19/01/2024

Bafna Gold
Data: Page:

Lab - 8

(1) Unification :- Algorithm:

Eg: knows (John, x) knows (John, Jane)
{ x / Jane }

Step 1 : If term1 or term2 is a variable or constant then:

(a) term1 or term2 are identical return NIL

(b) Else if term is available
if term1 occurs in term2
return FAIL

(c) else if term 2 is a variable.
if term 2 occurs in term1
return FAIL
Else
return { (term 1 / term 2) }

(d) else return FAIL

Step 2 : If predicate (term1) ≠ predicate (term2)
return FAIL

Step 3 : number of argument ≠
return FAIL

Step 4 : set (SUBST) to NIL

Step 5 : For i ≥ 1 to the number of elements in term1 Ste

(a) call unify (ith term1 , ith term2)
put result into S

(b) S = FAIL
return FAIL

(c) If S ≠ NIL
(a). Apply S to the remainder of both
L1 and L2
(b) SUBST = APPEND (S, SUBST)

Step 6 : Return SUBST.

9

Code :-

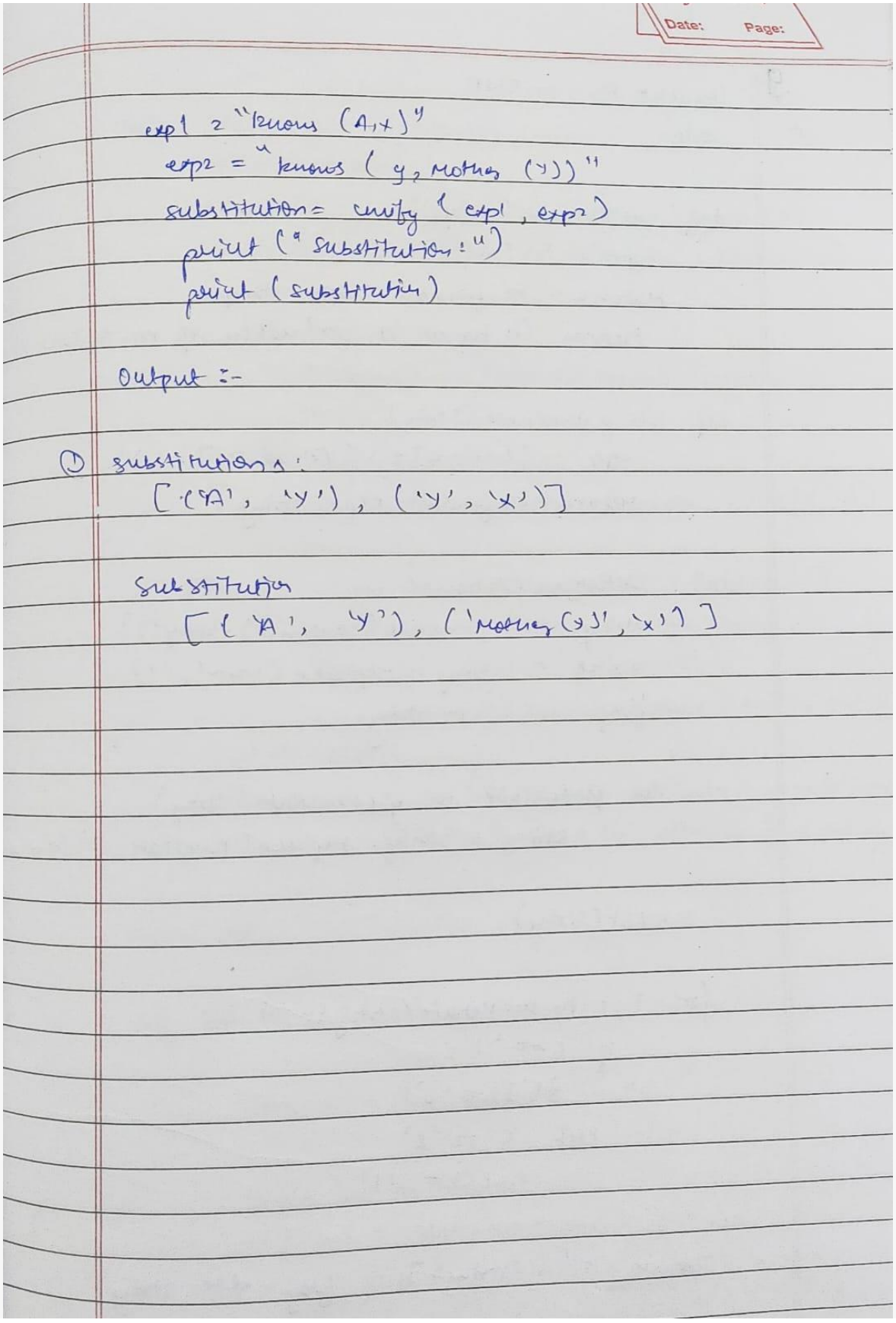
18 ⇒ unification :

```
(1)  import re

def getAttributes(expression):
    expression = expression.split("(")[1:]
    expression = "(".join(expression)
    expression = expression[:-1]
    expression = re.split("(?!\(.),(?!.\))", expression)
    return expression

def getInitial predicate(expression)
    return expression.split("(")[0]

def replaceAttributes(exp, old, new):
    attributes = getAttributes(exp)
    for index, val in enumerate(attributes):
        if val == old:
            attributes[index] = new
```

Bafna Gold
Date: Page:

```
predicate = getInitialPredicate(exp)
return predicate + "(" + ",".join(attributes) + ")"

def apply(exp, substitution):
    for substitution in substitution:
        new, old = substitution
        exp = replaceAttributes(exp, old, new)
    return exp.

def checkOccurs(var, exp):
    if exp.find(var) == -1:
        return false
    return true.

def getFirstPart(expression):
    attributes = getAttributes(expression)
    return attributes[0]

def unify(exp1, exp2):
    if exp1 == exp2:
        return []

    if isConstant(exp1) and isConstant(exp2):
        if exp1 != exp2:
            return false

    if isConstant(exp1)
        return [(exp2, exp1)]
```

```
if isConstant(exp2):
    return [(exp2, exp1)]

if getInitialPredicate(exp1) != getInitialPredicate(exp2):
    print("Predicates do not match")
    return False.

attributesCount1 = len(getAttributes(exp1))
attributesCount2 = len(getAttributes(exp2))

head1 = getFirstPart(exp1)
head2 = getFirstPart(exp2)

if not initialSubstitution:
    return False
if attributeCount1 == 1:
    return initialSubstitutions

tail1 = getRemainingPart(exp1)
tail2 = getRemainingPart(exp2)


exp1 = "knows(A,x)"
exp2 = "knows(y,y)"

Substitution = unify(exp1, exp2)
print("Substitutions:")
print(Substitutions)
```


Date:
Page:
exp1 = "knows (A,x)"
exp2 = "knows (y, Mother (y))"
substitution = unify (exp1, exp2)
print ("substitution:")
print (substitution)
Output :-
① substitution :
[('A', 'y'), ('y', 'x')]
Substitution
[('A', 'y'), ('Mother(y)', 'x')]

Output:

```
exp1 = "knows(X)"
exp2 = "knows(Richard)"
substitutions = unify(exp1, exp2)
print("Substitutions:")
print(substitutions)
```

```
Substitutions:
[('X', 'Richard')]
```

```
[ ] exp1 = "knows(A,x)"
exp2 = "knows(y,mother(y))"
substitutions = unify(exp1, exp2)
print("Substitutions:")
print(substitutions)
```

```
Substitutions:
[('A', 'y'), ('mother(y)', 'x')]
```