AI lab
Exam 2

Shreyash A
1BM18CS106
Batch C1

Program 4:

```
def getAttr(string):
    expr = '\([^)]+\)'
    matches = re.findall(expr, string)
    return [m for m in str(matches) if m.isalpha()]

def getpred(string):
    expr = '[a-z~]+\([A-Za-z,]+\)'
    return re.findall(expr, string)

def DeMorgan(sentence):
    string = ''.join(list(sentence).copy())
    string = string.replace('~~', '')
    flag = '[' in string
    string = string.replace('~[', '(')
    flag = '[' in string
    string = string.replace('~[', '(')
    string = string.replace(']', ')')
    for predicate in getPredicates(string):
        string = string.replace(predicate, f'~{predicate}')
    s = list(string)
```

```
def skm(sentence):
    skm_conts = [f'{chr(c)}' for c in range
                 (ord('A'), ord('Z')+1)]
    statement = ''.join(list(sentence).copy())
    matches = re.findall('[∀∃].', statement)
    for match in matches[::-1]:
        statement = statement.replace(s, s[1::-1])
    for predicate in getpred(statement):
        attributed = getAtr(predicate)
        if ''.join(attributes).islower():
            statement = statement.replace(match[1],
                        skm_constants.pop(0))
        else:
            aV = [a for a in attributes if not
                  a.islower()][0]
            statement = statement.replace(aV,
                f'{skm_const.pop(0)}({match[1]})')

return statement
```

```
import re
def fol_to_cnf(fol):
    statement = fol.replace("<=>", "_")
    while '_' in statement:
        i = statement.index('_')
        new_statement = '[' + statement[:i]
            + '=>' + statement[i+1:] + statement[i+1:]
        statement = new_statement
    for i, s in enumerate(statements):
        if '[' in s and ']' not in s:
            statements[i] += ']'

    for s in statements:
        statement = statement.replace(s, fol_to_
                                      cnf(s))
    while '-' in statement:
        i = statement.index('-')
        br = statement.index('[') if '[' in statement
        else 0

    while '-∀' in statement:
        i = statement.index('-')
        br = statement.index('[') if '[' in statement
        else 0
```

```
    Statement = ''.join(statement)
While '~∃' in statement:
    i = statement.index('~∃')
    s = list(statement)
    s[i], s[i+1], s[i+2] = '∃', statement[i+2], '~'
    statement = ''.join(statement)
statement = statement.replace('~[∀', '[~∀')
statement = statement.replace('~[∃', '[~∃')
expr = '(~[∀|∃].)'
statements = re.findall(expr, statement)
for s in statements:
    statement = statement.replace(s, fol_to_cnf(s))
expr = '~\[[^]]+\]'
statements = re.findall(expr, statement)
for s in statements:
    statement = statement.replace(s, DeMorgan(s))
return statement

print(skm(fol_to_cnf("∀x(likes(ram, x) => likes
                        (sita, x))")))
```