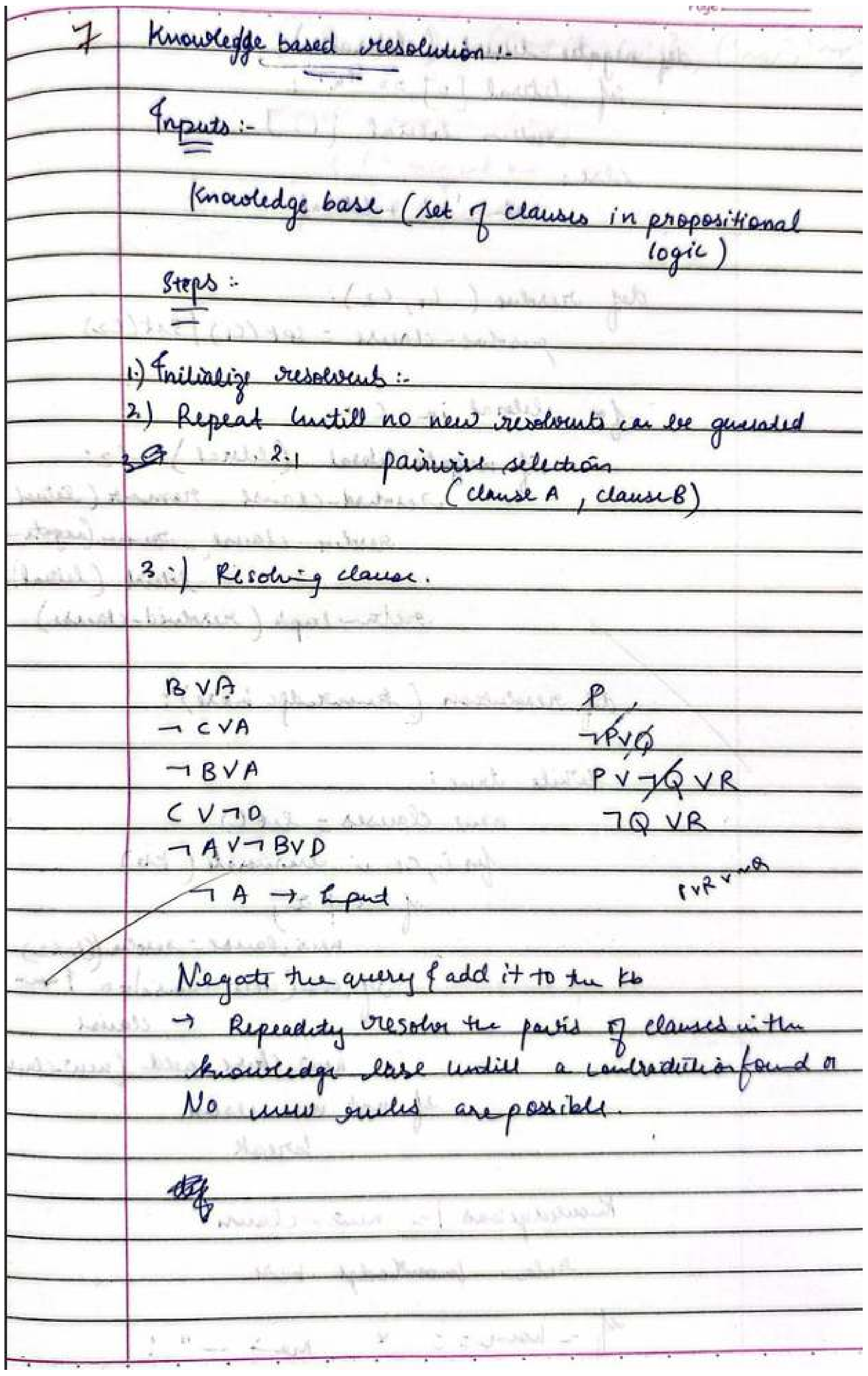
Knowledge-based resolution
Code:-



7 Knowledge based resolution :-

Inputs :-

Knowledge base (set of clauses in propositional logic)

Steps :-

1.) Initialize resolvents :-
2.) Repeat untill no new resolvents can be generated
3 2.1 pairwise selection
(clause A , clause B)

3.) Resolving clause.

$B \vee A$	P
$\neg C \vee A$	$\neg P \vee \cancel{Q}$
$\neg B \vee A$	$P \vee \cancel{\neg Q} \vee R$
$C \vee \neg D$	$\neg Q \vee R$
$\neg A \vee \neg B \vee D$	
$\neg A \rightarrow$ Input	$P \vee R \vee \neg Q$

Negate the query & add it to the kb
→ Repeadity resolve the pairs of clauses in the knowledge base untill a contradiction found or No new rules are possible.

[illegible]

```
def negate-literal (literal)
    if literal [0] == 'n' :
        return literal [1:]
    else:
        return 'n' + literal


def resolve ( c1, c2):
    resolve-clause = set(c1) | set(c2)

    for literal in c1:
        if negate-literal (literal) in c2:
            resolved-clause - remove (lite
            resolve_clause - remove(negat
                literal (liter
            return tuple ( resolved-clause)


def resolution ( knowledge base):

    While true :
        new clauses = set ()
        for i, c1 in enumerate ( kb)
            if i != j
                new clause - resolve (c1, c2)
            if len (new clause) = 0
                clause
            new-clause.add (newcla
        if not new-clause
            break

    knowledgebase |= new-clause
    return knowledge-base

if __name__ == "__main__" :
```

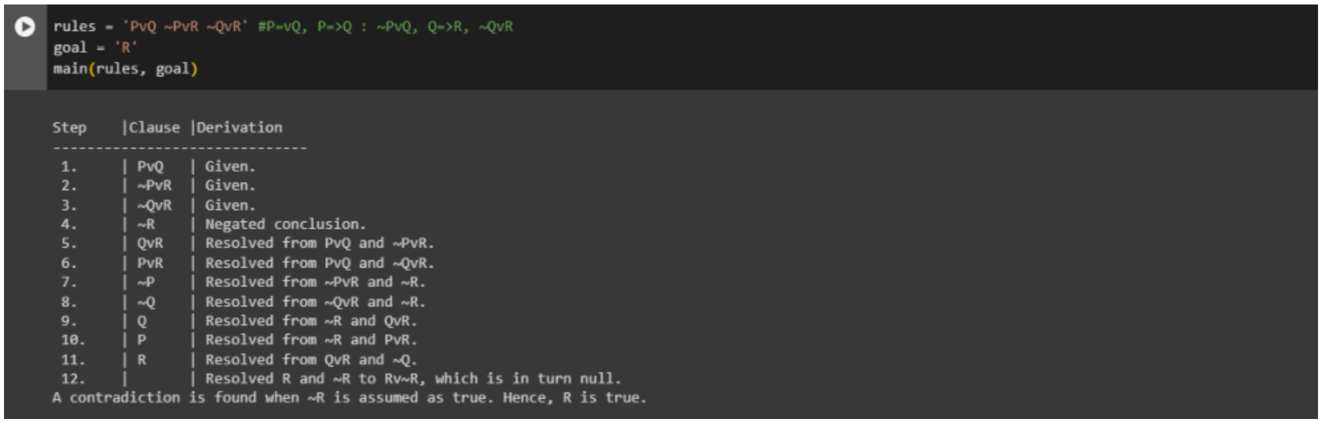
Page

```
kb = {('p', 'q'), ('~p', 'r'}, ('~q', '~r')

    result = resolution(kb)
     print ("original kb", kb)
     print ("Resolved kb", resul)
```

[illegible] 29/12

Output:-



```
rules = 'PvQ ~PvR ~QvR' #P=vQ, P=>Q : ~PvQ, Q=>R, ~QvR
goal = 'R'
main(rules, goal)
```

```
Step    |Clause |Derivation
-----------------------------
 1.     | PvQ   | Given.
 2.     | ~PvR  | Given.
 3.     | ~QvR  | Given.
 4.     | ~R    | Negated conclusion.
 5.     | QvR   | Resolved from PvQ and ~PvR.
 6.     | PvR   | Resolved from PvQ and ~QvR.
 7.     | ~P    | Resolved from ~PvR and ~R.
 8.     | ~Q    | Resolved from ~QvR and ~R.
 9.     | Q     | Resolved from ~R and QvR.
 10.    | P     | Resolved from ~R and PvR.
 11.    | R     | Resolved from QvR and ~Q.
 12.    |       | Resolved R and ~R to Rv~R, which is in turn null.
A contradiction is found when ~R is assumed as true. Hence, R is true.
```