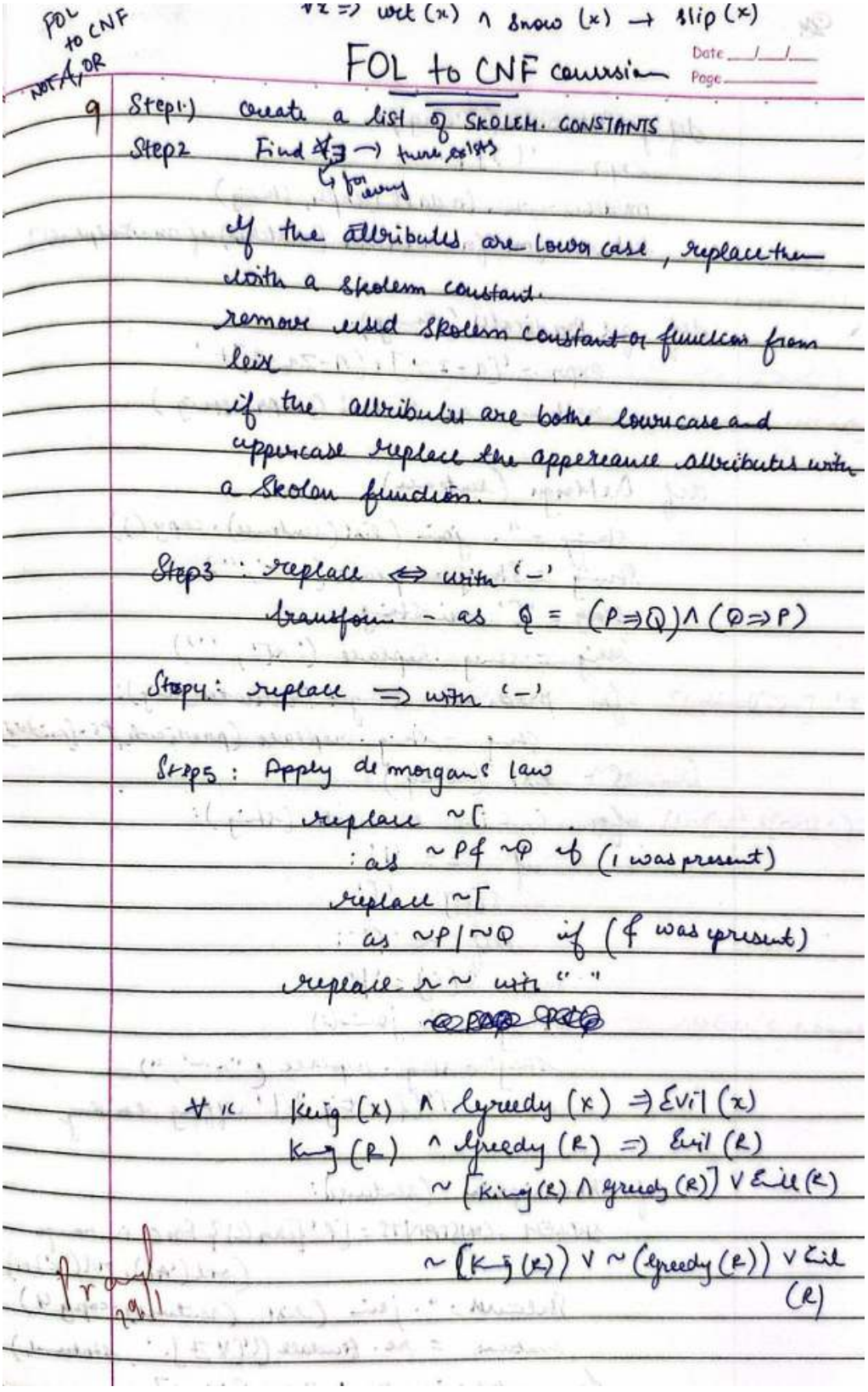
Fol to cnf
Code:-



FOL to CNF

$\forall x \Rightarrow wet(x) \wedge snow(x) \rightarrow slip(x)$

FOL to CNF conversion

Date __/__/__
Page

NOT, AND, OR

Step1:) Create a list of SKOLEM CONSTANTS

Step2 Find $\forall \exists$ → there exists
↳ for every

If the attributes are lowercase, replace them with a skolem constant.
remove used skolem constant or function from list
if the attributes are both lowercase and uppercase replace the appearance attributes with a Skolem function.

Step3 : replace $\Leftrightarrow$ with '→'
transform - as $P \Leftrightarrow Q = (P \Rightarrow Q) \wedge (Q \Rightarrow P)$

Step4: replace $\Rightarrow$ with '→'

Step5 : Apply de morgan's law
replace ~[
: as $\sim P \wedge \sim Q$ if (∧ was present)
replace ~[
: as $\sim P \vee \sim Q$ if (∨ was present)
replace ~~ with " "

$\forall x$ King(x) ∧ Greedy(x) ⇒ Evil(x)
King(R) ∧ Greedy(R) ⇒ Evil(R)
~[King(R) ∧ Greedy(R)] ∨ Evil(R)
~[King(R)) ∨ ~(Greedy(R)) ∨ Evil(R)

Proof

```
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)

def DeMorgan(sentence):
    string = ''.join(list(sentence).copy())
    string = string.replace('~~', '')
    flag = '[' in string
    string = string.replace('~[', '')
    for predicate in getPredicates(string):
        string = string.replace(predicate, f'~{predicate}')
    s = list(string)
    for i, c in enumerate(string):
        if c == '|':
            s[i] = '&'
        elif c == '&':
            s[i] = '|'
    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 reversed(matches[::-1]):
```

```
for predicate in get predicates (state):
    attributes = get Attributes (predicate)
    if "".join (attributes).isLower():
        Statement = Statement.replace (matches[1],
                                       Skolem_constants.pop(0))
    else:
        al = [a for a in attributes if a.isLower()]
        au = [a for a in attributes if not a.isLower()
              [0]
        return Statement

while '~' in Statement:
    i = Statement.index ('~∀')
    Statement = list (Statement)
    Statement[i] = , Statement [i+1], Statement[i+2]='∃
    Statement [i+2], '~'
    Statement = ''.join (Statement)
Statement = Statement.replace ('~[∀', '[~∀')
exps = '(~[∀|∃).)'
    for S in Statements:
        exps = '~'
    for S in statements:
        Statement = Statement.replace(S, Anyon
return Statement.
```

output:-

```
print(Skolemization(fol_to_cnf("animal(y)<=>loves(x,y)")))
print(Skolemization(fol_to_cnf("∀x[∀y[animal(y)=>loves(x,y)]]=>[∃z[loves(z,x)]]")))
print(fol_to_cnf("[american(x)&weapon(y)&sells(x,y,z)&hostile(z)]=>criminal(x)"))
```

```
[~animal(y)|loves(x,y)]&[~loves(x,y)|animal(y)]
[animal(G(x))&~loves(x,G(x))]|[loves(F(x),x)]
[~american(x)|~weapon(y)|~sells(x,y,z)|~hostile(z)]|criminal(x)
```