Artificial Intelligence Lab Test - 2

Name: Anusree Manoj K
USN: 1BM18CS017
Section: 5A
Date: 29-12-2020

Program 4: Writeup

```
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~]+\([A-Z a-z, ]+\)'
    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)
    for i, c in enumerate(string):
```


Anusree

AI Lab Test - 2

Anusree Manoj K
IBM18CS017
27-12-2020

```
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):
    SKOLEM_CONSTANTS = [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(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))
```


Anusree

AI Lab Test-2

Anusree Manoj K
1BM18CS017
29-12-2020

```
else:
    aL = [a for a in attributes if a.islower()]
    aU = [a for a in attributes 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:i] + '=>' + statement[:i]
                        + ']'
    statement = new_statement
    expr = '\[([^]]+)\]'
    statements = re.findall(expr, 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))
```


Anusree

AI Lab Test - 2

Anusree Manoj K
1BM18CS017
29-12-2020

```
while '-' in statement:
    i = statement.index('-')
    br = statement.index('[') if '[' in statement else 0
    new_statement = '~' + statement[br:i] + 'V' +
                    statement[i+1:]
    statement = statement[:br] + new_statement if br > 0
                else new_statement

while '~∀' in statement:
    i = statement.index('~∀')
    statement = list(statement)
    statement[i], statement[i+1], statement[i+2]
        = '∃', statement[i+2], '~'
    statement = ''.join(statement)
while '~∃' in statement
    i = statement.index('~∃')
    s = list(statement)
    s[i], s[i+1], s[i+2] = '∀', s[i+2], '~'
    statement = ''.join(s)
statement = statement.replace('~[∀', '[~∀')
statement = statement.replace('~[∃', '[~∃')
expr = '([~[∀V∃],)'
statements = re.findall(expr, statement)
for s in statements:
    statement = statement.replace(s, fol_to_cnf(s))
```


Anusree

AI Lab Test 2

Anusree Manoj K
1BM18CS017
29-12-2020

```
    expr = '~\([^]]+\)'
    statements = re.findall(expr, statement)
    for s in statements:
        statement = statement.replace(s, DeMorgan(s))

    return statement

def main():
    print("enter FOL:")
    fol = input()
    print("The CNF form of the given FOL is:")
    print(Skolemization(fol_to_cnf(fol)))

main()
```


Anusree