Program 2

```python
def main():
    print("Enter kb:")
    kb = input()
    print("Enter the query:")
    query = input()
    kb = kb.split(',')

    steps = (resolve kb, query)

    print(' Steps\t|clause\t|Derivation\t')

    print('_' * 30)

    i = 1
    for step in steps:
        print(f' {i}.\t| {step[0]}\t|
                    {steps[step][0]}\t')

        i += 1
```

Program-2

```
def resolve ( rules, goal):
    temp = rules. copy()
    temp += [negative(goal)]
    arpn = dict()
    for rule in temp:
        arpn [rule] = 'Given.'
    arpn [negative (goal)] = "Negative conclusion"
    i = 0
    while i < len (temp):
        n= len (temp)
        j = (i+1) % n
        clauses = []
        while j != i:
            term1 = split_terms ( temp [i])
            term2 = split_terms ( temp [j])
            for c in term1:
                if negative (c) in term2:
                    t1 = [t for t in term1 if t !=
                    t2 = [t for t in term2 if t !=
                    gen = t1 + t2
```

Samir Kumar
LBNI8CS091

```
if len (gen) == 2:
    if gen[0] != negative( gen[1]):
        clauses += [f'{gen[0]}v{gen[1]}']

    new = {f : f 'resolved' { temp[0] and
           temp[-1]}  what
           in ind[x in]

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

else:
    if contradiction ( goal, f '{temp1[0]}
                       v {temp2[0]}');


for clauses in clauses:
    if clause not in map and clauses =
            generate ((clause) and
            generate ((clause) not
            in map.
    temp. append

    j = ( j+1 )   n
    i+ = 1
return answer
```