST-2

Name - Parag Gattani
USN - 1BM18CS067

Prgm 2

```
import re

def main(rules, goal):
    rules = rules.split('')
    steps = resolve(rules, goal)
    print('\n Step \t | Clause \t | Derivation \t')
    print('-' * 30)
    i = 1
    for step in steps:
        print(f' {i}. \t | {step} \t | {steps[step]} \t')
        i += 1

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 = '(~*[ABCD])'
    terms = re.findall(exp, rule)
    return terms
split_terms('~A v C')

def contradiction(goal, clause):
    contradictions = [f'{goal} v {negate(goal)}',
    f'{negate(goal)} v {goal}']
    return clause in contradictions or reverse(clause) in contradictions
```


Cpanay

Date - 01/01/2024

Name - Parag Gattani
USN - 1BM18CS067

```
def resolve(rules, goal)
    temp = rules.copy()
    temp += [negate(goal)]
    steps = dict()
    for rule in temp:
        steps[rule] = 'Given.'
    steps[negate(goal)] = 'Negated conclusion.'
    i = 0
    while i < len(temp):
        n = len(temp).
        j = (i+1) % n
        clauses = []
        while j != i:
            terms1 = split_terms(temp[i])
            terms2 = split_terms(temp[j])
            for c in terms1:
                if negate(c) in terms2:
                    t1 = [t for t in terms1 if t != c]
                    t2 = [t for t in terms2 if t != negate(c)]
                    gen = t1 + t2
                    if len(gen) == 2:
                        if gen[0] != negate(gen[1]):
                            clauses += [f'{gen[0]} v {gen[1]}']
                        else:
                            if contradiction(goal, f'{gen[0]} v {gen[1]}'):
                                temp.append(f'{gen[0]} v {gen[1]}')
                    steps[""] = f"Resolved {temp[i]} and {temp[j]} to {tem[-1]}, which in turn null.
```


Parag

Date - 01/01/2021

Name - Parag Gattani
USN - 1BM18CS067

In A contradiction is found when {negate(goal)} is assumed as true.
Hence, {goal} is true".

```
        return steps
    elif len(gen) == 1:
        clauses += ['{' gen[0] '}']

    else:
        if contradiction(goal, '{terms1[0]} v {terms2[0]}'):
            temp.append('{terms1[0]} v {terms2[0]}')
            steps = {"] = " Resolved {temp[i]} and
                {temp[j]} to temp[-1]}, which is
            return null

        return steps
    for clause in clauses:
        if clause not in temp and clause != reverse(clause)
        and reverse(clause) not in temp:
            temp.append(clause)
            steps[clause] = 'Resolved from {temp[i]} and {temp[j]}'
                j = (j+1) % n
                i += 1
            return steps.

rules = 'A => B  C => D'
goal = 'A v C => B v D'
main(rules, 'A v C => B v D')
```


Sign