Name - Nimit Sajal
USN - 1BM18CS063
Date 1/1/21

AI Lab Test (2)

(4)th Program

English to FOL

english → All gardeners like the sun.

FOL ⇒ forall x, gardener(x) → likes(x, Sun)

```
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~]+\([A-z,]+\)'
    return re.findall(expr, string).

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).
```

(P 22)

```
for i, c in enumerate(string):
    if c == 'v'
        s[i] = '^'
    elif c == '^'
        s[i] = 'v'
string = ''.join(s)
string = string.replace('~~', '')
return f'[{string}]' if flag else string.

def skolemization(sentence):
    constants = [f'{chr(c)}' for c in range
                 (ord('A'), ord('Z')+1)]
statement = ''.join(list(sentence).copy()).
matches = re.findall('[Vx].', statement)
for match in matches[::-1]:
    statement = statement.replace(match, '')
~~for~~ statements = re.findall('\[[^\]]+\]',
                                statement)
for s in statements:
    statement = statement.replace(s, s[1:-1])
for predicate in get_predicates(statement)
    attributes = get_attributes(predicate)
    if ''.join(attributes).islower():
        statement = statement.replace(
            match[0], constant.pop(0))
```

```
else :
    aL = [a for a in attributes if a.is_lower()]
    aU = [a for a in attributes if a.is_upper()]
    ~~statement = statement.~~
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] + ']'
        statement = new_statement

for i, s in enumerate(statements):
    if '[' in s & ']' not in s:
        statement[i] += ']'

~~for~~
```

(PTO)

```
for s in statements:
    statement = statements.replace(s, fol_to_cnf(s))

while '-' in statement:
    i = statement.index('-')
    br = statement.index('[') if '[' in
         statement, else 0.
    new = '~' + statement[br:i] + 'v' +
          statement[i+1:].
    statement = statement[:br] + new if br>0
                else new.

return statement.

fol = input("Enter FOL sentence: \n").
print("\n The cnf form is :").
print(skolemization(fol_to_cnf(fol))).
```