LAB TEST - 2

yuvraj singh
1BM18CS126

Pgm 3

```
kb = []
def CLEAR():
    global kb
    kb = []

def TELL(sentence):
    global kb
    if isClause(sentence):
        kb.append(sentence)
    else:
        sentenceCNF = convertCNF(sentence)
        if not sentenceCNF:
            print("Illegal input")
            return
        if isANDList(sentenceCNF):
            for s in sentenceCNF[1:]:
                kb.append(s)
        else: kb.append(sentenceCNF)

def ASK(sentence):
    global kb
    if isClause(sentence):
        neg = negation(sentence)
    else:
        sentenceCNF = convertCNF(sentence)
        if not sentenceCNF:
            print("Illegal")
            return
        neg = convertCNF(negation(sentenceCNF))
```

```
ask_list = []
if isAndList(neg):
    for n in neg[1:]:
        nCNF = makeCNF(n)
        if type(nCNF).__name__ == 'list':
            ask_list.insert(0, nCNF)
        else:
            ask_list.insert(0, nCNF)
else:
    ask_list = [neg]
clauses = ask_list + kb[:]
while True:
    new_clauses = []
    for c1 in clauses:
        for c2 in clauses:
            if c1 is not in c2:
                resolved = resolve(c1, c2)
                if resolved == False:
                    continue
                if resolved == []:
                    return True
                new_clauses.append(resolved)
    if len(new_clauses) == 0:
        return False
    new_in_clauses = True
    for n in new_clauses:
        if n not in clauses:
            new_in_clauses = False
            clauses.append(n)
```

```
        if new_in_clauses:
            return False
    return ~~return~~ False

def resolve(arg_one, arg_two):
    resolved = False
    s1 = make_sentence(arg_one)
    s2 = make_sentence(arg_two)
    resolve s1 = ~~False~~ None
    resolve s2 = None
    for i in s1:
        if isNotList(i):
            a1 = i[1]
            a1_not = True
        else: a1 = i
              a1_not = False
        j in s2:
            if isNotList(j):
                a2 = j[1]
                a2_not = True
            else - a2 - not = False.
            if a1 == a2:
                if a1_not != a2_not:
                    if resolved:
                        return False
                    else resolved = True
                         resolve_s1 = i
                         resolve_s2 = j
                         break.
            if not resolved: False.

def make_sentence(arg):
    if isLiteral(arg) or
        isNotList(arg):
        return (arg)
    if isOrList(arg):
        return clear_duplicate
               (arg[1:])
    return

# negation
def negation(sentence):
    if isliteral(sentence):
        return ['not', sentence]
    if isNotlist(sentence):
        return sentence[1]
    if isAndlist(sentence):
        result = ['or']
        for i -> sentence[1:]:
            if isNotlist(sen)
                result.append(i[1])
            else: result.append
                  (['not', sen])
        return result
    if isOrlist(sentence):
        result = ['and']
        for i -> sent
            if isNotlist(sen)
                result.append(i)
            else
                result.append
                (['not', i])
```