Name:- Rushali. R. Shetty.

USN:- 1BM18CS086

Class:- 5 B

Date:- 01/01/2021

AI Lab 2

Prg:- FOL to CNF

```
import re

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

def getPredicates(string):
    expr = '[a-z~]+\([A-Za-z,]+\)'
    return re.findall(expr, string)
#apply demorgan law.
def DeMorgan(sentence):
    string = ''.join(list(sentence).copy())
    string = string.replace('~~', '')
    flag = '[' in string
    string = string.replace('~[', '')
    string = string.strip(']')
    for predicate in getPredicates(string):
        string = string.replace(predicate, f'~{predicate}')

    s = list(string)
```

```
for i, c in enumerate(string):
    if c == 'V':
        s[i] = '∧'

    elif c == '∧'
        s[i] = 'V'

    ~~s~~
  string = ''.join(s)
  string = string.replace('~~', '')
  return f'[{string}]' if flag else string.
# remove '∃'
def Skolemization(sentence):
  SKOLEM_CONSTANS = [f'{chr(c)}' for c in ~~range(~~
                                  range(ord('A'), ord('Z')+1)]
  statement = ''.join(list(sentence).copy())
  matches = re.findall('∀∃.', statement)
  for match in matches[::-1]:
      Statement = statement.replace(match, '')
      Statements = re.findall('\[\[[^]]+\]]', statement)
      for s in statements:
          Statement = statement.replace(s, s[1::-1])

      for predicate in getPredicates(statement):
          attributes = getAttributes(predicate)
          if ''.join(attributes).islower():
              statement = statement.replace(match[1],
                          SKOLEM_CONSTANTS.pop(0))
```

```
else:
    aL = [a for a in attributes if a.islower()]
    aU = [a for a in attribute if not a.islower()][0]
    statement = statement.replace(aU, f'{SKOLEM_CONSTANTS.
                pop(0)}({aL[0] if len(aL) else
                        match[1]})')

return statement.


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[:i] + ']'.

    # a<=>b ≡ a=>b ^ b=>a

    statement = new_statement.
    statement = statement.replace("=>",'-")

    # replace => using :- a=>b = ~a V b .

    # while '~∀' in statement:
        # replace ~∀ with ∃~
    while '~∃' in statement:
        # replace ~∃ with ∀~
```

next skolemization .

next step drop $\forall$

next step apply distribution $\vee$ over $\wedge$.

Steps :- (1) Eliminate implication.

(2) Move $\neg$ inwards.

(3) Standardize variables .

(4) skolemization .

(5) Drop $\forall$

(6) Distribute $\vee$ over $\wedge$.

My question :-

All gardeners like sun

$\forall x$ gardener $(x) \Rightarrow$ likes (x, sun)